Programozási

Verseny-feladatok

IV.

Nemes Tihamér
Nemzetközi Informatikai Tanulmányi Verseny
2005-2009

Szerkesztette: Zsakó László

A Nemes Tihamér NITV feladatsorait

Horváth Gyula (SzTE)
Zsakó László (ELTE)

vezetésével az NJSzT Országos Versenybizottsága mindenkori tagjai dolgozták ki:

Gulyás László (ELTE-AITIA), Szabadhegyi Csaba (ELTE), Szlávi Péter (ELTE).

Ezen túl az ELTE számos hallgatója vett részt a feladatok kidolgozásában és a verseny lebonyolításában.

A kiadvány a Neumann János Számítógép-tudományi Társaság által kiadott *Programozási versenyfeladatok tára 4* című kötet alapján készült.

ISBN 978-963-489-102-4

SZÉCHENYI 2020
Európai Unió
Európai Szociális Alap
MAGYARORSZÁG KORMÁNYA
BEFEKTETÉS A JÖVŐBE

Előszó

Magyarországon több kezdeti próbálkozás után 1985-ben szervezte meg az első országos középiskolai programozási versenyt Nemes Tihamér Országos Középiskolai Számítástechnikai Tanulmányi Verseny (NTOKSzTV) néven a Neumann János Számítógép-tudományi Társaság (NJSZT). A 9 évig kétfordulós versenyen 1989-ig egy korcsoportban, 1990-től külön korcsoportban indulhatnak a 14-15, illetve a 16-19 éves tanulók (azaz a középiskolák I.-II., illetve III.-V. osztályos diákjai). Az 1993/94-es tanévtől a versenyt három fordulóban rendezzük (az iskolai forduló és az országos döntő közé iktattunk egy regionális-megyei fordulót). A verseny első tíz helyezettje – a többi országos középiskolai tanulmányi versenyhez hasonlóan – érettségi, illetve felvételi kedvezményben részesül, és közülük válogatjuk ki az 1989 óta ugyancsak évente megrendezett Nemzetközi Informatikai Diákolimpia magyar résztvevőit is.

A Nemes Tihamér verseny, szerénytelenség nélkül megállapíthatjuk, népszerűvé vált a diákok körében: az utóbbi években 260-300 magyarországi középiskola kb. 2500-2500 I.-II., illetve III.-V. osztályos diákja vett részt a verseny első, iskolai fordulójában. Közülük kb. 600 diák jutott tovább a második fordulóba, majd kb. 180 a harmadikba. Említést érdemel, hogy a környező országokban élő, magyar anyanyelvű vagy magyarul jól beszélő diákok közül is egyre többen érdeklődnek a verseny iránt, illetve vesznek részt a versenyen; az Erdélyi Magyar Műszaki Tudományos Társaság szervezésében évente kb. 500 diák indul.

1991 óta különböző szervezési formákban az általános iskolások is részt vehetnek országos programozási jellegű versenyen. A határon túli résztvevők, valamint az általános iskolások megjelenése miatt a versenyt átkereszteltük, új neve: Nemes Tihamér Nemzetközi Informatikai Tanulmányi Verseny.

Sokak igényét elégítettük ki azzal, hogy a Nemes Tihamér NITV-t e korosztállyal bővítettük, a megrendezésre jelentkező megyéket (regionális versenybizottságokat) feladatokkal láttuk el, s megszerveztük az országos döntőt is.

Az NJSZT Országos Versenybizottsága sokak kívánságát teljesíti most azzal, hogy önálló kötetekben megjelenteti az eddigi versenyfeladatokat. Reméljük, hogy ezzel is segíteni tudjuk a leendő versenyzőket a felkészülésben, a tanáraikat pedig a számítástechnikai foglalkozások és szakkörök megtartásában.

Ebben a kötetben a 2005-2009 közötti Nemes Tihamér versenyek programozási feladatait ismertetjük. Minden egyes feladat után a javító tanároknak szóló megoldási és értékelési útmutatót is közöljük. A példatárban egyedülálló módon a feladatok részletes megoldását is közöljük, amelyek eddig még sehol nem jelentek meg. A versenyekre készülő diákoknak természetesen azt javasoljuk, hogy mielőtt a közölt megoldást és értékelést elolvasnák, saját maguk, önállóan próbálják megoldani a kitűzött feladatot. Nem törekedtünk szöveghűségre: ahol az eredeti megfogalmazást pontatlannak vagy hibásnak találtuk, módosítottunk a szövegen.

Tartalom

I.

Verseny-feladatok

Nemes Tihamér
Nemzetközi Informatikai Tanulmányi Verseny

2005. Első forduló

Ötödik-nyolcadik osztályosok

1. feladat: Képletek (27 pont)

Az alábbi programok az X és az Y változók értékeit módosítják.

```
Első(X,Y):
  X:=X+Y; Y:=X-Y; X:=X-Y
Eljárás vége.
Második(X,Y):
  X:=X+Y; Y:=Y-X; X:=-X-Y
Eljárás vége.
Harmadik(X,Y):
  X:=X-Y; Y:=2*Y+X; Y:=X+Y; X:=Y-2*X
Eljárás vége.
```

A. Mi lesz az X és az Y változók értéke az alábbi eljáráshívások végrehajtása után, ha kezdetben X=2, Y=3?

B. Adj példát olyan X és Y változó értékekre, amelyeket az egyes eljárások nem változtatnak meg!

C. Fogalmazd meg képlettel, hogy az egyes eljárások hogyan határozzák meg X és Y új értékét a hívás előtti értékükből!

Példa:

```
    (X,Y) új értéke:=(X+Y,X+2*Y)
```

2. feladat: Számkitaláló (24 pont)

Az alábbi algoritmusok egy N természetes szám (N≥1) alapján állítják elő a K természetes számot.

```
Alfa(N,K):
  K:=0
  Ciklus amíg N>0
    K:=K+N mod 10; N:=N div 10
  Ciklus vége
Eljárás vége.
Béta(N):
  K:=0
  Ciklus amíg N>0
    K:=K+1; N:=N div 10
  Ciklus vége
Eljárás vége.
Gamma(N,K):
  K:=0
  Ciklus amíg N>0
    Ha K<N mod 10 akkor K:=N mod 10
    N:=N div 10
  Ciklus vége
Eljárás vége.
```

A. Mi lesz az egyes eljárások eredménye az alábbi N=1,8,11,2004 esetén?

B. Fogalmazd meg szövegesen, hogy az egyes algoritmusok mit írnak ki N értékétől függően!

3. feladat: Átrendezés (24 pont)

Az alábbi algoritmus az N elemű X tömb elemeit rendezi át.

```
Valami(X,N,Y):
  A:=1; B:=N
  Ciklus amíg A<B
    Ciklus amíg A<B és X(A)≤Y       {*}
      A:=A+1
    Ciklus vége
    Ciklus amíg A<B és X(B)≥Y       {**}
      B:=B-1
    Ciklus vége
    Ha A<B akkor Csere(X(A),X(B))
  Ciklus vége
Eljárás vége.
```

A. Mi lesz az X tömbben az eljárás végrehajtása után, ha előtte X=(4,5,2,7,3,1,6) és Y=4?

B. Mi lesz az X tömbben az eljárás végrehajtása után, ha előtte X=(1,7,5,4,3,6,2) és Y=4?

C. Milyen elemet keres a {*}-gal, illetve a {**}-gal jelölt ciklus?

D. Mi az eljárás feladata, azaz hogyan fognak elhelyezkedni egymáshoz képest, illetve önmagukon belül a végrehajtása után a tömbben az Y-nal egyenlő, Y-nál kisebb, illetve Y-nál nagyobb elemek?

4. feladat: Robot (25 pont)

Egy robot egy négyzetrácsos tábla bal felső sarkában áll. Egyszerre vagy jobbra, vagy pedig lefelé léphet egy mezőt. Egyes mezőkön gyöngyök vannak, másokon pedig csapdák. Ha a robot olyan mezőre lép, ahol gyöngy van, akkor az összes ott található gyöngyöt felveszi. Ha olyan mezőre lép, ahol csapda van, akkor csak úgy tud továbbmenni, ha az összes gyöngyét ott hagyja. A robotot el kell juttatni a jobb alsó sarokba úgy, hogy a végén lehető legtöbb gyöngy legyen nála!

A baloldali táblázatban R jelenti a robot helyét (itt még egyetlen gyöngy sincs nála), egész számok jelzik, hogy az egyes mezőkön hány gyöngy található, és X jelöli a csapdákat. Töltsd ki a jobboldali táblázatot úgy, hogy minden mezőbe azt a lehető legnagyobb számot írd, ahány gyöngye lehet a robotnak, bárhonnan is lép az adott mezőre!

R	1		1	5	
		1	X	X	X
		1			2
2	X			1	
	1	1			

0					
			0	0	0
	0				

Kilencedik-tizedik osztályosok

1. feladat: Mátrixelem (20 pont)

A következő két algoritmus a mellékelt mátrix egy elemének helyét adja meg (a sorokat föntről lefelé, az oszlopokat balról jobbra sorszámozzuk, 1-től kezdődően).

	1. oszl.									M. oszl.
1. sor	0	0	0	0	0	0	0	0	0	0
	0	1	2	3	2	3	2	3	2	0
	0	3	4	5	4	3	2	1	2	0
	0	5	4	6	7	2	1	2	3	0
	0	6	7	2	8	9	12	1	1	0
	0	5	8	9	12	8	7	6	3	0
	0	4	10	7	6	5	4	3	2	0
	0	3	1	1	2	3	4	5	1	0
	0	2	1	1	16	15	14	9	6	0
N. sor	0	0	0	0	0	0	0	0	0	0

```
Első:
  i:=2; j:=2; tovább:=igaz
  Ciklus amíg tovább
    p:=i; q:=j
    Ha T(i,j-1)>T(p,q) akkor p:=i; q:=j-1
    Ha T(i,j+1)>T(p,q) akkor p:=i; q:=j+1
    Ha T(i-1,j)>T(p,q) akkor p:=i-1; q:=j
    Ha T(i+1,j)>T(p,q) akkor p:=i+1; q:=j
    Ha i=p és j=q akkor tovább:=hamis különben i:=p; j:=q
  Ciklus vége
Eljárás vége.
```

```
Második:
  i:=2; j:=2; tovább:=igaz
  Ciklus amíg tovább
    Ha T(i,j+1)>T(i,j) akkor j:=j+1 különben
    Ha T(i+1,j)>T(i,j) akkor i:=i+1 különben
    Ha T(i,j-1)>T(i,j) akkor j:=j-1 különben
    Ha T(i-1,j)>T(i,j) akkor i:=i-1 különben tovább:=hamis
  Ciklus vége
Eljárás vége.
```

A. Milyen tulajdonságú elem esetén állnak meg az algoritmusok – az adott elem és a szomszédjai értékétől hogyan függ a megállás?

B. Mi lesz a futás végén az **i** és a **j** változó értéke az egyes algoritmusok esetén?

C. Adott **i** és **j** esetén a két algoritmus ciklusmagjában milyen tulajdonságú pont lesz a következő **i** és **j**?

D. Minek kell teljesülnie a mátrixra, hogy az algoritmusok a mátrixok belsejéből bárhonnan indulva biztosan a mátrix maximális elemét találják meg?

2. feladat: Keresés (18 pont)

Az alábbi programok az N elemű (N>1) 0 és 100 közötti egész számokat tartalmazó T tömb elemei alapján számítják ki S értékét.

```
Első(T,N,S):
  i:=N; S:=0
  Ciklus amíg S=0
    Ha T(i)=0 akkor S:=i
    i:=i-1
  Ciklus vége
Eljárás vége.
Harmadik(T,N,S,K):
  i:=1; D:=0
  Ciklus amíg D<K
    Ha T(i)=0 akkor D:=D+1
    i:=i+1
  Ciklus vége
  S:=i-1
Eljárás vége.
```

```
Második(T,N,S):
  i:=1; V:=igaz
  Ciklus amíg V vagy T(i)>0
    Ha T(i)=0 akkor V:=hamis
    i:=i+1
  Ciklus vége
  S:=i
Eljárás vége.
```

A. Mi lesz az S változó értéke az eljárások végrehajtása után, ha T=(3,0,7,0,0,3), K=3?

B. Milyen feltétel kell a T vektorra az egyes eljárásokban, hogy a ciklusmagban ne legyen indexhiba?

C. Fogalmazd meg szövegesen, hogy az egyes eljárások hogyan határozzák meg S értékét, ha nem állnak le futási hibával!

3. feladat: Lista (22 pont)

Egy ún. LISTA adatszerkezetet hozunk létre nevekből egy tömbben. A tömb minden eleme két értéket tartalmaz: egy nevet, s az ábécé sorrendben őt követő elem sorszámát. A legutolsó elemnél a következő elem sorszáma 0, a legelső elem sorszámát pedig a FEJ nevű változóban találjuk. A tömb üres (nem használt) elemei közül is ismerjük az első sorszámát (ÜRES nevű változó), s minden üres elem esetén ismerjük a következő üres elem sorszámát.

1.		3
2.	Lajos	6
3.		0
4.	Éva	2
5.	Alajos	8
6.	Pista	10
7.		9
8.	Barnabás	4
9.		1
10.	Zoli	0

A mellékelt tömb esetén FEJ=5, ÜRES=7. Az üres helyeken is lehet korábbról ottmaradt név.

Két műveletet definiálunk:

`Beszúr(NÉV)`: az üres elemek közül az elsőt lefoglalja, oda beírja a nevet, majd a listába beteszi az ábécé szerinti helyére.

`Töröl(NÉV)`: a listában megkeresi a nevet, ahol megtalálta, azt az elemet kiveszi a listából és beteszi az üresek közé az eddigi legelső üres hely elé (a nevet nem törli ki belőle).

A mellékelt listára a `Beszúr(Albert)`, `Töröl(Zoli)`, `Beszúr(Demeter)`, `Beszúr(Aladár)` műveleteket alkalmazzuk. Add meg az egyes műveletek elvégzése után a FEJ és az ÜRES változók értékét, valamint a tömb azon sorait, amelyek megváltoztak!

4. feladat: Újság (18 pont)

Egy újság minden számában egy 1 oldalas hirdetést jelentet meg. Hetenként vesznek fel hirdetési igényeket. Minden igénylő megadja, hogy mennyit fizet a hirdetéséért, ha adott sorszámú napig megjelenik az újságban. Úgy kell kiválasztani az egyes napokra a hirdetéseket, hogy az újságnak a lehető legnagyobb bevétele legyen.

A következő számpár sorozatokban a sorszámozott számpárok első tagja mindig a hirdetés legutolsó lehetséges megjelenési napja, a második pedig az érte fizetett összeg. Add meg mindegyikre, hogy mennyi belőlük az újság lehető legnagyobb bevétele, s az egyes napokon a felsorolás sorrendjében hányadik hirdetés jelenjen meg!

A. 1:(6,1000), 2:(3,200), 3:(6,1200), 4:(6,800), 5:(5,500), 6:(2,600), 7:(2,300), 8:(1,400), 9:(5,700), 10:(1,400)

B. 1:(2,1500), 2:(2,1200), 3:(2,1000), 4:(4,500), 5:(4,600), 6:(4,700), 7:(7,1000), 8:(7,800), 9:(7,200), 10:(7,100)

C. 1:(5,200), 2:(3,300), 3:(7,300), 4:(1,400), 5:(3,400), 6:(7,500), 7:(1,600), 8:(5,800), 9:(3,800), 10:(5,800)

Példa:

```
1:(7,1000), 2:(2,500), 3:(2,400), 4:(1,300), 5:(4,100)
```

esetén az 1,2,3,5 sorszámú hirdetések adják a legnagyobb bevételt, 2000 forintot, s egy lehetséges hirdetés elosztás a 7 napra: (3,2,-,5,-,-,1), azaz a hét első napján a 3. igénylő hirdetése jelenik meg, a hét utolsó napján pedig az 1. igénylőé. De sok más jó elosztás is van, pl. a 3 és a 2 felcserélhető, az 5 eggyel előbbre hozható, a 7-est is előre lehet hozni valamelyik üres helyre, …

5. feladat: Kocka (22 pont)

Építőkockából úgy lehet stabil tornyot építeni, hogy kisebb kockára nem lehet nagyobbat, illetve könnyebb kockára nem lehet nehezebbet tenni. Van 10 kockánk, a súlyuk szerint csökkenő sorrendbe rakva, melyek magassága: 10,7,6,4,11,3,8,14,5,9. Meg kell adni a belőlük építhető legmagasabb torony magasságát.

A feladat megoldásához töltsd ki az alábbi táblázatot, amelyben M(i) a legmagasabb olyan torony magassága, ahol az i-edik kocka van legfelül! A kitöltött táblázat alapján add meg a legmagasabb építhető torony magasságát!

M(1)	M(2)	M(3)	M(4)	M(5)	M(6)	M(7)	M(8)	M(9)	M(10)

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Balaton (16 pont)

A Balatonra egy négyzethálót fektettünk, s minden pontjára megadtuk, hogy az szárazföld (1-es érték) vagy pedig víz (0-s érték). A következő 4 algoritmus a mellékelt négyzetháló egy elemének helyét adja meg (a sorokat föntről lefelé, az oszlopokat balról jobbra sorszámozzuk, 1-től kezdődően).

```
Első:
  i:=1; j:=1
  Ciklus amíg i≤N és T(i,j)≠0
    Ha j=M akkor j:=1; i:=i+1 különben j:=j+1
  Ciklus vége
Eljárás vége.

Második:
  i:=1; j:=1
  Ciklus amíg j≤M és T(i,j)≠0
    Ha i=N akkor i:=1; j:=j+1 különben i:=i+1
  Ciklus vége
Eljárás vége.

Harmadik:
  i:=N; j:=M
  Ciklus amíg i≥1 és T(i,j)≠0
    Ha j=1 akkor j:=M; i:=i-1 különben j:=j-1
  Ciklus vége
Eljárás vége.
```

```
Negyedik:
  i:=N; j:=M
  Ciklus amíg j≥1 és T(i,j)≠0
    Ha i=1 akkor i:=N; j:=j-1 különben i:=i-1
  Ciklus vége
Eljárás vége.
```

	1. oszl.												M. oszl.
1. sor	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	1	1	1	1	1	1	1	1	1	1	0	1	1	1
	1	1	1	1	1	1	0	1	1	0	0	0	1	1
	1	1	1	1	0	0	0	1	0	0	0	0	0	1
	1	1	0	0	0	0	0	1	0	0	0	0	0	1
	1	0	0	0	0	0	0	0	0	0	0	0	0	1
	1	0	0	0	0	0	0	0	0	0	0	0	1	1
	1	1	0	0	0	0	0	0	0	0	1	1	1	1
	1	1	0	0	0	1	1	1	1	1	1	1	1	1
N. sor	1	1	1	1	1	1	1	1	1	1	1	1	1	1

A. Mi lesz a futás végén az i és a j változó értéke az egyes algoritmusok esetén?

B. Milyen tulajdonságú elem esetén állnak meg az algoritmusok?

<u>2. feladat</u>: Mit csinál? (20 pont)

Az alábbi algoritmus N és K (1<K<N) termés* * * zetes szám ismeretében N értéket olvas be, majd ír ki, csupán a sorrendjüket változtathatja meg.

```
Valami(N,K):
  Ciklus i=1-től K-ig       {*}
    Be: Y; j:=i-1
    Ciklus amíg j>0 és X(j)>Y
      X(j+1):=X(j); j:=j-1
    Ciklus vége
    X(j+1):=Y
  Ciklus vége

  Ciklus i=K+1-től N-ig     {**}
    Ki: X(1); Be: Y; j:=2
    Ciklus amíg j≤K és X(j)<Y
      X(j-1):=X(j); j:=j+1
    Ciklus vége
    X(j-1):=Y
  Ciklus vége
  Ciklus i=1-től K-ig       {***}
    Ki: X(i)
  Ciklus vége
Eljárás vége.
```

A. Mit ír ki, ha a bemenet: N=7, K=2, adatok: 2,1,3,4,6,5,7?

B. Mit ír ki, ha a bemenet: N=7, K=4, adatok: 4,3,2,1,7,6,5?

C. Mi a feladata az algoritmusnak (a példák alapján)?

D. Milyen feltételnek kell teljesülnie az X vektorra, hogy a példákból következő eredményt kapjuk

E. Mi a feladata a {*}-gal jelölt ciklusnak?

F. Mi a feladata a {**}-gal jelölt ciklusnak?

G. Mi a feladata a {***}-gal jelölt ciklusnak?

3. feladat: Lista (22 pont)

Egy ún. LISTA adatszerkezetet hozunk létre nevekből egy tömbben. A tömb minden eleme két értéket tartalmaz: egy nevet, s az ábécé sorrendben őt követő elem sorszámát. A legutolsó elemnél a következő elem sorszáma 0, a legelső elem sorszámát pedig a FEJ nevű változóban találjuk. A tömb üres (nem használt) elemei közül is ismerjük az első sorszámát (ÜRES nevű változó), s minden üres elem esetén ismerjük a következő üres elem sorszámát.

1.		3
2.	Lajos	6
3.		0
4.	Éva	2
5.	Alajos	8
6.	Pista	10
7.		9
8.	Barnabás	4
9.		1
10.	Zoli	0

A mellékelt tömb esetén FEJ=5, ÜRES=7. Az üres helyeken is lehet korábbról ottmaradt név.

Két műveletet definiálunk:

`Beszúr(NÉV)`: az üres elemek közül az elsőt lefoglalja, oda beírja a nevet, majd a listába beteszi az ábécé szerinti helyére.

`Töröl(NÉV)`: a listában megkeresi a nevet, ahol megtalálta, azt az elemet kiveszi a listából és beteszi az üresek közé az eddigi legelső üres hely elé (a nevet nem törli ki belőle).

A mellékelt listára a `Beszúr(Albert)`, `Töröl(Zoli)`, `Töröl(Alajos)`, `Beszúr(Aladár)` műveleteket alkalmazzuk. Add meg az egyes műveletek elvégzése után a FEJ és az ÜRES változók értékét, valamint a tömb azon sorait, amelyek megváltoztak!

4. feladat: Újság (18 pont)

Egy újság minden számában egy 1 oldalas hirdetést jelentet meg. Hetenként vesznek fel hirdetési igényeket. Minden igénylő megadja, hogy mennyit fizet a hirdetéséért, ha adott sorszámú napig megjelenik az újságban. Úgy kell kiválasztani az egyes napokra a hirdetéseket, hogy az újságnak a lehető legnagyobb bevétele legyen.

A következő számpár sorozatokban a sorszámozott számpárok első tagja mindig a hirdetés legutolsó lehetséges megjelenési napja, a második pedig az érte fizetett összeg. Add meg mindegyikre, hogy mennyi belőlük az újság lehető legnagyobb bevétele, s az újságban melyik hirdetések melyik napokon jelenjenek meg!

A. 1:(6,1000), 2:(3,200), 3:(6,1200), 4:(6,800), 5:(5,500), 6:(2,600), 7:(2,300), 8:(1,400), 9:(5,700), 10:(1,400)

B. 1:(2,1500), 2:(2,1200), 3:(2,1000), 4:(4,500), 5:(4,600), 6:(4,700), 7:(7,1000), 8:(7,800), 9:(7,200), 10:(7,100)

C. 1:(5,200), 2:(3,300), 3:(7,300), 4:(1,400), 5:(3,400), 6:(7,500), 7:(1,600), 8:(5,800), 9:(3,800), 10:(5,800)

Példa:

`1:(7,1000), 2:(2,500), 3:(2,400), 4:(1,300), 5:(4,100)`

esetén az 1,2,3,5 sorszámú hirdetések adják a legnagyobb bevételt, 2000 forintot, s egy lehetséges hirdetés elosztás a 7 napra: (1: 7. nap, 2: 2. nap, 3: 1. nap, 5: 4. nap). Sok más jó elosztás is van, pl. a 3. és a 2. felcserélhető, az 5. egy nappal előbbre hozható, az elsőt is előre lehet hozni valamelyik üres napra, …

5. feladat: Kocka (22 pont)

Építőkockából úgy lehet stabil tornyot építeni, hogy kisebb kockára nem lehet nagyobbat, illetve könnyebb kockára nem lehet nehezebbet tenni. Van 10 kockánk, a súlyuk szerint csökkenő sorrendbe rakva, melyek magassága: 10,7,6,4,11,3,8,14,5,9. Meg kell adni a belőlük építhető legtöbb kockából álló torony kockaszámát.

A feladat megoldásához töltsd ki az alábbi táblázatot, amelyben M(i) a legtöbb kockából álló olyan torony kockaszáma, ahol az i-edik kocka van legfelül! A kitöltött táblázat alapján add meg a legtöbb kockából álló torony kockaszámát!

M(1)	M(2)	M(3)	M(4)	M(5)	M(6)	M(7)	M(8)	M(9)	M(10)

2005. Második forduló

Ötödik-nyolcadik osztályosok

1. feladat: Számok franciául (27 pont)

A francia nyelvben a legfeljebb 3-jegyű számok írására az alábbi szabályok vannak:

A 0-16 számoknak önálló neve van – *zéro, un, deux, trois, quatre, cinq, six, sept, huit, neuf, dix, onze, douze, treize, quatorze, quinze, seize*.

A 17-19 számok a 10-ből és a 7..9 számok nevéből keletkeznek, kötőjellel – pl. *dix-sept*.

A 20,30,40,50,60 számoknak önálló neve van: *vingt, trente, quarante, cinqante, soixante*.

A 21,31,41,51,61 számokban a tízes helyiérték mögé kerül az *et un*, azaz pl. a 61 *soixante et un* lesz.

Ha nem egyesre végződik, akkor kötőjellel írjuk mögé az egyjegyű számot, azaz pl. a 27 *vingt-sept* lesz.

A 70 franciául *soixante-dix* (60+10), 71-től 79-ig úgy képződnek belőle a számok mint a 11-19 közötti kétjegyűek, azaz 71 – *soixante et onze*, 72 *soixante-douze*, 79 *soixante-dix-neuf*.

A 80 franciául *quatre-vingts* (4*20 – az s betű csak a 80-ban van), 81-től 99-ig ehhez jön hozzá az 1..19, azaz pl. 83 – *quatre-vingt-trois*, 88 – *quatre-vingt-huit*, 90 – *quatre-vingt-dix*, 97 – *quatre-vingt-dix-sept*.

Írj programot, amely beolvas egy 0 és 99 közötti egész számot, majd kiírja franciául!

2. feladat: Születésnap (28 pont)

Ismerjük N ember ($2 \le N \le 100$) születésnapját, mindegyikről tudjuk, hogy melyik hónap hányadik napján született.

Készíts programot, amely megad két embert, akinek a legközelebb van egymáshoz a születésnapja, illetve azt, hogy hány nap különbség van közöttük!

Ha több ilyen is lenne, akkor is elég egy ilyen párt megadni! Az embereket a beolvasásban megadott sorrend azonosítja, azaz az 1-es sorszámú az elsőnek beolvasott ... A megoldás során feltehető, hogy szökőév nincs, azaz az év 365 napos.

Példa:

Bemenet:	Kimenet:
N=5 3 10 6 8 1 15 10 3 12 31	emberek: 3 5 különbség: 15 nap

3. feladat: Szövegjavítás (20 pont)

A helyesírás szabályai szerint szövegben írásjelek elé nem teszünk szóközt (vessző, pont, kérdőjel, felkiáltójel, pontosvessző, kettőspont), utána viszont teszünk, de csak egyet. A szövegben sehol nem lehet egymás mellett egynél több szóköz.

Készíts programot, amely beolvas egy szöveget, majd kiírja a képernyőre a helyesírás szabályai szerint!

Példa:

```
Bemenet: Ez egy   hibásan ,rosszul írt szöveg    .
Kimenet: Ez egy hibásan, rosszul írt szöveg.
```

Kilencedik-tizedik osztályosok

1. feladat: Gazda (10 pont)

Egy gazdának van néhány útja, amelyek mentén nyárfákat ültetett. Minden két szomszédos nyárfa közé pontosan egy szilvafát ültetett.

A gazda halála előtt a következőt mondta legidősebb fiának: kiválaszthatsz magadnak N darab nyárfát, és a tiéd lesz minden két szomszédos nyárfád közötti szilvafa is! A fiú úgy szeretné kiválasztani az N darab nyárfát, hogy a lehető legtöbb fája legyen.

Az 1. út 4 nyárfával (a körök a nyárfák, a szilvafák a köztük levő szakaszokon vannak)

A 2. út 8 nyárfával

A 3. út 6 nyárfával

Készíts programot, amely az utak és a kiválasztható nyárfák számának ismeretében kiszámítja, hogy a fiúnak a legjobb esetben hány szilvafája lesz!

A *GAZDA.BE* állomány első sora a kiválasztható nyárfák számát ($1 \le N \le 10\,000$) és az utak számát ($1 \le K \le 1000$) tartalmazza. A második sorban K darab egész szám van, az egyes utak melletti nyárfák száma.

A *GAZDA.KI* állomány egyetlen sorába azt a számot kell írni, ahány szilvafája lehet a fiúnak a legjobb esetben!

Példa:

GAZDA.BE	GAZDA.KI
10 3 4 8 6	8

2. feladat: Javít (20 pont)

A helyesírás szabályai szerint szövegben írásjelek elé nem teszünk szóközt (vessző, pont, kérdőjel, felkiáltójel, pontosvessző, kettőspont), utána viszont teszünk. Ugyancsak nem teszünk szóközt a nyitó zárójelek után és a csukó zárójelek elé (háromféle van: gömbölyű, szögletes és kapcsos). A nyitó zárójel elé kell tenni szóközt és a csukó után is kell, hacsak nem írásjel követi. Ha egyszerre két szabályt kellene alkalmazni (pl. írásjel után kell, csukó zárójel előtt nem kell szóköz), akkor a tiltó szabály az erősebb.

Készíts programot, amely egy szöveget átalakít a helyesírás szabályai szerint!

A *JAVIT.BE* állományban egyetlen sor van (legfeljebb 10 000 karakter), amely a hibásan beírt szöveget tartalmazza.

A *JAVIT.KI* állományba egyetlen sort kell írni, a javított szöveget.

Példa:

```
JAVIT.BE:
Ez egy hibásan ,rosszul( rossz zárójelezéssel)írt szöveg  .
JAVIT.KI:
Ez egy hibásan, rosszul (rossz zárójelezéssel) írt szöveg.
```

3. feladat: Pakolás (15 pont)

Egy konténer raktárban egy sorban tárolják a konténereket, ahol `N` darab konténernek van hely. Mivel a konténerek kiszállítása az igénynek megfelelően történik, ezért egy adott időpontban a raktárban lévő `M` konténer tetszőlegesen helyezkedik el. A raktárosnak időnként át kell rendezni a raktárban lévő konténereket úgy, hogy folyamatosan egymás mellett legyenek. Az átrendezést bizonyos konténerek egyesével történő átrakásával lehet végezni. Egy `i`-edik konténerhelyen lévő konténert a `j`-edik konténerhelyre csak akkor rakhatunk át, ha köztük minden konténerhely üres. Ha az átrendezés során az `i`-edik konténerhelyen lévő konténert a `j`-edik konténerhelyre rakja át, ennek költsége `i-j` abszolút értéke. A raktáros az optimális átrendezést keresi, tehát amelyre az összköltség minimális.

Készíts programot, amely kiszámítja a raktár optimális átrendezésének költségét!

A *pakol.be* állomány első sora a raktár konténerhelyeinek számát tartalmazza ($2 \le N \le 10\,000$). A második sor pontosan `N` egész számot tartalmaz. A sorban minden szám vagy `0`, vagy `1`. A raktárban az *i*-edik helyen akkor és csak akkor van konténer, ha a második sorban az `i`-edik szám `1`. Legalább egy konténer van a raktárban.

A *pakol.ki* állomány első sora egyetlen számot tartalmazzon, az optimális átrendezés költségét! A második sor is egyetlen számot tartalmazzon, a legkisebb konténerhelynek a sorszámát, amely az átrendezés után konténert tartalmaz! Több megoldás esetén bármelyik kiírható.

Példa:

pakol.be	pakol.ki
10	8
1 0 1 1 0 0 1 1 0 1	2

4. feladat: Ajándék (15 pont)

Osztályod Mikulás ünnepségre készül. Minden tanuló választott magának egy másik tanulót, akinek ajándékot szeretne adni. Kiderült, hogy így lehet olyan tanuló, aki nem kap senkitől sem ajándékot. Az osztály felhatalmazta az osztályfőnököt, hogy a lehető legkevesebb módosítást elvégezze, hogy mindenki továbbra is egy ajándékot adjon és pontosan egy ajándékot kapjon.

Készíts programot, amely kiszámítja a szükséges legkevesebb módosítások számát, amelynek hatására mindenki pontosan egy ajándékot kap, és meg is ad egy ilyen módosítást!

A `mikulas.be` állomány első sora a tanulók számát tartalmazza (2≤N≤1000). A második sor pontosan N egész számot tartalmaz. A sorban az i-edik szám annak a tanulónak a sorszáma, akinek az i-edik tanuló ajándékot szeretne adni (saját magának biztosan nem ad).

A `mikulas.ki` állomány első sora egyetlen számot tartalmazzon, a lehető legkevesebb szükséges módosítások K számát! A következő K sor mindegyike egy-egy módosítást tartalmazzon, két egész számot: i j! Ez azt jelenti, hogy a módosítás következtében az i-edik tanuló a j-edik tanulónak fog ajándékot adni. Több megoldás esetén bármelyik kiírható.

Példa:

mikulas.be

```
9
2 3 4 5 3 5 8 9 8
```

mikulas.ki

```
3
5 6
6 1
9 7
```

5. feladat: Vállalat (15 pont)

Egy vállalat központjában a munkatársak olyan beosztásban dolgoznak, hogy a vállalat igazgatóján kívül mindenkinek pontosan egy főnöke van, és mindenkinek legfeljebb két közvetlen beosztottja van. Mindenki csak a közvetlen beosztottjának adhat utasítást. Az igazgató azt akarja tudni, hogy kik azok a munkatársak, akikhez az általa kiadott utasítás pontosan K lépésben jut el.

Készíts programot, amely a beosztotti viszonyok és K ismeretében meghatározza azokat a munkatársakat, akikhez az igazgató utasítása pontosan K lépésben jut el!

A `vallalat.be` állomány első sora a munkatársak számát (2≤N≤1000) és a kérdésben szereplő K számot tartalmazza. A munkatársakat az 1,…,N számokkal azonosítjuk, az igazgató sorszáma 1. A további N sor mindegyike a munkatárs két közvetlen beosztottjának sorszámát tartalmazza. Ha csak egy beosztottja van, akkor az egyik szám 0, ha egy sincs, akkor mindkettő 0. Az i+1-edik sorban az i-edik munkatárs közvetlen beosztottjai vannak.

A `vallalat.ki` állomány első sora egyetlen számot tartalmazzon, azoknak a munkatársaknak az M számát, akikhez az igazgató utasítása pontosan K lépésben jut el! A második sor pontosan M számot tartalmazzon, a kérdésben szereplő munkatársak sorszámát!

Példa:

vallalat.be

```
10 2
2 3
8 7
4 0
5 6
0 0
0 0
0 0
9 10
0 0
0 0
```

vallalat.ki

```
3
8 7 4
```

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Fák (14 pont)

Egy fasorba N fát ültettek balról jobbra, egy vonalba. Mindegyik fának ismerjük a magasságát és a bal szélső fáról vett távolságát. Ha egy fát kivágunk, akkor az a jobboldali szomszédja felé dől, s amelyik szomszédjára rádől, az is kidől.

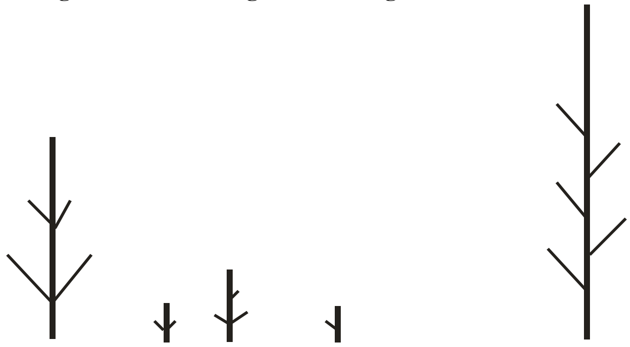
Írj programot, amely megadja, hogy minimum hány fát kell kivágnunk ahhoz, hogy az összes fa kidőljön, s melyik fa kivágása okozza a legtöbb fa kidőlését!

A `fak.be` állomány első sorában a fák száma van (1≤N≤1000). A következő N sor mindegyike egy-egy fa leírását tartalmazza: a bal szélső fától vett távolságát (0≤T≤1000) és a fa magasságát (1≤M≤100). A fákat balról jobbra haladva adjuk meg.

A `fak.ki` állomány első sorába a minimálisan kivágandó fák számát kell írni, a második sorába pedig annak a kivágandó fának a sorszámát, amely kivágása esetén a legtöbb fa fog kidőln!.

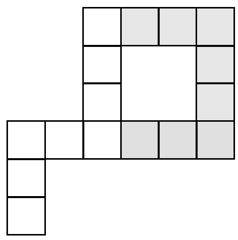
Példa:

```
fak.be          fak.ki
5               3
0 6             1
3 1
5 2
8 1
15 10
```



2. feladat: Pince (16 pont)

Pincét fúrnak: az E,J,B utasítások hatására egy egységnyi részt fúrnak, jobbra, illetve balra fordulnak 90 fokkal. A ()-ben levő részek a főágról leágazást jelentenek, előbb mindig a főágat fúrják végig, s utána kezdenek az elágazásokhoz. Szabályok: két ág nem érhet össze, sőt a sarkával sem találkozhat. Önmagában minden lépés biztosan olyan, hogy fúrni is kell, azaz teljesen nem mennek bele már kifúrt részbe (pl. nincs olyan sorozat, hogy `EJJE`).



Írj programot, amely megadja, hogy a pince helyes-e, s ha nem, akkor melyik lépésben találkoznak össze az ágak!

A `pince.be` állomány egyetlen sorában a pincét leíró karaktersorozat van (legfeljebb 10 000 karakter a sor végéig). A pincefúrás közben az X- és az Y-koordináta nem lép ki a [-100..100] intervallumból. A kezdőkoordináta a (0,0).

A `pince.ki` állományba egyetlen számot tartalmazó sort kell írni: 0-t, ha a pince helyes, illetve az I számot, ha a pincét leíró karaktersorozatban az I-edik karakter hatására fúrt szakasznál találkoznak össze az ágak!

Példa:

```
pince.be                            pince.ki
EEEJEEB(JEEEBEEEBEE)EEE      19
```

A példában az elágazás ásásának utolsó lépésében ütközünk a főág végébe, szürkítetten szerepel az elágazás.

3. feladat: Osztály (15 pont)

Egy osztályba N tanuló jár. Minden tanuló ismeri néhány osztálytársának telefonszámát.

Írj programot, amely megadja azt a tanulót, akitől egy hír az ismert telefonokon keresztül továbbadva előbb-utóbb az osztály legtöbb tanulójához eljut!

Az `osztaly.be` állomány első sorában a tanulók száma (1≤N≤100) van. A következő N sor mindegyike egy-egy tanuló által ismert telefonszámú tanulókat ír le, az állomány i+1-edik sorában azoknak a tanulóknak a sorszáma van, akikét az i-edik tanuló ismeri. Mindegyik sorban legfeljebb N-1 különböző egész szám van, és 0-val zárva: az ismert telefonszámú tanulók sorszáma.

Az `osztaly.ki` állományba egyetlen sort kell írni, annak a tanulónak a sorszámát, akitől a legtöbb tanulóhoz eljuthat egy hír! Ha több ilyen tanuló van, akkor bármelyik sorszáma kiírható.

Példa:

`osztaly.be`

```
5
3 5 0
3 4 0
0
2 3 0
2 0
```

`osztaly.ki`

```
1
```

4. feladat: Vonat (15 pont)

Egy hosszú vasútvonal mentén `N` város helyezkedik el, minden városnak pontosan egy vasútállomása van a vonalon. Ismerjük a vonalon közlekedő vonatokat. Minden vonat adott i-edik városból indul és adott j-edik városba közlekedik és közben nem áll meg egyetlen közbülső állomáson sem. Az 1. városból indulva, vonattal közlekedve a lehető legtöbb várost szeretnénk meglátogatni.

Írj programot, amely kiszámítja, hogy az `1.` városból indulva mennyi a legtöbb meglátogatható város, és meg is ad egy útvonalat, amelyen haladva a legtöbb város meglátogatható!

A `vonat.be` állomány első sorában a városok száma ($1 \le N \le 200$) és a járatok száma ($1 \le M \le 3000$) van. A további `M` sor mindegyike két egész számot tartalmaz, a járat indulási és érkezési állomását ($1 \le Ind_i < Érk_i \le N$).

A `vonat.ki` állomány első sorába egyetlen egész számot kell írni, a legtöbb meglátogatható város `K` számát, beleértve az `1.` induló várost is! A második sor pontosan `K-1` számot tartalmazzon, a járatok bemenetbeli sorszámait az utazás sorrendjében! Több megoldás esetén bármelyik kiírható.

Példa:

`vonat.be`

```
5 7
1 2
1 3
2 4
3 5
2 3
4 5
3 4
```

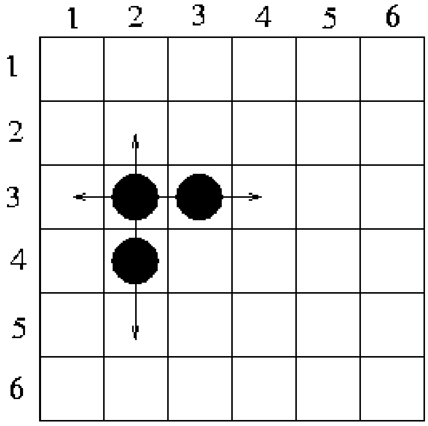
`vonat.ki`

```
5
1 5 7 6
```

5. feladat: Játék (15 pont)

Tekintsük a Solitaire játéknak azt a változatát, amelyet ***6x6***-os négyzetrácsos táblán lehet játszani. A táblára három fekete korongot helyeznek három különböző mezőre, ez a kezdeti játékállás. A játék során minden lépésben egy korongot lehet mozgatni az alábbi szabály szerint.

- Csak üres mezőre lehet lépni.
- A négy szomszédos mező valamelyikére lehet lépni, balra, jobbra, felfelé vagy lefelé.
- Ha a lépés irányába eső szomszédos mezőn van korong, akkor azt az egy korongot át lehet lépni.



A (3,2) mezőn álló korong négy lehetséges lépése: (2,2), (3,1), (5,2), (3,4), mint az ábrán látható.

Írj programot, amely kiszámítja, hogy adott kezdeti játékállásból legkevesebb hány lépés végrehajtásával lehet eljutni adott végállásba!

Az `jatek.be` két sort tartalmaz, az első sor a kezdeti játékállást, a második pedig a végállást írja le. Mindkét sor 6 egész számot tartalmaz, a három korong koordinátáit. Az i-edik (i=1,2,3), számpár az i-edik korong sor, illetve oszlopkoordinátáját jelenti. A sorokat fentről lefelé, az oszlopokat balról jobbra sorszámozzuk 1-től 6-ig. A három korong sorrendje közömbös a végállásban!

Az `jatek.ki` állomány első és egyetlen sora egy egész számot tartalmazzon, azon legkevesebb lépések számát, amennyi lépéssel el lehet jutni a kezdeti játékállásból a végállásba!

Példa:

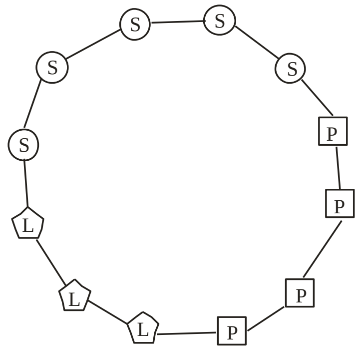
jatek.be	jatek.ki
3 2 3 3 4 2 2 3 3 3 3 4	3

2005. Harmadik forduló

Ötödik-nyolcadik osztályosok

1. feladat: Nyaklánc (25 pont)

Egy nyakláncon különböző színű gyöngyök találhatók. Tudjuk, hogy minden szín kezdőbetűje más, így a gyöngyöket a színük kezdőbetűjével adjuk meg. A lánc gyöngyeit egy tetszőleges pontjától kezdve adjuk meg.



Készíts programot, amely beolvassa a gyöngyök számát (1≤N≤100), majd az N gyöngy színét, s ezek alapján kiírja, hogy hány egyforma gyöngyből áll és milyen színű az egymás mellett levő leghosszabb egyszínű gyöngyökből álló szakasz!

Példa:

```
Bemenet: N=12, gyöngyök: SSLLLPPPPSSS
Kimenet: 5 gyöngy, S színű
```

2. feladat: Csapat (25 pont)

Egy kézilabdacsapatban egy mérkőzésen N játékos játszhat (7≤N≤14), de egyszerre 7 játékos lehet a pályán. Ismerjük, hogy milyen sorszámú játékosok kezdik a mérkőzést, valamint azt, hogy mikor cseréltek (ki helyére ki jött be).

Készíts programot, amely beolvassa a játékosok számát, a kezdetben a pályán levő 7 játékos sorszámát, majd a cserék számát (1≤M≤100), majd az M cserét. Minden cseréről tudjuk, hogy hányadik percben, milyen sorszámú játékos helyére milyen sorszámú játékos jött be. A program ezek alapján írja ki, hogy melyik játékos hány percet töltött a pályán! (A mérkőzés 60 perces.)

Példa:

```
Bemenet: N=10; Kezdőcsapat=1,2,3,4,5,6,7; M=5;
cserék: (5. perc,1,10),(5. perc,2,8),(10. perc,10,1),
(30. perc,3,10),(35. perc,6,9)
Kimenet: játékos    perc
              1     55
              2     5
              3     30
              4     60
              5     60
              6     35
              7     60
              8     55
```

```
9       25
10      35
```

3. feladat: Állomások (25 pont)

Egy nemzetközi vonat több napon keresztül megy az egyik végállomásáról a másik végállomására. Ismerjük minden állomásra az érkezési időt. A vonat a végállomás kivételével minden állomáson pontosan 10 percet várakozik, majd továbbindul.

Készíts programot, amely beolvassa az állomások számát (2≤N≤100), a kezdő állomásról indulási időt (0≤óra≤23, 0≤perc≤59), majd pedig a további N-1 állomásra érkezési időt (óra, perc). A program ezekből számítsa ki, hogy

A. hány perc volt a leghosszabb időszak, amikor a vonat sehol sem állt meg;

B. a vonat mely állomások között haladt (vagy mely állomásokon állt) éjfélkor!

Megjegyzés: Feltehető, hogy két szomszédos állomás közötti menetidő kisebb 24 óránál!

Példa:

```
Bemenet: N=7; indulás=9 óra 20 perc; érkezések: (13,30), (19,45), (4,00), (16,30), (23,55), (6,30).
Leghosszabb menetidő: 740 perc  {a 4. és az 5. állomás között}
Éjfélkor: 3.-4. állomás között halad
Éjfélkor: 6. állomáson áll
```

Kilencedik-tizedik osztályosok

1. feladat: Verseny (18 pont)

Egy csapatversenyben N csapat vesz részt, a csapatokat 1 és N közötti sorszámukkal azonosítjuk. Ismerjük M mérkőzés eredményét.

Írj programot az alábbi feladatokra!

A. Adj meg két csapatot, amelyek már legyőzték egymást!

B. Add meg azokat a csapatokat, akik már játszottak, de még senki nem győzte le őket!

C. Adj meg egy csapatot, amely „közvetve legyőzte magát” (azaz pl. A ilyen, ha A legyőzte B-t, B legyőzte C-t, ..., Y legyőzte Z-t és Z legyőzte A-t)!

A *VERSENY.BE* állomány első sorában a csapatok száma (1≤N≤100) és a mérkőzések száma (1≤M≤10 000) van. A következő M sorban egy-egy mérkőzés eredménye van: két szám (i és j). Jelentése: az i-edik csapat legyőzte a j-edik csapatot.

A *VERSENY.KI* állomány első sorába a két egymást legyőző csapat sorszámát, a második sorba az összes B. feladatban kért csapat sorszámát ki kell írni! A harmadik sorba egy önmagát közvetve legyőző csapat sorszáma kerüljön! Mind a három sorra igaz, hogy ha nincs megfelelő csapat, egy üres sornak akkor is ki kell kerülnie az állományba!

Példa:

VERSENY.BE	VERSENY.KI
6 7	1 3
1 2	5
1 3	1
3 1	
2 4	
4 1	
5 2	
5 3	

2. feladat: Királyok (18 pont)

Árpád-házi királyokról tároljuk születési, illetve uralkodási adataikat. Csak olyan esetet vizsgálunk, amikor folyamatosan volt király, egyszerre csak egy, és mindegyik király csak egy időintervallumban uralkodott.

Készíts programot, amely megadja az alábbiakat!

A. Mettől meddig volt a leghosszabb időszak, amikor olyan uralkodók voltak, akik az apjukat követték a trónon (beleszámítva a legelsőt, aki még nem az apját követte)!

B. Hányan voltak legtöbben testvérek, akik mindegyike uralkodott valamikor!

C. Hány olyan király volt, akinek volt gyereke, de egyik sem lett király!

A *KIRALY.BE* állomány első sorában az uralkodók száma van (1≤N≤100). A következő N sor mindegyike 3 adatot tartalmaz: a király uralkodásának kezdő- és végző évét időrendben, valamint a nevét (a nevek biztosan különbözőek). A következő sorban az Árpád-ház családtagjai leszármazási kapcsolatai száma van (1≤M≤1000). Az ezt követő 2*M sor páronként egy-egy szülői kapcsolatot tartalmaz: a pár első tagja a szülő, a második pedig a gyerek nevét.

A *KIRALY.KI* állomány első sorbáa 2 időpontot kell írni: az A feladatban kért időszak kezdő- és végző évét! Ha senki sem az apját követte a trónon, akkor ez a sor legyen üres! A második sorba a B feladat megoldása kerüljön! A harmadik sorba a C feladatban kért királyok számát kell írni!

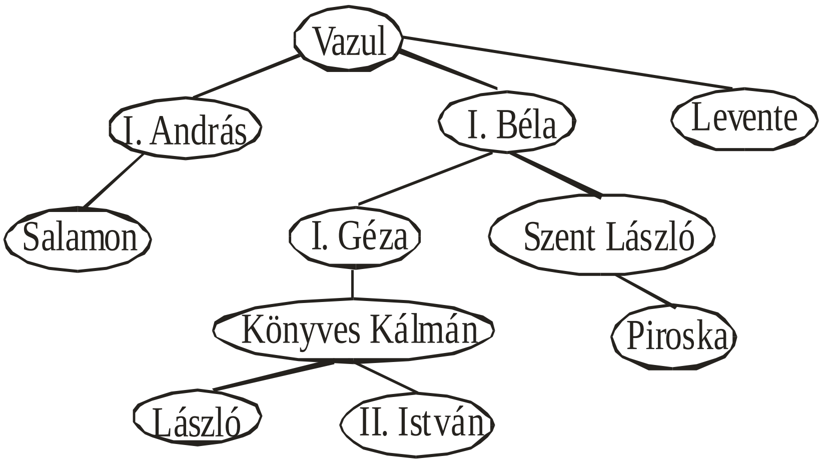
Példa:

KIRALY.BE

```
7
1046 1060 I. András
1060 1063 I. Béla
1063 1074 Salamon
1074 1077 I. Géza
1077 1095 Szent László
1095 1116 Könyves Kálmán
1116 1131 II. István
10
Vazul
I. András
Vazul
I. Béla
Vazul
Levente
I. András
Salamon
I. Béla
I. Géza
I. Béla
Szent László
I. Géza
Könyves Kálmán
Szent László
Piroska
Könyves Kálmán
László
Könyves Kálmán
II. István
```

KIRALY.KI

1095 1131	{Könyves Kálmán és fia}
2	{pl. I. András és I. Béla}
1	{Szent László lánya}



3. feladat: Úthálózat (21 pont)

Egy város utcái NxM-es négyzethálós elrendezésűek. Minden kereszteződésben pontosan 4 utca találkozik, kivéve esetleg a város szélén levő kereszteződéseket. A kereszteződéseket a sor-, illetve az oszlop-koordinátájuk adja meg, mindkettőt 1-től sorszámozzuk, a bal felső sarok az (1,1) koordinátájú pont. Egyes utcák egyirányúak, mások pedig kétirányúak. Az utcák irányítottságát a kereszteződésekben adjuk meg, egy-egy 8 bites számmal az alábbi táblázat szerint (a biteket hátulról sorszámozva):

1. bit: 1-es, ha a felfelé vezető utcából lehet a kereszteződésbe jutni (lefelé irány)

2. bit: 1-es, ha a felfelé vezető utcába lehet menni a kereszteződésből (felfelé irány)

3. bit: 1-es, ha a jobbra vezető utcából lehet a kereszteződésbe jutni (balra irány)

4. bit: 1-es, ha a jobbra vezető utcába lehet menni a kereszteződésből (jobbra irány)

5. bit: 1-es, ha a lefelé vezető utcából lehet a kereszteződésbe jutni (felfelé irány)

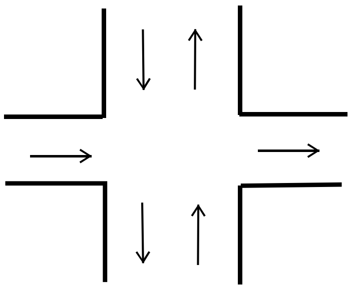
6. bit: 1-es, ha a lefelé vezető utcába lehet menni a kereszteződésből (lefelé irány)

7. bit: 1-es, ha a balra vezető utcából lehet a kereszteződésbe jutni (jobbra irány)

8. bit: 1-es, ha a balra vezető utcába lehet menni a kereszteződésből (balra irány)

Készíts programot, amely a kereszteződések ismeretében megadja, hogy a tervezett úthálózat mely kereszteződésekben hibás!

Példa:



bináris kódja 01111011,

A lehetséges – egyszerű – hibák:

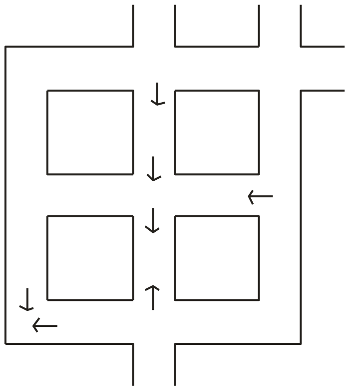
- a kereszteződésből semerre sem lehet kimenni;
- a kereszteződésbe sehonnan nem lehet bemenni;
- a kereszteződés valamelyik utcáján sem bejönni, sem kimenni nem lehet;
- egy utca két végpontján levő kereszteződésekben ellentmondó adatok vannak az utcáról (ekkor mindkét kereszteződés hibás!);
- a városból sehol sem lehet kijutni (ekkor a (0,1) kereszteződést tekintjük hibásnak;
- a városba sehol sem lehet kívülről bejönni (ekkor az (1,0) kereszteződés a hibás).

Az *UT.BE* állomány első sorában az úthálózatot leíró négyzetháló sorai (`1≤N≤100`) és oszlopai (`1≤M≤100`) száma van. A következő N sor mindegyike pontosan M számot tartalmaz: az egyes kereszteződéseket leíró 8 bites számok tízes számrendszerbeli alakját.

Az *UT.KI* állomány első sorába a hibás kereszteződések K számát kell írni! A következő K sor mindegyikébe egy-egy hibás kereszteződés pozícióját kell írni! A pozíció a kereszteződés sor- (`1≤sor≤N`) és oszlopindexe (`1≤oszlop≤M`), vagy pedig az 1 0, illetve a 0 1 kód (az utolsó két hiba esetén)! Ha egy kereszteződés több szempontból is hibás, akkor is csak egyszer szabad kiírni!

Példa: (az ábrán csak az egyirányú utakat jelöljük)

UT.BE	UT.KI
3 3	5
60 239 255	2 1
63 237 179	2 2
5 254 195	2 3
	3 1
	3 2



4. feladat: Zenekar (18 pont)

Egy népszerű zenekar a következő évre vonatkozó fellépéseit tervezi. Sok meghívása van fellépésre, ezek közül kell a zenekarnak választani, hogy melyeket fogadja el. Minden fellépés pontosan egy napot foglal el. Minden beérkezett meghívási igény egy (e,u) számpárral adott, ami azt jelenti, hogy az igénylő azt szeretné, hogy a zenekar olyan k sorszámú napon tartson nála koncertet, hogy e≤k≤u. A zenekarnak az a célja, hogy a lehető legtöbb fellépése legyen.

Készíts programot, amely kiszámítja, hogy mely meghívásokat fogadja el, hogy a következő évben a lehető legtöbb fellépése legyen, és a programod adjon is meg egy beosztást!

A *ZENEKAR.BE* állomány első sorában a meghívások száma (1≤N≤1000) van. A következő N sor mindegyike egy-egy meghívás lehetséges intervallumát tartalmazza (1≤e≤u≤365).

A *ZENEKAR.KI* állomány első sora a vállalható legtöbb fellépések M számát tartalmazza! A következő M sor mindegyikének első száma egy elfogadott meghívás sorszáma legyen, a második pedig annak a napnak a sorszáma, amelyik napon teljesíti a zenekar a fellépést! Több megoldás esetén bármelyik megadható.

Példa:

ZENEKAR.BE	ZENEKAR.KI
6	5
2 4	4 1
1 4	2 2
3 5	3 3
1 3	5 4
3 5	6 5
2 5	

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Konténer (15 pont)

Egy konténer raktárban N db konténer van egy sorban tárolva. A konténereket el akarják szállítani, ezért mindegyikre rá van írva, hogy melyik városba kell szállítani. A városokat 1-től 3-ig sorszámozzák. A konténereket át kell rendezni úgy, hogy balról jobbra először az 1-essel, majd a 2-essel, végül a 3-assal jelölt konténerek álljanak. A raktár majdnem tele van, csak az utolsó konténer után van egy konténer számára szabad hely. A rendezést a konténerek fölött mozgatható robottal végezhetjük, amely egy lépésben kiemel a helyéről egy konténert és átteszi azt a szabad helyre, ezzel az átmozgatott konténer helye lesz szabad.

Írj programot, amely kiszámítja, hogy legkevesebb hány lépésben lehet rendezni a konténersort, majd megadja a rendezést! A rendezés végén a szabad helynek a sor végén kell lennie!

A *KONTENER.BE* állomány első sorában a konténerek száma van (1≤N≤10 000). A második sor N egész számot tartalmaz. Az i-edik szám annak a városnak a sorszáma (1 és 3 közötti érték), ahova az i-edik konténert szállítani kell.

A *KONTENER.KI* állomány első sorába a rendezés végrehajtásához minimálisan szükséges lépések M számát kell írni! A következő M sor tartalmazza a sorrendben végrehajtandó mozgatásokat, soronként egy-egy mozgatást! Minden sorban két egész szám legyen; i és j, ami azt jelenti, hogy a lépés során az i-edik helyen lévő konténert a j-edik helyre kell átmozgatni! A kezdetben üres hely sorszáma N+1.

Példa:

KONTENER.BE	KONTENER.KI
12 2 1 2 3 1 3 2 3 2 3 1 2	10 9 13 4 9 1 4 5 1 3 5 11 3 6 11 12 6 8 12 13 8

2. feladat: Bérlet (15 pont)

Egy vállalkozónak van egy nagy értékű munkagépe, amit bérbe ad más vállalkozóknak. Sok megrendelése van a következő 200 napra. Minden megrendelés egy (`e,u`) számpárral adott, ami azt jelenti, hogy az igénylő az `e` sorszámú naptól az `u` sorszámú napig kívánja bérelni a gépet. Mivel a bérleti díj minden napra azonos, ezért a bérbeadónak az a célja, hogy olyan megrendeléseket fogadjon el (és teljesítsen), hogy a lehető legtöbb nap legyen bérbe adva a gépe.

Készíts programot, amely kiszámítja, hogy mely megrendeléseket fogadja el, hogy a gépe a lehető legtöbb napon legyen bérbe adva, és a programod adjon is meg egy beosztást!

A *BERLET.BE* állomány első sorában a megrendelések száma (1≤N≤300) van. A következő `N` sor mindegyike két egész számot tartalmaz (1≤e≤u≤200), ami azt jelenti, hogy a megrendelő az `e` naptól az `u` napig kívánja bérbe venni a gépet.

A *BERLET.KI* állomány első sora a lehető legtöbb nap számát tartalmazza, amelyre a vállalkozó bérbe tudja adni a gépét! A második sorba az elfogadott megrendelések `M` számát kell írni! A harmadik sor `M` egész számot tartalmazzon, azoknak a megrendeléseknek a sorszámait (tetszőleges sorrendben), amelyeket a bérbeadó elvállalt! (Több megoldás esetén bármelyik megadható.)

Példa:

BERLET.BE	BERLET.KI
5 2 7 3 10 13 15 8 11 20 25	19 4 1 4 3 5

3. feladat: Park (15 pont)

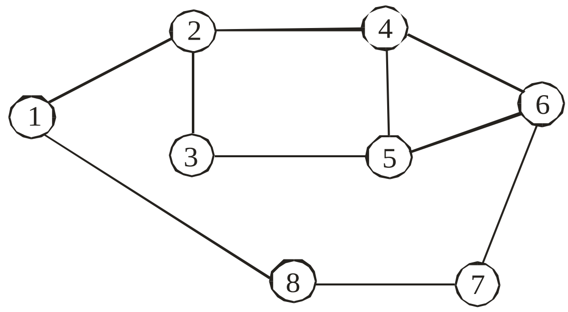
A városi vidámpark több részlegből áll. Az egyes részlegeket kétirányú utak kötik össze. Az úthálózat olyan, hogy bármely részlegtől legfeljebb három közvetlen út vezet más részleghez, kivéve a főbejáratot tartalmazó részleget, onnan legfeljebb két másik részleghez vezet közvetlen út. Egy részleghez érve csak a részlegen keresztül lehet másik útra lépni. Minden részleghez el lehet jutni – esetleg más részlegeken keresztül – a főbejáratot tartalmazó részlegtől. Minden részlegbe csak az oda szóló belépőjeggyel lehet bemenni. Kedvezményesen lehet venni olyan belépőjegy köteget, amely minden részlegbe pontosan három jegyet tartalmaz.

Készíts programot, amely kiszámít egy olyan séta útvonalat, amely a főbejáratot tartalmazó részlegtől indul, oda ér vissza és minden részleget tartalmaz, de minden részleget legfeljebb háromszor!

A PARK.BE állomány első sorában a vidámpark részlegeinek száma (`1≤N≤1000`), és a részlegeket közvetlenül összekötő utak száma (`1≤M≤3000`) van. A részlegeket az `1,…,N` számokkal azonosítjuk, a főbejárat az 1. részlegnél van. A következő `M` sor mindegyike két egész számot tartalmaz (`1≤a,b≤1000, a≠b`), ami azt jelenti, hogy az `a` részleget közvetlen kétirányú út köti össze a `b` részleggel.

A PARK.KI állomány első sorába egy olyan séta útvonalat kell írni, amely a főbejárattal (1) kezdődik és végződik, minden részleget tartalmaz, de legfeljebb háromszor, továbbá az egymást követő részlegek között van közvetlen út! (Több megoldás esetén bármelyik megadható.)

```
Példa:
PARK.BE             PARK.KI
8 10                1 2 4 5 3 5 6 7 8 7 6 5 4 2 1
1 2
2 3
2 4
5 4
3 5
4 6
5 6
6 7
7 8
1 8
```



4. feladat: Malom (15 pont)

A Malomipari Vállalat M malomban őröl és csomagol lisztet. A lisztet N városba kell elszállítani úgy, hogy a szállítási összköltség a lehető legkisebb legyen.

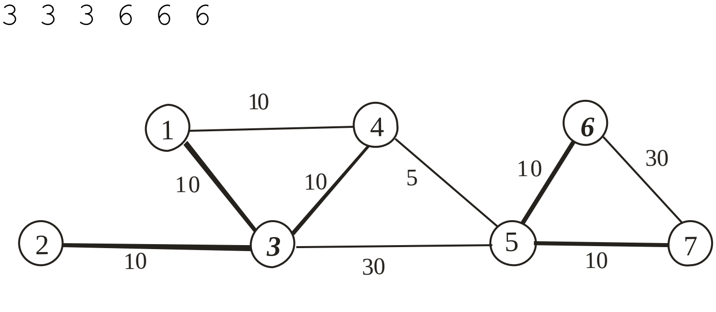
Írj programot, amely megadja a minimális szállítási költséget, illetve minden városra, hogy oda melyik malomból kell szállítani a lisztet!

A *`MALOM.BE`* állomány első sorában a városok száma (`1≤N≤100`), a malmok száma (`1≤M≤N`), valamint a városok közötti utak száma (`1≤U≤10 000`) van. A második sorban M szám, azon városok sorszáma, ahol van malom. A következő U sor mindegyikében két-két olyan város sorszáma, amelyek között van közvetlen út és a két város közötti szállítási költség van. A szállítási költség pozitív egész szám, értéke legfeljebb `200`.

A *`MALOM.KI`* állomány első sorába a minimális szállítási költséget kell írni, a második sorba pedig N malomsorszámot: az i-edik sorszám annak a városnak a sorszáma legyen, amelyikben levő malomból az i-edik városba kell szállítani a lisztet! Amelyik városba sehonnan sem lehet lisztet szállítani, ott a 0 sorszámot kell kiírni!

Példa:

```
MALOM.BE            MALOM.KI
7 2 9               60
3 6                 3 3 3 3 6 6 6
1 4 10
1 3 10
2 3 10
3 4 10
3 5 30
4 5 5
5 6 10
5 7 10
6 7 30
```



5. feladat: Téglalap (15 pont)

Adott a síkon `N` db. pont és egy ezektől különböző `Q` pont. Meg kell határozni egy olyan egyenes állású téglalapot (oldalai párhuzamosak a tengelyekkel), amelyre teljesül az alábbi három feltétel:

A `Q` pont a téglalap belsejében van (nem lehet a határán).

A téglalap mind a négy oldalán pontosan egy-egy pontja van a ponthalmaznak (a négy pont nem feltétlenül különböző).

A ponthalmaz egyetlen pontja sem esik a téglalap belsejébe.

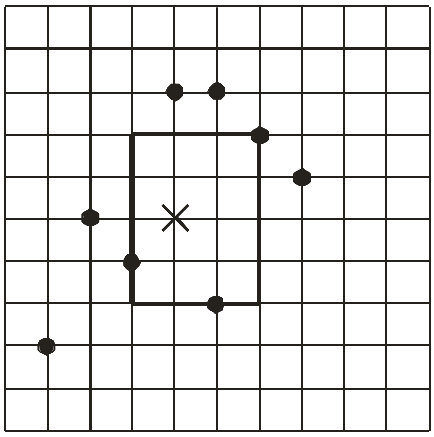
Készíts programot, amely kiszámít egy olyan téglalapot, amely teljesíti a három feltétel mindegyikét, ha van ilyen téglalap!

Az LAP.BE állomány első sorában a `Q` pont koordinátái vannak (`0<A,B≤30 000`). A második sor a ponthalmaz pontjainak számát tartalmazza (`2≤N≤10 000`). A következő `N` sor mindegyike a ponthalmaz egy pontjának x- és y-koordinátáját tartalmazza (`0<X,Y≤30 000`).

Az LAP.KI állomány első és egyetlen sorába négy egész számot kell írni! Az első két szám a feltételt kielégítő téglalap bal-alsó sarkának x- és y-koordinátája, a második két szám pedig a téglalap jobb-felső sarkának x- és y-koordinátája. Ha nincs olyan téglalap, amely kielégíti a feltételt, akkor a 0 0 0 0 számnégyest kell kiírni! (Több megoldás esetén bármelyik megadható.)

Példa:

```
LAP.BE              LAP.KI
4 5                 3 3 6 7
8
1 2
2 5
4 8
5 8
6 7
7 6
5 3
3 4
```



A verseny végeredménye:

I. korcsoport

1.	Gévay Gábor	Táltos Tehetséggondozó Általános Iskola, Szeged
	Hegedűs Tamás	Földes Ferenc Gimnázium, Miskolc
3.	Pálinkás István	5.sz. Általános Iskola, Gyula
	Grósz Dániel	Fazekas Mihály Gimnázium, Budapest
5.	Danner Gábor	Fő Fasori Általános Iskola, Szeged
6.	Radnai Ágnes	Veres Péter Gimnázium, Budapest
7.	Szendrei Péter	Károlyi István 12 évfolyamos Gimnázium, Budapest
8.	Bacsó András	Árvay József Gyakorló Általános Iskola, Sárospatak
9.	Varga László	Fazekas Mihály Gimnázium, Debrecen
10.	Lőrincz Máté	Péterfy Sándor Evangélikus Iskolaközpont, Győr
	Szendrei Balázs	Károlyi István 12 évfolyamos Gimnázium, Budapest

II. korcsoport

1. Vincze János	Fazekas Mihály Gimnázium, Debrecen
2. Nagy Gergely	Fazekas Mihály Gimnázium, Budapest
3. Eisenberger András	Fazekas Mihály Gimnázium, Budapest
4. Peregi Tamás	Berzsenyi Dániel Gimnázium, Budapest
5. Korándi Dániel	Fazekas Mihály Gimnázium, Budapest
Vámosi Péter	Berzsenyi Dániel Evangélikus Gimnázium, Sopron
7. Nagy Csaba	Fazekas Mihály Gimnázium, Budapest
Haszpra Zsolt	Árpád Gimnázium, Budapest
9. Badics Alex	Kőkúti Általános Iskola, Tata
10.Kunovszki Péter	Kisfaludy Károly Gimnázium, Mohács

III. korcsoport

1. Torma Róbert	Berzsenyi Dániel Gimnázium, Budapest
2. Stippinger Marcell	Széchenyi István Gimnázium, Sopron
3. Kormányos Balázs	Radnóti Miklós Gimnázium, Szeged
4. Strenner Balázs	Teleki Blanka Gimnázium, Székesfehérvár
5. Ludányi Ákos	Neumann János Középiskola és Kollégium, Eger
Megyesi Péter	Fazekas Mihály Gimnázium, Budapest
Nagy Szabolcs	Garay János Gimnázium, Szekszárd
Kótyuk Gergely	Krúdy Gyula Gimnázium, Nyíregyháza
9. Papp Gábor	Neumann János Középiskola és Kollégium, Eger
10.Tassy Gergely	Veres Péter Gimnázium, Budapest
Szabó Gábor	Neumann János Középiskola és Kollégium, Eger
12.Suba Gergely	Szabad Waldorf Óvoda, Általános Iskola és Gimnázium, Fót
Nikházy László	Kazinczy Ferenc Gimnázium és Szakközépiskola, Győr
14.Cséri Tamás	Zrínyi Miklós Gimnázium, Zalaegerszeg
15.Molnár Dömötör	Fazekas Mihály Gimnázium, Budapest
16.Leskó Dániel	Árpád Vezér Gimnázium, Sárospatak

2006. Első forduló

Ötödik-nyolcadik osztályosok

<u>1. feladat</u>: Képlet (12 pont)

Az alábbi képlettel egy legfeljebb kétjegyű X számot alakítunk át.

```
X:=((X*X) div 10) mod 100
```

A. Mit számol ki az algoritmus X=12, X=42, X=63?

B. Adj meg 3 olyan X értéket, amelynél a képlet kiszámolása után X értéke nem változik!

<u>2. feladat</u>: Óra (20 pont)

Az alábbi algoritmusrészletben az A(6) vektor az óra aktuális állását tartalmazza (pl. A(1)=6, A(2)=5, A(3)=4, A(4)=3, A(5)=2, A(6)=1 esetén a pontos idő 12 óra 34 perc 56 másodperc). Az algoritmus az így tárolt időt növeli meg 1 másodperccel, azonban nem mindig működik helyesen. Milyen esetekben hibás? Fogalmazd meg szöveggel és adj példát!

```
i:=0
Ciklus
  i:=i+1; A(i):=A(i)+1
  Ha i=1 vagy i=3 vagy i=5 akkor x:=9 különben x:=6
  Ha A(i)≥x akkor A(i):=0
amíg A(i)=0
Ciklus vége
```

<u>3. feladat</u>: Átrendezés (24 pont)

Az alábbi algoritmusrészletek az M1, M2 és M3 változók értékeit rendezik át valamilyen sorrendbe.

A. Mi lesz M1, M2, M3 értéke az algoritmus végrehajtása után, ha előtte M1=3, M2=1, M3=5?

B. Melyik esetben milyen lesz a változók sorrendje az algoritmus végrehajtása után?

```
1. Ha M1<M2 vagy M1<M3
      akkor Ha M2>M1 és M2>M3 akkor s:=M1; M1:=M2; M2:=s
                             különben s:=M1; M1:=M3; M3:=s
   Ha M2<M3 akkor s:=M2; M2:=M3; M3:=s

2. Ha M1>M2 akkor s:=M1; M1:=M2; M2:=s
   Ha M2<M3 akkor s:=M2; M2:=M3; M3:=s
   Ha M1<M3 akkor s:=M1; M1:=M3; M3:=s

3. Ha M1<M2 akkor s:=M1; M1:=M2; M2:=s
   Ha M1>M3 akkor s:=M1; M1:=M3; M3:=s
   Ha M1>M2 akkor s:=M1; M1:=M2; M2:=s
```

<u>4. feladat</u>: Vonalkód (18 pont)

Egy vonalkód-olvasó berendezés speciális vonalkódokat használ. Minden árut 1000 és 6999 közötti négyjegyű számmal azonosítanak. A vonalkódban minden számjegyet négyjegyű bináris számmal írnak le, a bináris 0-kat vékony, az 1-eseket pedig vastag vonallal jelölik. Ismerjük két árucikk azonosító számát és vonalkódját:

1920=▮ | | | ▮ | | ▮ | ▮ | | | | | |, illetve 6375=▮ ▮ | ▮ ▮ ▮ | | ▮ ▮ ▮ ▮ ▮ | ▮ |.

Tudjuk továbbá, hogy az áruk vonalkódja olyan, hogy mindkét irányból olvasva is felismerhető, azaz az 1-től 6-ig levő számok kódját visszafelé nem használhatjuk.

A. Add meg a két árucikk kódja alapján a 0,1,2,3,5,6,7,9 számjegyek vonalkódját!

B. Add meg, hogy milyen kódja lehet a 4-esnek és a 8-asnak, és döntsd el, hogy ezek közül a „két irányból olvasva is felismerhető” feltétel szerint melyik a 4-es kódja!

5. feladat: Szöveg átalakító (26 pont)

Az alábbi algoritmusrészlet az N (N≥1) karaktert tartalmazó S tömböt alakítja át, csillag karakter biztos nincs benne. A `kar(D)` függvény előállítja a D kódú karaktert.

```
j:=1; D:=1
Ciklus i=2-től N-ig
  Ha S(i)=S(j) akkor D:=D+1; S(j+D-1):=S(i)
  különben Ha D≤3 akkor j:=j+D                        {*}
           különben S(j):='*'; S(j+1):=kar(D); j:=j+3 {**}
           D:=1; S(j):=S(i)
  Elágazás vége
Ciklus vége
```

A. Mi lesz a futás végén az S tömb tartalma, ha kezdetben S az 'ABCAAAABBBCCCCCA' szöveget tartalmazza?

B. Milyen esetben hajtódik végre a {*}-gal, illetve a {**}-gal jelölt elágazás-ág?

C. Fogalmazd meg szavakkal, hogy mi az átalakítás elve!

D. Milyen esetben működik hibásan az algoritmus?

Kilencedik-tizedik osztályosok

1. feladat: Áramkör (20 pont)

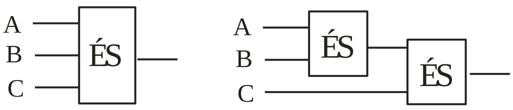
Elektronikus áramköröket építhetünk fel ÉS-, VAGY-, valamint NEM-kapukból. Ezek működését egy-egy táblázattal adhatjuk meg:

A	B	A és B
0	0	0
0	1	0
1	0	0
1	1	1

A	B	A vagy B
0	0	0
0	1	1
1	0	1
1	1	1

A	nem A
0	1
1	0

Elvileg három bemenetes ÉS-kaput is készíthetnénk, de azt megvalósíthatjuk kettő két bemenetes ÉS-kapuval:



, azaz ÉS(A,B,C)=(A és B) és C.

Az alábbi táblázatok három bemenetes áramkörök működését írják le. Add meg, hogy melyik hogyan valósítható meg a lehető legkevesebb két bemenetes ÉS-, illetve VAGY-kapukból, valamint egy bemenetes NEM-kapukból!

A	B	C	?1
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

A	B	C	?2
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

A	B	C	?3
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

A	B	C	?4
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

2. feladat: Kupac (20 pont)

A kupac adatstruktúra egy olyan N elemű tömbként képzelhető el, ahol a tömb minden i eleméről tudjuk, hogy T(i)≤T(2*i) és T(i)≤T(2*i+1), ha 2*i≤N, illetve 2*i+1≤N. Az alábbi két eljárást írtuk hozzá:

```
Betesz(T,N,X):
  i:=N+1
  Ciklus amíg i>1 és T(i div 2)>X
    T(i):=T(i div 2); i:=i div 2
  Ciklus vége
  T(i):=X; N:=N+1
Eljárás vége.
```

```
Kivesz(T,N,X):
  X:=T(1); Y:=T(N); i:=1; N:=N-1
  Ciklus amíg i≤N div 2 és (Y>T(2*i) vagy Y>T(2*i+1))
    Ha 2*i+1>N vagy T(2*i)≤T(2*i+1)
        akkor T(i):=T(2*i); i:=2*i
     különben T(i):=T(2*i+1); i:=2*i+1
  Ciklus vége
  T(i):=Y
Eljárás vége.
```

Kezdetben a tömb tartalma legyen: N=10, T=(3,8,5,11,9,6,20,13,12,10)!

A. Mi lesz a T tömb tartalma a Betesz(T,N,7), a Betesz(T,N,8) és a Betesz(T,N,4) műveletek végrehajtása után? Mindegyik után add meg a T tömböt, a másodikat az első eljáráshívás eredményére, a harmadikat a második eljáráshívás eredményére alkalmazd!

B. Mi lesz a T tömb tartalma és az X változó a Kivesz(T,N,X) egyszeri, illetve kétszeri alkalmazása után? Az eredeti T tömbre add meg a választ!

3. feladat: Rágógumi (22 pont)

Egy dobozban piros, fehér és zöld rágógumikat tárolunk. Véletlenszerűen kiveszünk 2 darabot. Ha egyformák, akkor egy fehéret visszateszünk (feltesszük, hogy van nálunk elég fehér). Ha különbözők, akkor csak a fehéret tarthatjuk meg, a másfélét vissza kell tenni. Ha egynél több rágógumi maradt, akkor a maradékra fenti algoritmus újra kezdődik.

A: Hogyan változik a piros, a fehér, illetve a zöld rágógumik száma az algoritmus végrehajtása során?

B. A rágógumik számának milyen lényeges tulajdonsága nem változik meg az algoritmus végrehajtása során? Hogyan változik a rágógumik száma?

C. Milyen kiinduló állapot esetén kerülhetünk végtelen ciklusba az algoritmus végrehajtása során?

D. Mitől függ, hogy a végén hány rágógumi marad és milyen színű?

4. feladat: Fazekas (18 pont)

Egy fazekas műhelyében sorban várakoznak a kiégetésre váró tárgyak. Minden tárgyról tudjuk, hogy mennyi az a legkevesebb idő, ami a kiégetéséhez kell. Az égetésre váró tárgyakat az érkezésük sorrendjében kell kiégetni. Egyszerre több tárgyat is rakhatunk a kemencébe, azonban legfeljebb annyit, amennyi a kemence K kapacitása. Az égetési idő egy menetben mindig a kemencébe rakott tárgyak minimális égetési idejének a maximuma kell legyen.

Jelöljük `Opt(i)`-vel az első `i` tárgy kiégetéséhez szükséges minimális időt!

A. Add meg, hogy 10,8,20,25,30,12,40 égetési idők és K=3 esetén hogyan néz ki az `Opt` vektor!

B. Adj képletet `Opt(i)` kiszámítására tetszőleges K esetén, `Idő(i)`-vel jelöld az i-edik tárgy kiégetéséhez szükséges időt!

5. feladat: Logika (20 pont)

Egy logikai programozási nyelven alapismereteket és következtetési szabályokat adhatunk meg. Alapismeret lehet például:

```
apja("Nagy János","Nagy Péter").
anyja("Fekete Éva","Nagy Péter").
```

Szabályok például:

```
szülője(X,Y) ha apja(X,Y) vagy anyja(X,Y).
nagyszülője(X,Y) ha szülője(X,Z) és szülője(Z,Y).
őse(X,Y) ha szülője(X,Y) vagy szülője(X,Z) és őse(Z,Y).
```

Az utolsó szavakkal megfogalmazva: akkor őse az X az Y-nak ha szülője, vagy pedig akkor, ha van olyan Z, akinek X a szülője és a Z őse az Y-nak.

A. Milyen rokonsági kapcsolatot határoznak meg az alábbi szabályok:

```
A1. rokon1(X,Y) ha szülője(Z,Y) és anyja(X,Z).
A2. rokon2(X,Y) ha anyja(X,Y) vagy apja(Z,Y) és rokon2(X,Z).
```

B. Írd meg a következő rokoni kapcsolatokat leíró szabályokat:

B1. Az X anyai dédanyja az Y-nak.

B2. Az X az Y-nak férfi őse.

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Kapuk (16 pont)

Elektronikus áramköröket építhetünk fel kizárólag a NEM-VAGY (NOR) kapu használatával. Azaz a többi kapu (pl. VAGY, ÉS, KIZÁRÓVAGY, EKVIVALENCIA, NEM) ezzel az eggyel megvalósítható. Például NOT(A)=NOR(A,A). Működésüket egy-egy táblázattal adhatjuk meg:

A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

A	NOT A
0	1
1	0

A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

A	B	A EQU B
0	0	1
0	1	0
1	0	0
1	1	1

A	B	A NOR B
0	0	1
0	1	0
1	0	0
1	1	0

Add meg, hogy az AND, OR, XOR, EQU kapukat hogyan lehet megvalósítani a lehető legkevesebb NOR kapuval!

2. feladat: Kupac (20 pont)

A kupac adatstruktúra egy olyan N elemű tömbként képzelhető el, ahol a tömb minden i eleméről tudjuk, hogy T(i)≤T(2*i) és T(i)≤T(2*i+1), ha 2*i≤N, illetve 2*i+1≤N. Az alábbi eljárást írtuk hozzá: (feltehető, hogy T(N+1) értéke biztos kisebb X-nél)

```
Módosít(T,N,i,X):
  Ha T(i)>X akkor
    Ciklus amíg i>1 és T(i div 2)>X
      T(i):=T(i div 2); i:=i div 2
    Ciklus vége
  különben ha T(i)<X akkor
    Ciklus amíg i≤N div 2 és (X>T(2*i) vagy X>T(2*i+1))
      Ha 2*i+1>N vagy T(2*i)≤T(2*i+1)
          akkor T(i):=T(2*i); i:=2*i
       különben T(i):=T(2*i+1); i:=2*i+1
    Ciklus vége
  Elágazás vége
  T(i):=X
Eljárás vége.
```

Kezdetben a tömb tartalma legyen: N=10, T=(3,8,5,11,9,6,20,13,12,10)!

A. Mi lesz a T tömb tartalma a Módosít(T,N,8,7), a Módosít(T,N,7,10) és a Módosít(T,N,10, 2) műveletek végrehajtása után? Mindegyik után add meg a T tömböt, a másodikat az első eljáráshívás eredményére, a harmadikat a második eljáráshívás eredményére alkalmazd!

B. Mi lesz a T tömb tartalma a Módosít(T,N,1,7), majd utána a Módosít(T,N,2,11) műveletek végrehajtása után? Az eredeti T tömbre add meg a választ!

<u>3. feladat</u>: Bors (24 pont)

Egy dobozban fekete és fehér borsszemeket tárolunk. Véletlenszerűen kiveszünk 3 szemet. Ha mindhárom fekete, akkor nem teszünk semmit. Ha két fekete van köztük, akkor mind a hármat visszarakjuk. Ha két fehér van köztük, akkor a feketét visszarakjuk. Ha mindhárom fehér, akkor pedig egy fehéret rakunk vissza. Ha kettőnél több bors maradt, akkor a maradékra fenti algoritmus újra kezdődik.

A: Hogyan változik a fekete, illetve a fehér borsszemek száma az algoritmus végrehajtása során?

B. A borsszemek számának milyen lényeges tulajdonsága nem változik meg az algoritmus végrehajtása során?

C. Milyen kiinduló állapot esetén kerülhetünk végtelen ciklusba az algoritmus végrehajtása során?

D. Mitől függ, hogy a végén hány bors marad és azok milyen színűek?

<u>4. feladat</u>: Tükörszó (18 pont)

Egy karaktersorozat *tükörszó*, vagy *palindrom*, ha szimmetrikus, azaz ha balról jobbra és jobbról balra olvasva azonos.

Például 2 karakter beszúrásával az S=„Ab3bd” karaktersorozat palindrommá alakítható („dAb3bAd” vagy „Adb3bdA” is lehet belőle). Kettőnél kevesebb karakter beszúrásával azonban ebből a karaktersorozatból nem állítható elő palindrom.

Jelöljük `M(i,j)`-vel minden `(i,j) i≤j` indexpárra, hogy az `S[i..j]=S[i]...S[j]` szó legkevesebb hány betű-beszúrással tehető tükörszóvá!

A. Add meg, hogy `S=`”FAKANÁL” esetén hogyan néz ki az `M` mátrix!

B. Adj képletet `M(i,j)` kiszámítására!

<u>5. feladat</u>: Logika (22 pont)

Egy logikai programozási nyelven alapismereteket és következtetési szabályokat adhatunk meg.

Alapismeret lehet például:

```
apja(”Nagy János”,”Nagy Péter”).
anyja(”Fekete Éva”,”Nagy Péter”).
```

Szabályok például:

```
szülője(X,Y) ha apja(X,Y) vagy anyja(X,Y).
nagyszülője(X,Y) ha szülője(X,Z) és szülője(Z,Y).
őse(X,Y) ha szülője(X,Y) vagy szülője(X,Z) és őse(Z,Y).
```

Az utolsó szavakkal megfogalmazva: akkor őse X Y-nak, ha szülője, vagy pedig akkor, ha van olyan Z, akinek X a szülője és a Z őse az Y-nak.

A. Milyen rokonsági kapcsolatot határoznak meg az alábbi szabályok:

```
A1. rokon1(X,Y) ha szülője(Z,Y) és apja(X,Z).
```

```
A2. rokon2(X,Y) ha apja(X,Y) vagy anyja(Z,Y) és rokon2(X,Z).
```

B. Írd meg a következő rokoni kapcsolatokat leíró szabályokat:

B1. Az X anyai dédapja az Y-nak

B2. Az X olyan férfiutódja Y-nak, akinek van gyereke

2006. Második forduló

Ötödik-nyolcadik osztályosok

1. feladat: Mozi (27 pont)

Iskolád közönségszervezőjeként Te tartod a kapcsolatot egy mozival. A mozi az alábbi kedvezményeket biztosítja, egyszerre közülük csak az egyik vehető igénybe:

Csoportos kedvezmény:

A csoport létszáma	A kedvezmény mértéke
10 fő alatt	0%
10-19 fő	5%
20-29 fő	8%
30-40 fő	12%
40 fő fölött	14%

Iskolai kedvezmény:

A csoport létszáma	Ebből ingyenjegyet kap
5 fő alatt	0 fő
5-11 fő	1 fő
12-19 fő	2 fő
20-28 fő	3 fő
29-40 fő	4 fő
40 fő fölött	5 fő

Diákkedvezmény: egyénileg is jár, mértéke 10%.

Készíts programot, amely beolvassa a moziba menők számát (`1≤N≤100`), majd megadja, hogy a háromféle kedvezményből melyiket kell igénybe venni, hogy a lehető legkevesebbe kerüljenek a jegyek!

Megjegyzés: a jegyek árát nem ismerjük, de a százalékok alapján a kérdés megválaszolható.

Példa:

```
Bemenet: 3      Kimenet: DIÁKKEDVEZMÉNY
Bemenet: 5      Kimenet: ISKOLAI KEDVEZMÉNY
```

2. feladat: Utazás (28 pont)

A Piripócs-Kukutyin vasútvonalon pontosan N állomás van ($2 \leq N \leq 100$), beleértve a két végállomást is. A kalauz egy út során minden állomásról feljegyezte, hogy a vonat mikor ért oda, illetve mikor hagyta el az állomást. Ez a két szám egyenlő, ha a vonat az állomáson nem állt meg.

Készíts programot, amely beolvassa (időrendben) az indulási és érkezési időpontokat, majd megadja

A. azt az állomást, ahol a vonat a legrövidebb ideig állt; valamint

B. azt a két legmesszebb levő állomást, amelyek között a vonat egy állomáson sem állt meg!

Példa:

Bemenet: N=7

	1.	2.	3.	4.	5.	6.	7
Érkezik		8:50	9:20	9:40	9:55	10:15	10:40
Indul	8:40	8:50	9:25	9:40	9:55	10:25	

```
Legrövidebb ideig állt: 3. állomás
Legmesszebb levő állomások: 3-6. állomás
```

Lehetséges eredmények még:

```
Nem állt meg sehol
Mindenhol megállt
Nincs közbülső állomás
```

3. feladat: Hasáb (20 pont)

Készíts programot, amely beolvas N keresztnevet ($1 \leq N \leq 30$), a hasábok számát ($1 \leq K \leq 3$), majd a keresztneveket kiírja a képernyőre K hasábban! A hasáb szélessége legyen a benne levő leghosszabb keresztnév hossza (feltehető, hogy az összes név elfér a képernyőn és három hasáb is kifér egy sorba)! A hasábokat egymástól egyetlen szóköz válassza el! A hasábokon belül a neveket jobbra kell igazítani!

Példa: (a példában függőleges vonalak jelölik a képernyő bal szélét és a harmadik hasáb jobb szélét)

```
Bemenet:              Kimenet
N=8, K=3              |  Szilvia    Éva Eszter|
Szilvia               |   Mónika Csilla  Anikó|
Mónika                |Krisztina   Nóra       |
Krisztina
Éva
Csilla
Nóra
Eszter
Anikó
```

Kilencedik-tizedik osztályosok

1. feladat: Átló (10 pont)

Egy N oldalú szabályos sokszögbe átlókat húzunk egymás után. Az átlókat a két összekötött csúcs sorszámával adjuk meg, a sorszámok 1 és N közötti egész számok.

Írj programot, amely megadja, hogy hányadik átlónál fordul elő először, hogy az átló korábbi átlókat metsz valahol!

Az *ATLO.BE* állomány első sorában a csúcsok száma (`1≤N≤100`) és a behúzandó átlók száma (`1≤M≤1000`) van. A következő M sor mindegyikében egy-egy átló két végpontjának sorszáma van.

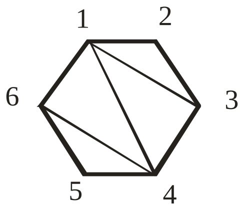
Az *ATLO.KI* állomány egyetlen sorába annak az átlónak a sorszámát kell írni, amely metsz valamely korábban behúzott átlót! Ha egyetlen átló sem metszi egyik korábbit sem, akkor 0-t kell kiírni!

<u>Példa:</u>

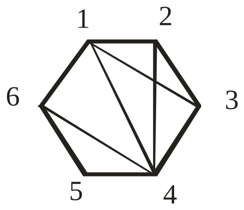
```
ATLO.BE
6 5
1 3
4 6
4 1
2 4
2 5
```

```
ATLO.KI
4
```

a 3. átló után:



a 4. átló után:



<u>2. feladat</u>: Csőposta (15 pont)

Egy vállalat két épülete (A és B) között csőpostát üzemeltet, amiben mindkét irányban haladhatnak csomagok, ha persze nem jön szembe egy másik csomag. A csomagok a csőben különböző sebességgel haladhatnak.

Ha egy gyorsabb csomag ugyanabban az irányban haladva utolér egy lassabb csomagot, akkor összeütköznek és gyorsabb lelassul a lassabb sebességére. Ha szemben találkoznak, akkor útjukat a gyorsabb irányába, a gyorsabb sebességével folytatják. Egyforma sebesség esetén a később induló irányába haladnak tovább.

Készíts programot, amely megadja, hogy hány csomag ütközik össze valamelyik előzővel és melyek ezek!

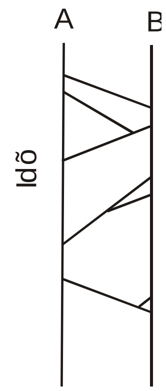
A *CSOPOSTA.BE* állomány első sorában a csomagok száma van (`1≤N≤1000`). A következő `N` sor mindegyikében egy-egy csomag leírása található indulási idő szerint növekvő sorrendben. Minden sor első karaktere a csomag induló helye (A vagy a B betű), ettől egy szóközzel elválasztva következik az indulás ideje és az út megtételéhez szükséges idő. (A példában az első csomag a 10. percben indul A-ból és 10 perc múlva érkezik meg B-be.)

A *CSOPOSTA.KI* állomány első sorába az ütközések `K` számát kell írni, a következő `K` sorba pedig azon csomagok sorszámát, amelyek valamelyik korábbival ütköznek!

<u>Példa</u>:

```
CSOPOSTA.BE
7
A 10 10
A 15 15
B 25 10
B 40 20
B 45 10
A 70 10
B 75 20
```

```
CSOPOSTA.KI
3
3
5
7
```

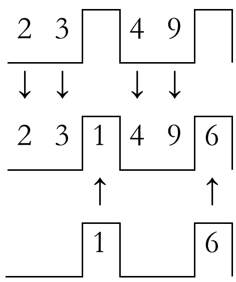


<u>3. feladat</u>: Sorozat (10 pont)

Adott egy egész számokból álló sorozat.

Készíts programot, amely kiszámít két olyan monoton növekvő sorozatot, amelyekből fésűs egyesítéssel megkapható a bemeneti sorozat!

A *SOROZAT.BE* állomány első sorában a bemeneti sorozat elemeinek száma (`0≤N≤1000`) van. A második sor pontosan `N` egész számot tartalmaz, a bemeneti sorozatot. A sorozat minden elemére teljesül, hogy `0≤x≤30 000`.



A *SOROZAT.KI* állomány első és egyetlen sora a `0 0` számpárt tartalmazza, ha a bemeneti sorozat nem állítható elő két monoton növekvő sorozat fésűs egyesítéseként, egyébként az első sorban a fésűs egyesítéshez kiszámított első sorozat `K` elemszáma álljon! A második sor pontosan `K` számot tartalmazzon, az első sorozat elemeit! A harmadik sor tartalmazza a második sorozat `L` elemszámát! A negyedik sor pontosan `L` számot tartalmazzon, a második sorozat elemeit! A kiszámított sorozatok egyike üres is lehet, ekkor a `0` számot és üres sort kell kiírni! A bemeneti sorozat minden eleme pontosan az egyik kiszámított sorozat eleme. Több megoldás esetén bármelyik megadható.

<u>Példa:</u>

SOROZAT.BE	SOROZAT.KI
6	4
2 3 1 4 9 6	2 3 4 9
	2
	1 6

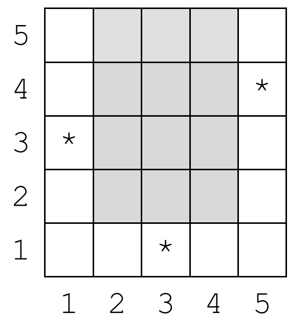
<u>4. feladat</u>: Térkép (20 pont)

Egy síkbeli csillagtérképen `N` csillag található.

Készíts programot, amely megadja az `MxM`-es térképen a legnagyobb területű téglalapot, amelyben nincs egyetlen csillag sem és a téglalap oldalai párhuzamosak a térkép széleivel!

A *TERKEP.BE* állomány első sorában a csillagok száma (`0≤N≤1000`) és a térkép mérete (`1≤M≤100`) van. A következő `N` sor mindegyike egy-egy csillag x-, illetve y-koordinátáját (`1≤x,y≤M`) tartalmazza.

A *TERKEP.KI* állomány első sorába a legnagyobb csillagmentes téglalap méretét kell írni, a második sorba pedig a téglalap bal alsó, valamint jobb felső sarkának x- és y-koordinátáját!



<u>Példa:</u>

TERKEP.BE	TERKEP.KI
3 5	12
1 3	2 2 4 5
3 1	
5 4	

<u>5. feladat</u>: Randevú (20 pont)

Ádám és Éva szeretne találkozni. Éva az `E` városban, Ádám pedig az `A` városban van és. az `R` városban akarnak találkozni. Vonattal kívánnak utazni, és ismerik a teljes menetrendet. A menetrend `N` várost tartalmaz, és azt, hogy mely városok között van vonatjárat. Minden vonat adott i-edik városból indul és adott j-edik városba közlekedik és közben nem áll meg egyetlen közbülső állomáson sem. Mindketten olyan útvonalon akarnak utazni, hogy a lehető legkevesebbszer kelljen átszállni.

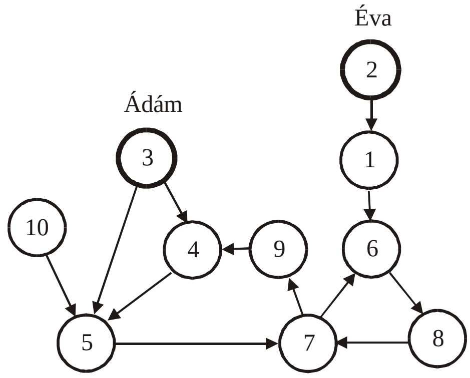
Készíts programot, amely meghatároz Ádám és Éva számára egy-egy legkevesebb átszállásos útvonalat!

A *RANDI.BE* állomány első sora a városok számát (`1≤N≤200`), Éva tartózkodási helyét, Ádám tartózkodási helyét (`1≤E≠A≤N`), az R találkahelyet (`1≤R≤N, R≠A, R≠E`) és a járatok számát (`1≤M≤3000`) tartalmazza. A további `M` sor mindegyikében a járat `i` indulási és a járat `j` érkezési állomása van (`1≤i,j≤N`). Bármely `i` és `j` városra legfeljebb egy járat van `i`-ből `j`-be.

A *RANDI.KI* állomány első sorába a `0 0` számpárt kell írni, ha akár Ádám, akár Éva nem tud eljutni a találkahelyre, egyébként az első sor olyan `K M` számpárt tartalmazzon, hogy Éva `K` város, Ádám pedig `M` város érintésével tud eljutni a találkahelyre! Ekkor a második sor Éva útvonalát, a harmadik pedig Ádám útvonalát tartalmazza! Az útvonalakba beleszámít Éva és Ádám kiindulási tartózkodási helye is.

<u>Példa:</u>

RANDI.BE	RANDI.KI
10 2 3 7 12	5 3
2 1	2 1 6 8 7
1 6	3 5 7
7 6	
6 8	
8 7	
7 9	
9 4	
5 7	
10 5	
3 5	
3 4	
4 5	



Tizenegyedik-tizenharmadik osztályosok

<u>1. feladat</u>: Kiállítás (15 pont)

Egy kiállítás három napon keresztül folyamatosan nyitva tart éjjel-nappal. A látogatóknak előre meg kellett venniük a jegyet, mégpedig úgy, hogy meg kellett mondaniuk, hogy mikor érkeznek, és mikor távoznak a kiállításról. A kiállítás szervezői így pontosan tudják, hogy mikor nem lesz senki a kiállításon. Azt tervezik, hogy csak azokra az időszakokra biztosítanak személyzetet, amikor lesz látogató.

Készíts programot, amely kiszámítja azokat az időintervallumokat, amikor személyzetet kell biztosítani!

A *KIALLIT.BE* állomány első sorában a jegyet váltott látogatók száma ($1 \le N \le 20\,000$) van. A következő `N` sor mindegyikében az érkezés ideje és a távozás ideje ($1 \le A < B \le 4320$) van.

A *KIALLIT.KI* állomány első sorába azoknak az időintervallumoknak az `M` számát kell írni, amelyekre a szervezőknek személyzetet kell biztosítani! A további `M` sor mindegyike egy-egy ilyen időintervallumot adjon meg, az intervallum kezdetét és végét! Az intervallumoknak nem lehet közös pontja, és az összhosszuk a lehető legkisebb legyen! Ha egy látogató az `X` időpontban érkezik és az `Y` időpontban távozik, akkor a személyzetnek jelen kell lennie mind az `X`, mind az `Y` időpontban! Az intervallumokat kezdőpontjuk szerint növekvő sorrendben kell kiírni!

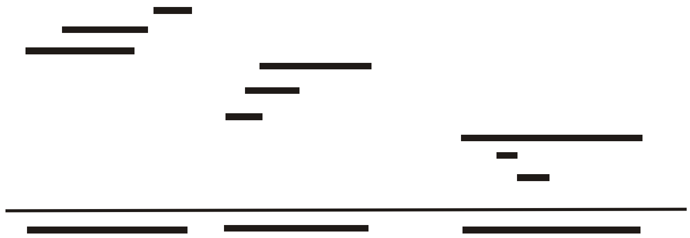
Példa:

KIALLIT.BE

```
9
8 10
3 8
1 7
14 20
13 16
12 14
25 35
27 28
28 30
```

KIALLIT.KI

```
3
1 10
12 20
25 35
```



2. feladat: Futár (15 pont)

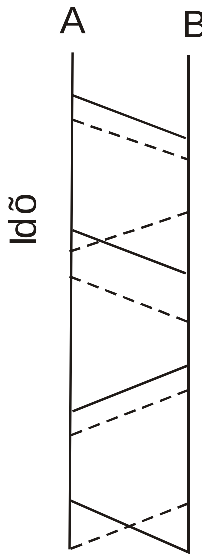
Egy vállalat két telephelye (A és B) között csomagok kézbesítésére két futárt alkalmaz. A futárok a távolságot mindig O perc alatt teszik meg. Ha éppen szemben haladnak egymással, akkor találkozhatnak.

Készíts programot, amely megadja, hogy a futárok hányszor, illetve hányadik kézbesítésükkor találkozhatnak egymással út közben!

A *FUTAR.BE* állomány első sorában az első és a második futár kézbesítéseinek száma (1≤N,M≤1000), továbbá a távolság megtételéhez szükséges idő (1≤O≤100) van. A következő N sorban az első, az azt követő M sorban pedig a második futár kézbesítéseit írtuk le, mindegyiket indulási idő szerint növekvő sorrendben. Minden kézbesítéshez tartozó sor egy betűvel (A vagy B) kezdődik – annak a telephelynek az azonosítójával, ahonnan a futárnak el kell indulnia. Ezt követi egy szóközzel elválasztva az indulás ideje (0≤idő≤20 000). (Feltesszük, hogy a futár az indulás idejében a megfelelő telephelyen van.)

A *FUTAR.KI* állomány első sorába a találkozások K számát kell írni! A következő K sor mindegyikében két szám legyen: az első és a második futár kézbesítésének sorszáma, ami alatt találkozhatnak! Ezek a sorok a találkozási idő szerint növekvő sorrendben legyenek!

Példa: (az ábrán a második futár útját szaggatott vonallal jelöltük)



FUTAR.BE

```
4 5 10
A 10
A 40
B 70
A 100
A 15
B 35
A 50
B 75
B 100
```

FUTAR.KI

```
2
2 2
4 5
```

3. feladat: Konferencia (15 pont)

Egy konferencián párhuzamosan K szekcióban tarthatnak előadást. Minden szekció L egymás utáni előadásból áll. Az előadásokat témák szerint csoportosították, összesen N darab témát neveztek meg. Teljesül, hogy K*L=az előadások száma.

Készíts programot, amely úgy osztja a témákat szekciókba, hogy minden téma előadásai azonos szekcióban, egymás után legyenek! Tudjuk, hogy biztosan létezik megoldás.

A *KONF.BE* állomány első sorában a szekciók száma (1≤K≤5), az egymás utáni blokkok száma (1≤L≤10), valamint a témák száma (1≤N≤50) van. A következő N sorban az egyes témák neve (pontosan 5 karakteren), s tőle egy szóközzel elválasztva a témához tartozó előadások száma van.

A *KONF.KI* állományba pontosan K sort kell írni, az i-edik sorba az i-edik szekció témanevei kerüljenek a beosztás sorrendjében, azaz pontosan L darab 5 karakteres név, egy-egy szóközzel elválasztva!

<u>Példa:</u>

KONF.BE

```
2 3 4
AAAAA 1
BBBBB 2
CCCCC 2
DDDDD 1
```

KONF.KI

```
BBBBB BBBBB AAAAA
CCCCC CCCCC DDDDD
```

<u>4. feladat</u>: Terület (15 pont)

Egy négyzet alakú területre egy befestett sokszöget rajzoltunk, amelynek oldalai párhuzamosak a képernyő szélével.

Készíts programot, amely megadja, hogy hány képpontból áll a sokszög!

A *TERULET.BE* állomány első sorában a sokszög sarokpontjainak száma (4≤N≤1000) és a terület mérete (1≤M≤200) van. A következő N sor mindegyikében egy-egy pont x- és y-koordinátája van (1≤x,y≤M). A legelső pont a legkisebb x-koordinátájú pont (azok közül is a legkisebb y-koordinátájú), s innen kezdve az óramutató járása szerint soroltuk fel a pontokat.

A *TERULET.KI* állomány egyetlen sorába azon pontok számát kell írni, amelyek a sokszöghöz tartoznak!

<u>Példa</u>: (az ábrán X jelöli a sarokpontokat, O a határvonalakat, s szürkére festettük a sokszöghöz tartozó összes pontot)

TERULET.BE

```
10 9
1 1
1 8
4 8
4 7
3 7
3 3
5 3
5 7
7 7
7 1
```

TERULET.KI

```
50
```

9									
8	X	O	O	X					
7	O		X	X	X	O	X		
6	O		O		O		O		
5	O		O		O		O		
4	O		O		O		O		
3	O		X	O	X		O		
2	O						O		
1	X	O	O	O	O	O	X		
	1	2	3	4	5	6	7	8	9

<u>5. feladat</u>: Találka (15 pont)

Ádám és Éva szeretne találkozni. Éva az E városban, Ádám pedig az A városban van. Vonattal kívánnak utazni, és ismerik a teljes menetrendet. A menetrend N várost tartalmaz, és azt, hogy mely városok között van vonatjárat. Minden vonat adott i-edik városból indul és adott j-edik városba közlekedik és közben nem áll meg egyetlen közbülső állomáson sem.

Elhatározták, hogy ha Ádám el tud jutni vonattal Éva városába, akkor csak Ádám fog utazni. Ha Ádám nem tud odautazni Évához, de Éva el tud utazni Ádám városába, akkor csak Éva fog utazni. Ha egyikük sem tud odamenni a másik városába, akkor keresnek egy olyan harmadik várost, ahova mindketten el tudnak utazni.

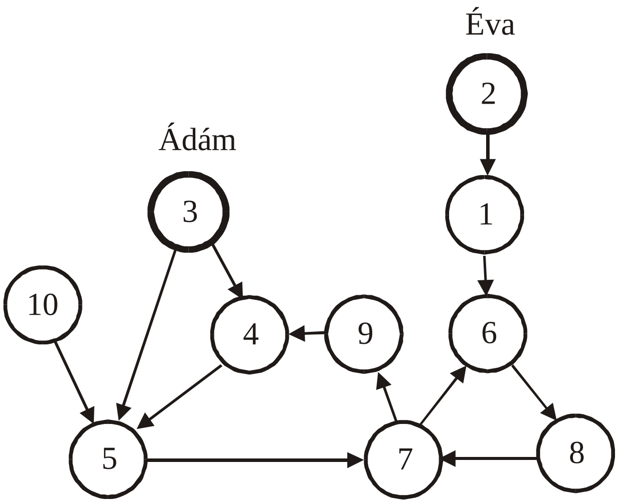
Készíts programot, amely meghatároz egy találkahelyet, ahol Ádám és Éva találkozat, és meg is ad egy-egy útvonalat Ádám és Éva számára!

A *TALALKA.BE* állomány első sora a városok számát (1≤N≤200), Éva tartózkodási helyét, Ádám tartózkodási helyét (1≤E≠A≤N) és a járatok (1≤M≤3000) számát tartalmazza. A további M sor mindegyikébe egy járat indulási és érkezési állomása (1≤Ind,Érk≤N) van. Bármely i és j városra legfeljebb egy járat van i-ből j-be.

A *TALALKA.KI* állomány első sorába a 0 0 számpárt kell írni, ha nem tudnak találkozni! A 0 M számpárt kell írni, ha Ádám el tud jutni vonattal Éva városába, és ekkor a második sor pontosan M számot tartalmazzon, azon városok sorozatát, amelyen keresztül Ádám eljut Éva városába! Ha Ádám nem tud eljutni Éva városába, de Éva el tud jutni Ádám városába, akkor az állomány első sora egy K 0 számpárt tartalmazzon! A második sor ekkor pontosan K számot tartalmazzon, azon városok sorozatát, amelyen keresztül Éva eljut Ádám városába! Ha egyikük sem tud odamenni a másik városába, akkor az első sor olyan K M számpárt tartalmazzon, hogy Éva K város, Ádám pedig M város érintésével tud eljutni a találkahelyre! Ekkor a második sor Éva útvonalát, a harmadik pedig Ádám útvonalát tartalmazza! Az útvonalakba beleszámít Éva és Ádám kiindulási tartózkodási helye is.

Példa:

TALALKA.BE	TALALKA.KI
10 2 3 12	5 3
2 1	2 1 6 8 7
1 6	3 5 7
7 6	
6 8	
8 7	
7 9	
9 4	
5 7	
10 5	
3 5	
3 4	
4 5	



2006. Harmadik forduló

Ötödik-nyolcadik osztályosok

1. feladat: Számrendszer (20 pont)

Ismerjük az A (1≤A≤32 767) pozitív egész számot, valamint azt is tudjuk, hogy az A számjegyeinek összege valamilyen számrendszerben éppen S. A számrendszer alapszáma biztosan nem nagyobb A+1-nél.

Készíts programot, amely beolvassa az adatokat, majd kiírja a képernyőre, hogy az A szám számjegyeinek összege mely számrendszerekben lehet éppen S!

Példa:

```
    Bemenet:  A=127, S=3
    Kimenet: Lehetséges számrendszer=5, 63, 125
```

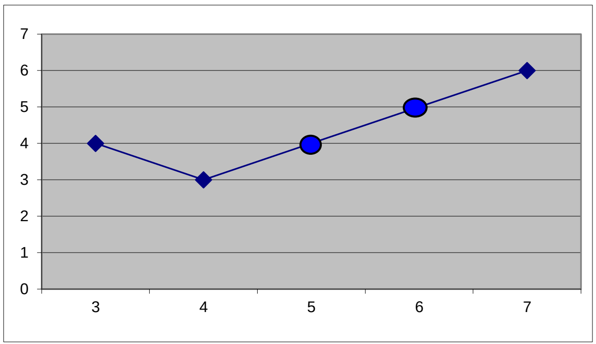
Magyarázat: 127=**1***5^3+**0***5^2+**0***5+**2**, 127=**2***63+**1**, 127=**1***125+**2**

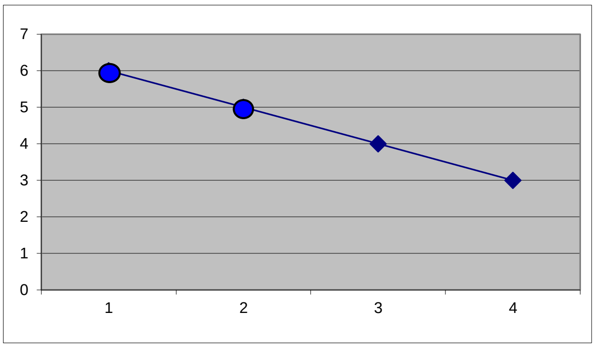
2. feladat: Mérés (32 pont)

N napon át mértük a Balaton hőmérsékletét ($2 \leq N \leq 100$). Ez minden nap pozitív valós szám volt. Néhány napon azonban elfelejtettünk mérni, ezekre a napokra -1-es értéket kaptunk. legfeljebb N-2 darab -1-es lehet.

Készíts programot, amely beolvassa az adatokat, majd a -1-esek helyére közelítő értéket számol, lineárisan közelítve a hiányzó értékeket, s a kapott méréssorozatot kiírja a képernyőre!

A lineáris közelítés a következőt jelenti: Ha a bemeneti adatsor: -1,-1,4,3,-1,-1,6,-1,-1, akkor az 1. ábrán látható módon az ötödik helyre bekerül a 4, a hatodik helyre pedig az 5. A 2. ábra mutatja, hogy a bal szélen levő -1-esek helyére a 6 és az 5 kerül. Hasonló elven a jobb szélen levő -1-esek helyére 7-et és 8-at kell tenni.





Példa:

```
Bemenet: -1,-1,4,3,-1,-1,6,-1,-1
Kimenet: 6,5,4,3,4,5,6,7,8
```

3. feladat: Tükörszó (23 pont)

Egy szóból a tükörképét úgy kapjuk meg, hogy a betűk sorrendjét megfordítjuk (például: mád – dám). A jó tükörszóban azonban a két vagy három karakterből álló mássalhangzók jó sorrendben maradnak (például ősz – sző). Sőt a hosszú többjegyű mássalhangzókat is meg kell hagyni (például: hosszú – ússzoh)!

Készíts programot, amely beolvas egy szót, majd kiírja a tükörszavát!

Kilencedik-tizedik osztályosok

1. feladat: Ismerősök (18 pont)

Egy Internetes ismerős keresési fórumra N ember iratkozott fel. Mindegyik megadta, hogy a fórumon szereplők közül kiket ismer. Az ismeretség kölcsönös.

Készíts programot, amely megadja

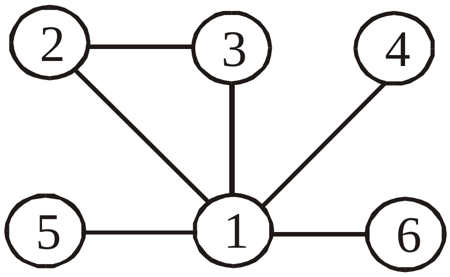
A. azokat az embereket, akik ismerik egymást, de más közös ismerősük nincs;

B. azokat az embereket, akik nem ismerik egymást, de minden ismerősük közös!

Az *ISMER.BE* állomány első sorában az emberek száma ($2 \leq N \leq 100$), és az ismeretségek száma ($0 \leq M \leq 5000$) van. A következő M sor mindegyikében két egymást ismerő ember sorszáma van.

Az *ISMER.KI* állomány első sorába az A részfeladat megoldásai K számát, a második sorába pedig az A feladat megoldásait kell írni! A harmadik sorba a B részfeladat megoldásai L száma kerüljön, a negyedik sorba pedig a B feladat megoldásai! A második sorba K, a negyedik sorba L számpár kerüljön, egy-egy szóközzel elválasztva, tetszőleges sorrendben!

Példa:



ISMER.BE	ISMER.KI
6 6	3
1 2	1 4 1 5 1 6
1 3	3
1 4	4 5 4 6 5 6
2 3	
1 5	
1 6	

2. feladat: Csoportok (18 pont)

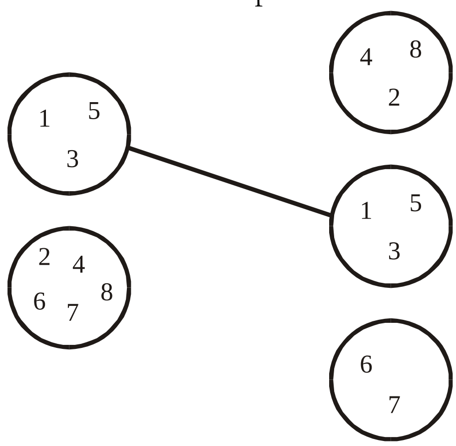
Egy iskola diákjai választhattak, hogy milyen nyelvet szeretnének tanulni, illetve hogy testnevelés órán milyen sportággal szeretnének foglalkozni. Minden diák egyetlen nyelvet és egyetlen sportágat választhatott.

Készíts programot, amely megadja, hogy hány olyan A nyelv és B sport pár van, hogy azok halmaza, akik az A nyelvet választották, megegyezik azok halmazával, akik a B sportot választották!

A *CSOPORT.BE* állomány első sorában a diákok (1≤M≤1000), a nyelvek (1≤N≤100) és a sportágak (1≤S≤100) száma van. A következő N sorban az egyes nyelveket, az azt követő S sorban pedig az egyes sportágakat választó tanulók sorszáma van. Minden egyes ilyen sor egy darabszámmal (DB) kezdődik, amelyet DB darab tanuló sorszáma követ.

Az *CSOPORT.KI* állomány egyetlen sorába a kétféle szempont szerint azonos csoportok számát kell írni!

Példa:



CSOPORT.BE	CSOPORT.KI
8 2 3	1
3 1 3 5	
5 2 4 8 7 6	
3 4 8 2	
3 5 1 3	
2 6 7	

(Az 1,3,5 sorszámú tanuló a nyelv szerint is és a sportág szerint is ugyanabban a csoportban van.)

3. feladat: Szemtanúk (19 pont)

A Rendőrség szemtanúkat keres egy rendezvényen történt gyanús események kivizsgálásához. A rendezvény szervezői feljegyezték minden vendégről, hogy mikor érkezett és mikor távozott.

A Rendőrség ki akar választani a lehető legkevesebb számú vendéget úgy, hogy minden gyanús esemény időpontjához legyen olyan kiválasztott vendég, aki jelen volt az esemény időpontjában. Ha egy gyanús esemény az X időpontban történt, akkor az olyan vendég, aki az E időpontban érkezett és a T időpontban távozott szemtanúja volt az eseménynek, ha E≤X≤T.

Készíts programot, amely megadja, hogy minimálisan hány vendéget kell kiválasztania a Rendőrségnek, hogy minden gyanús esemény időpontjához legyen olyan kiválasztott vendég, aki jelen volt az esemény időpontjában!

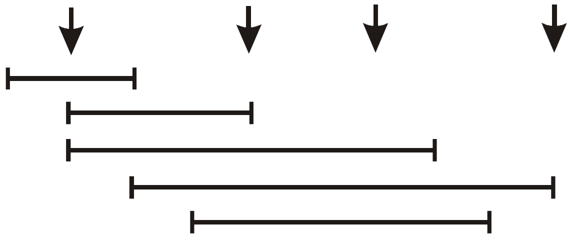
A *RKERES.BE* állomány első sorában a vendégek (1≤M≤1000) és a gyanús események (1≤N≤300) száma van. A következő M sor mindegyikében egy vendég érkezési és távozási időpontja (1≤E<T≤20 000) van. A vendégek adatai érkezési idejük szerint monoton nemcsökkenő sorrendben vannak. Az utolsó sor N pozitív egész számot tartalmaz, a gyanús események időpontjait monoton növekvő sorrendben.

Az *RKERES.KI* állomány első sorába a kiválasztandó vendégek K minimális számát kell írni! A második sorba kell kiírni a kiválasztott vendégek sorszámait (tetszőleges sorrendben)! Ha nincs megoldás, akkor az első sorba a 0 számot kell írni! Több megoldás esetén bármelyik megadható.

Példa:

```
RKERES.BE
5 4
1 3
2 5
2 8
3 10
4 9
2 5 7 10
```

```
RKERES.KI
2
1 4
```



4. feladat: Hálózat (20 pont)

Minden számítógépes hálózat csomópontokból és bizonyos csomópont-párokat összekötő kétirányú adatátvitelt biztosító közvetlen vonalakból épül fel. Egy hálózat összefüggő, ha bármely két csomópontja között van közvetlen vonalakból álló útvonal.

Készíts programot, amely megadja, hogy minimálisan hány új közvetlen vonalat kell kiépíteni, hogy a hálózat összefüggő legyen!

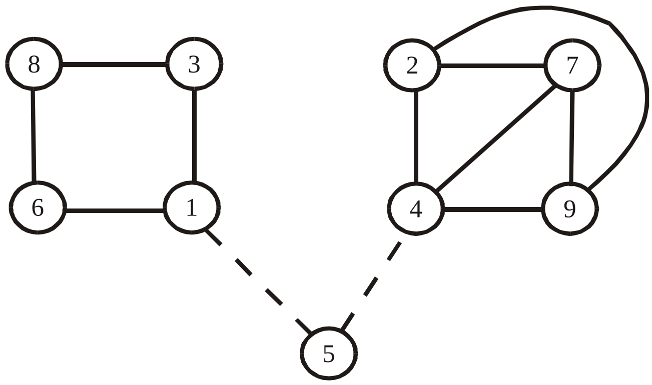
A *HALOZAT.BE* állomány első sorában a csomópontok (1≤N≤100) és a már kiépített közvetlen vonalak (1≤M≤5000) száma van. A csomópontokat az 1...N számokkal azonosítjuk. A következő M sor mindegyikében két egész szám van, egy már kiépített közvetlen vonal két végpontja.

Az *HALOZAT.KI* állomány első sorába a kiépítendő új vonalak K minimális számát kell írni! A további K sor mindegyike egy kiépítendő új vonalat tartalmazzon, annak a két csomópontnak a sorszámát, amely között az új vonalat ki kell építeni! Több megoldás esetén bármelyik megadható.

Példa:

```
HALOZAT.BE
9 10
1 3
2 7
6 8
8 3
1 6
4 9
4 2
4 7
7 9
2 9
```

```
HALOZAT.KI
2
1 5
5 4
```



Tizenegyedik-tizenharmadik osztályosok

1. feladat: Barátok (15 pont)

Egy osztályban szociometriai felmérést végeztek. Minden tanuló megadta egy (-1000,1000)-es skálán, hogy az osztályban kit mennyire szeret. A pozitív számok rokonszenvet, a negatívak pedig ellenszenvet jelentenek. A baráti csoportok úgy alakulnak, hogy mindenki a neki legszimpatikusabb tanulóval van egy csoportban, ha van neki egyáltalán szimpatikus tanuló az osztályban.

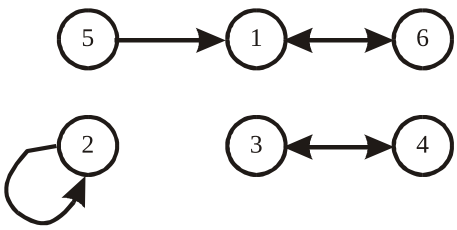
Készíts programot, amely megadja az osztály baráti csoportjait!

A *BARATOK.BE* állomány első sorában a tanulók száma (2≤N≤1000) van. A következő N sor mindegyikében N szimpátia érték van, az i-edik sor j-edik száma azt jelenti, hogy az i-edik tanulónak mennyire szimpatikus a j-edik tanuló. Saját magát mindenki biztosan 0 szimpátiára értékeli. Egy soron belül egyforma számok nem lehetnek!

A *BARATOK.KI* állomány első sorába a baráti csoportok K számát kell írni! A következő K sor mindegyikébe egy-egy baráti csoport tanulói sorszáma kerüljön! Mindegyik sorban annyi tanuló sorszáma legyen, ahányan abba a baráti csoportba tartoznak! A baráti csoportok tagjai tetszőleges sorrendben kiírhatók.

Példa:

BARATOK.BE	BARATOK.KI
6	3
0 2 3 4 5 6	1 5 6
-1 0 -3 -4 -5 -6	2
-1 -2 0 10 1 2	3 4
1 2 10 0 4 5	
6 5 4 3 0 1	
6 5 4 3 2 0	



2. feladat: Részhalmazok (15 pont)

Egy iskola diákjai választhattak, hogy milyen nyelvet szeretnének tanulni, illetve hogy testnevelés órán milyen sportággal szeretnének foglalkozni. Minden diák egyetlen nyelvet és egyetlen sportágat választhatott.

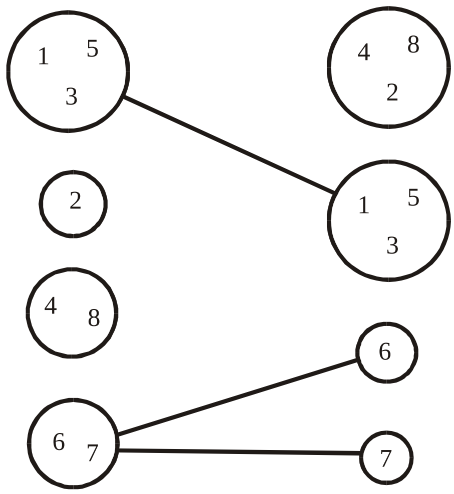
Készíts programot, amely megadja, hogy hány olyan nyelv van, amelyre igaz, hogy ha egy valamilyen sportággal foglalkozó tanuló ezt a nyelvet választotta, akkor mindenki, aki ezzel a sportággal foglalkozik, is ezt a nyelvet választotta!

A *RESZH.BE* állomány első sorában a diákok ($1 \le M \le 1000$), a nyelvek ($1 \le N \le 100$) és a sportágak ($1 \le S \le 100$) száma van. A következő N sorban az egyes nyelveket, az azt követő S sorban pedig az egyes sportágakat választó tanulók sorszáma van. Minden egyes ilyen sor egy darabszámmal (DB) kezdődik, amelyet DB darab tanuló sorszáma követ.

Az *RESZH.KI* állomány egyetlen sorába az adott tulajdonságú nyelv szerinti csoportok számát kell írni!

Példa:

RESZH.BE	RESZH.KI
8 4 4	2
3 1 3 5	
1 2	
2 4 8	
2 7 6	
3 4 8 2	
3 5 1 3	
1 6	
1 7	



(Az 1,3,5 sorszámú tanuló a nyelv szerint is és a sportág szerint is ugyanabban a csoportban van. A harmadik és negyedik sportágat választók részhalmazának uniója éppen a negyedik nyelvet tanuló részhalmaz: $\{7,6\}=\{6\}\cup\{7\}$.)

3. feladat: Vásárlás (15 pont)

Egy kiránduláson N helyen tudunk dobozos üdítőitalt vásárolni. A megvett italos dobozokat egy K doboz kapacitású hátizsákba tesszük. Egy doboz üdítőitalt 1 km megtétele alatt iszunk meg.

Készíts programot, amely a boltok távolságának ismeretében kiszámítja, hogy minimum hány boltban kell üdítőitalt vásárolnunk, hogy végigihassuk az utat és az N+1-edik helyre érve éppen elfogyjon az utolsó doboz üdítő!

A *VASAR.BE* állomány első sorában a boltok száma (1≤N≤1000) és a hátizsák kapacitása (1≤K≤100) van. A következő N sor mindegyikében a következő állomás távolsága, valamint az állomáson megvásárolható üdítőital dobozok száma van.

A *VASAR.KI* állomány egyetlen sorába a vásárlások minimális számát kell írni! Ha a feladat nem oldható meg, akkor a -1-es számot írjuk ki!

Példa:

VASAR.BE	VASAR.KI
6 10	3
4 10	
3 3	
2 3	
2 2	
2 2	
2 2	

4. feladat: Színezés (15 pont)

Egy N emeletes fehér épület bizonyos emeleteit a szépség kedvéért pirosra szeretnénk festeni. Csak olyan festést tartunk elfogadhatónak, amelynél szomszédos szinteket nem festünk pirosra. A színezéseket N+1 elemű 0-1 számsorozattal kódoljuk: 1-es jelöli a piros, 0-s pedig a fehér színű emeletet. Az első szám jelenti a földszint, az utolsó pedig az N. emelet színét.

Készíts programot, amely megadja, hogy az épület hányféleképpen színezhető ki, valamint a lexikografikus (ábécé szerinti) K-adik színezést!

A *SZIN.BE* állomány első sorában az emeletek száma (0≤N≤40) és K (1≤K≤100 000 000) értéke van.

A *SZIN.KI* állomány első sorába a színezések lehetséges számát kell írni! A második sorba a K. színezést kell kiírni: az emeletek növekvő sorrendjében N+1 darab egész számot, ahol 0 jelöli a fehér, 1 pedig a pirosra festett szintet!

Példa:

SZIN.BE	SZIN.KI
3 4	8
	0 1 0 0

Sorrendben a jó festések: 0 0 0 0, 0 0 0 1, 0 0 1 0, 0 1 0 0, 0 1 0 1, 1 0 0 0, 1 0 0 1, 1 0 1 0.

5. feladat: Két útvonal (15 pont)

Egy vállalatnak N városban van telephelye. A központi telephely az 1. városban van. Alkatrészeket kell kiszállítani a központi telephelyről két különböző, U és V városba két kamionnal, az egyiknek az U, a másiknak a V városba kell mennie. Ismerjük, hogy mely városok között van közvetlen út. A korlátozások miatt a két kamion olyan útvonalon közlekedhet, amely különböző városokon keresztül halad.

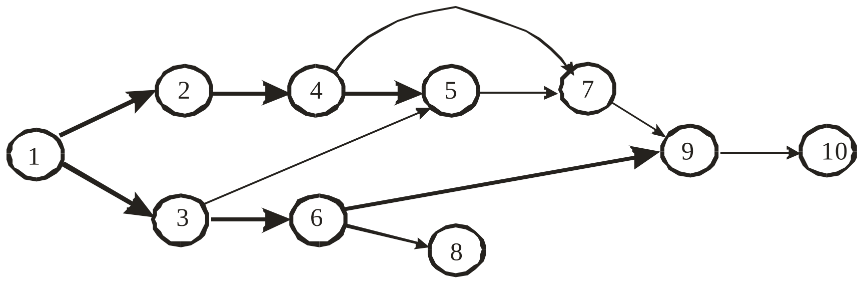
Készíts programot, amely kiszámít egy olyan U-ba és egy olyan V-be vezető útvonalat, hogy a két útvonalban csak a kiindulási pont (a központi telephely) közös!

A *KETUT.BE* állomány első sorában a városok száma (3≤N≤100), az U és V város sorszáma (2≤U,V≤N,U≠V) és a közvetlen utak (2≤M≤3000) száma van. A következő M sor mindegyikében két szám van: X Y ami azt jelenti, hogy X városból van Y városba út, amin X-ből Y-ba lehet menni, de fordítva nem. Minden közvetlen útra teljesül, hogy X<Y:

A *KETUT.KI* állomány első sorába az U-ba vezető útvonalon lévő városok r számát, és a V-be vezető útvonalon lévő városok s számát kell írni (beleértve a kiindulási központi telephely 1 sorszámát)! A második sor az U-ba vezető, a harmadik pedig a V-be vezető útvonalat tartalmazza! Ha nincs megoldás, akkor a `0 0` számpárt kell kiírni az első sorba! Több megoldás esetén bármelyik megadható.

Példa:

KETUT.BE	KETUT.KI
10 9 5 12	4 4
1 2	1 3 6 9
1 3	1 2 4 5
2 4	
3 6	
4 5	
5 7	
6 8	
7 9	
3 5	
4 7	
6 9	
9 10	



A verseny végeredménye:

I. korcsoport

1. Hegedűs Tamás	Földes Ferenc Gimnázium, Miskolc
2. Éles András	Fazekas Mihály Gimnázium, Debrecen
Mészáros András	Révai Miklós Gimnázium, Győr
Szendrei Péter	Károlyi István 12 évfolyamos Gimnázium, Budapest
5. Pólya Málna	Krúdy Gyula Gimnázium, Nyíregyháza
6. Pálinkás István	5.sz. Általános Iskola, Gyula
7. Szigetvári Áron	Fazekas Mihály Gimnázium, Budapest
Rutai Richárd	Szent István Gimnázium, Budapest
9. Wagner Zsolt	Fazekas Mihály Gimnázium, Budapest
10.Seller Tamás	Radnóti Miklós Gimnázium, Szeged

II. korcsoport

1. Szalkai Balázs	Lovassy László Gimnázium, Veszprém
2. Nagy Gergely Gábor	Fazekas Mihály Gimnázium, Budapest
3. Danner Gábor	Ságvári Endre Gimnázium, Szeged
4. Korándi Dániel	Fazekas Mihály Gimnázium, Budapest
5. Grósz Dániel	Fazekas Mihály Gimnázium, Budapest
6. Peregi Tamás	Berzsenyi Dániel Gimnázium, Budapest
7. Kis Dániel	Neumann János Középiskola és Kollégium, Eger
Eisenberger András	Fazekas Mihály Gimnázium, Budapest
9. Pósfai Gergely	Ságvári Endre Gimnázium, Szeged
10.Gévay Gábor	Ságvári Endre Gimnázium, Szeged

III. korcsoport

1.	Ludányi Ákos	Neumann János Középiskola és Kollégium, Eger
2.	Nikházy László	Kazinczy Ferenc Gimnázium és Szakközépiskola, Győr
3.	Jámbor Attila	Neumann János Középiskola és Kollégium, Eger
4.	Vincze János	Fazekas Mihály Gimnázium, Debrecen
5.	Paulin Roland	Fazekas Mihály Gimnázium, Budapest
6.	Engedy Balázs	Dobó István Gimnázium, Eger
7.	Lehel Gábor	Kazinczy Ferenc Gimnázium és EüSzKI, Győr
8.	Jobbágy László	Árpád Vezér Gimnázium, Sárospatak
9.	Kormányos Balázs	Radnóti Miklós Gimnázium, Szeged
10.	Acsai Péter	Arany János Református Gimnázium, Nagykőrös
11.	Soltész Zoltán	ELTE Apáczai Csere János Gimnázium, Budapest
12.	Ozsvárt László	Verseghy Ferenc Gimnázium, Szolnok
13.	Poór Márk	Neumann János Középiskola és Kollégium, Eger
14.	Mészáros Balázs	Berzsenyi Dániel Gimnázium, Budapest
15.	Kunovszki Péter	Kisfaludy Károly Gimnázium, Mohács

2007. Első forduló

Ötödik-nyolcadik osztályosok

1. feladat: Képlet (20 pont)

Az alábbi programrészlet az `x` és az `y` változókból számolja ki `a`, `b`, `c`, `d` értékét:

```
p:=x+y; q:=x-y; r:=abs(p); s:=abs(q)
a:=(p+s)/2; b:=(p-s)/2
c:=abs(s-r)/2; d:=x-(q+s)/2
```

A. Mi lesz az `a`, `b`, `c`, `d` változók értéke, ha x=8, y=-5?

B. Fogalmazd meg általánosan, hogyan függ az `a`, `b`, `c`, `d` értéke `x`-től és `y`-tól!

2. feladat: Paradicsomok (19 pont)

N paradicsom érik egymás mellett, egyesek közülük már pirosak. A paradicsomok a következő szabályok szerint érnek be:

A. Egy zöld paradicsom egy időegység múlva piros lesz, ha ugyanolyan távolságra van tőle balra és jobbra is a legközelebbi piros paradicsom, de egyik sem a szomszédja.

B. Ha a sorban az első két piros paradicsom távolsága egymástól tetszőleges K érték, akkor az elsőtől K-1 távolságra balra levő zöld paradicsom piros lesz.

C. Ha a sorban az utolsó két piros paradicsom távolsága egymástól tetszőleges K érték, akkor az utolsótól K+1 távolságra jobbra levő zöld paradicsom piros lesz.

Írd a Z betűk alá, hogy az alábbi paradicsom-sorokban mely időegységben lesznek a paradicsomok pirosak!

```
A: Z Z Z Z P Z Z Z Z Z Z Z P P Z Z Z P Z Z Z Z Z Z
           0               0 0       0

B: Z Z Z P Z Z Z Z Z P P Z Z Z P Z P Z Z Z Z Z Z Z
         0           0 0       0   0
```

Példa:

```
0. időegység: Z Z Z Z Z P Z Z Z P
1. időegység: Z Z P Z Z P Z P Z P
2. időegység: P Z P Z Z P Z P Z P
a számok:     2   1     0   1   0
```

3. feladat: Kétirányból (24 pont)

A következő 2 algoritmus az S szó betűit vizsgálja, egyszerre haladva elölről hátra, illetve hátulról előre.

```
Alfa(S):
  i:=1; j:=hossz(S)
  Ciklus amíg i≤j és S(i)=S(j)
    i:=i+1; j:=j-1
  Ciklus vége
  Ha i>j akkor Ki: ″JÓ″ különben Ki: ″ROSSZ″
Eljárás vége.
```

```
Béta(S):
  i:=1; j:=hossz(S)
  Ciklus amíg i≤j és S(i)≠S(j)
    i:=i+1; j:=j-1
  Ciklus vége
  Ha i≤j akkor Ki: i különben Ki: ”ROSSZ”
Eljárás vége.
```

A. Mit írnak ki a fenti eljárások, ha S=”kertel”?

B. Milyen szavakra írja ki az `Alfa` eljárás a JÓ választ? Adj konkrét példát is, és fogalmazd meg általánosan is!

C. Milyen szavakra ír ki a `Béta` eljárás sorszámot, és mi ez a sorszám? Adj konkrét példát is, és fogalmazd meg általánosan is!

4. feladat: Két index (37 pont)

A következő 3 algoritmus valamilyen értelemben legnagyobb elemeket próbál megkeresni a csak pozitív számokat tartalmazó N elemű X tömbben.

```
Első(N,X,A,B):
  A:=0; B:=0; X(0):=-1
  Ciklus i=1-től N-ig
    Ha X(i)>X(A) akkor A:=I
  Ciklus vége
  Ciklus i=1-től N-ig
    Ha X(i)>X(B) és X(i)<X(A) akkor B:=I
  Ciklus vége
Eljárás vége.
```

```
Második(N,X,A,B):
  A:=0; B:=0; X(0):=-1
  Ciklus i=1-től N-ig
    Ha X(i)>X(A) akkor B:=A; A:=I
    különben ha X(i)>X(B) akkor B:=I
  Ciklus vége
Eljárás vége.
```

```
Harmadik(N,X,A,B):
  A:=0; B:=0; X(0):=-1
  Ciklus i=1-től N-ig
    Ha X(i)≥X(A) akkor B:=A; A:=I
  Ciklus vége
Eljárás vége.
```

A. Mi lesz az A és a B változó értéke az egyes algoritmusok esetén, ha N=6, X=(3,5,2,5,4,5)?

B. Adj N=5-re olyan X vektort, amelyre az első algoritmus esetén B=0 marad!

C. Adj N=5-re olyan X vektort, amelyre a harmadik algoritmus esetén B=0 marad!

D. Fogalmazd meg szavakkal, hogy az egyes algoritmusok esetén A, illetve B milyen tulajdonságú X-beli elem indexe lesz a futás végén!

Kilencedik-tizedik osztályosok

1. feladat: Hamming kód (14 pont)

Egy négybites üzenetet a $b_1b_2b_3b_4$ bitekből áll. Az üzenet továbbításakor legfeljebb 1 hiba keletkezhet, azaz lehet, hogy egy 1-es bit 0-ra, vagy egy 0-s bit 1-esre változik. Ennek felismeréséhez újabb 3 bitet számolunk ki az alábbi képletekkel:

$b_5=(b_2+b_3+b_4) \bmod 2$; $b_6=(b_1+b_3+b_4) \bmod 2$; $b_7=(b_1+b_2+b_4) \bmod 2$.

Így a továbbított kód a $b_1b_2b_3b_4b_5b_6b_7$ bitekből fog állni.

A. Az 1100011, 1010110, 1111111, 0101010, 0110011 üzenetekből melyek helyesek és melyek hibásak?

B. Add meg a kódját az 1001, illetve az 1010 üzenetnek!

2. feladat: Számjegyek vizsgálata (23 pont)

Az alábbi két eljárás a 0 és 9 közötti értékeket tartalmazó `N` elemű `X` vektor alapján határozza meg `k`, `v` és `h` értékét.

```
Alfa(N,X,k,v,h):
  k:=1; v:=1; h:=1; j:=1; a:=1
  Ciklus i=1-től N-ig
    Ha X(i)=X(i+1) akkor a:=a+1
    különben ha a>h akkor k:=j; v:=i; h:=a
             a:=1; j:=i+1
  Ciklus vége
Eljárás vége.
```

```
Béta(N,X,k,h):
  T:=(0,...0); k:=0; h:=0
  Ciklus i=1-től N-ig
    T(X(i)):=T(X(i))+1
    Ha T(X(i))>h akkor k:=X(i); h:=T(X(i))
  Ciklus vége
Eljárás vége.
```

A. Mi lesz `k`, `v` és `h` értéke az egyes eljárások esetén, ha `N`=10, `X`=(1,3,5,5,5,4,3,3,2,3)?

B. Fogalmazd meg általánosan a két eljárás feladatát, azaz hogyan függ `k`, `v` és `h` értéke a bemenő adatoktól!

3. feladat: Unió (17 pont)

Az {1…N} halmazt felbontjuk közös rész nélküli részhalmazokra. A `T` vektorban ezen részhalmazokat írjuk le úgy, hogy minden egyes részhalmazból kijelölünk egy-egy elemet, ami a részhalmazt azonosítja, azaz a részhalmaz összes `i` eleméhez ugyanaz a `T(i)` érték tartozik. Ha például a {3,5} részhalmazt nézzük és a kijelölt elem az 5, akkor a `T` vektorban T(3)=5 és T(5)=5 lesz. Amíg egy részhalmaz 1 elemű, addig ezt a T(i)=-1 érték jelzi.

Két részhalmaz összevonásához (unió) csupán a két részhalmazt azonosító elemet kell megadnunk az alábbi eljárásnak:

```
Unió(A,B):
  Ha T(A)=-1 akkor Ha T(B)=-1 akkor T(A):=B; T(B):=B
                          különben T(A):=T(B)
  különben ha T(B)=-1 akkor T(B):=T(A)
  különben X:=T(B)
           Ciklus i=1-től N-ig
             Ha T(i)=X akkor T(i):=T(A)
           Ciklus vége
Eljárás vége.
```

A. Add meg, hogy mi lesz a 10 elemű T halmaz az alábbi műveletek elvégzése után (minden lépés után külön-külön, azaz 7 adatsort kell leírni):

Unió(1,5); Unió(2,4); Unió(8,9); Unió(1,9); Unió(8,5); Unió(7,6),Unió(2,9)

B. Milyen esetben nem változtat a T tömbön az eljárásban szereplő ciklus?

C. Hogyan kellene kijavítani az eljárást, hogy a B esetbeli feltétel esetén gyorsabb megoldást kapjunk?

4. feladat: Multihalmaz (25 pont)

Két állattenyésztő az állatait ábécésorrendben tartja nyilván, mindegyik állatból lehet több is (pl. kecske,ló,ló,ló,nyúl,nyúl). Az egyik állatai az N elemű A vektorban, a másiké pedig az M elemű B vektorban vannak.

```
Első(N,A,M,B,K,C):
  i:=1; j:=1; K:=0
  Ciklus amíg i≤N és j≤M
    Ha A(i)<B(j) akkor i:=i+1
    különben ha A(i)>B(j) akkor j:=j+1
    különben K:=K+1; C(K):=A(i); i:=i+1; j:=j+1
  Ciklus vége
Eljárás vége.
```

```
Második(N,A,M,B,K,C):
  i:=1; j:=1; K:=0; A(N+1):="zzzzz"; B(M+1):="zzzzz"
  Ciklus amíg i<N+1 vagy j<M+1
    Ha A(i)<B(j) akkor K:=K+1; C(K):=A(i); i:=i+1
    különben ha A(i)>B(j) akkor K:=K+1; C(K):=B(j); j:=j+1
    különben K:=K+1; C(K):=A(i); i:=i+1; j:=j+1
  Ciklus vége
Eljárás vége.
```

A. Mi lesz az egyes eljárások hatására K és a C vektor értéke, ha N=6, A=(kecske,ló,ló,ló, nyúl,nyúl), M=5, B=(kecske,kecske,ló,szamár,szamár)?

B. Mi lenne a fenti bemenetre az eredménye a második eljárásnak, ha a ciklusfeltétele (i≤N és j≤M) lenne?

C. Fogalmazd meg szavakkal, hogy az egyes eljárások esetén a két vektor mely tulajdonságú elemei kerülnek be a C vektorba!

D. Milyen tulajdonságú A és B vektor esetén lesz a két eljárás végeredménye ugyanaz?

5. feladat: Kép (21 pont)

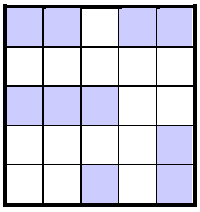
Egy 5*5-ös képre a következő ciklikus léptetés (minden képpont eggyel elmozdul valamilyen irányba, a kilépők a túlsó oldalon belépnek) műveleteket alkalmazhatjuk:

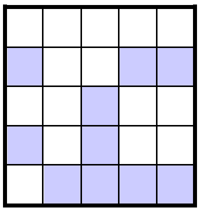
B: ciklikus léptetés balra

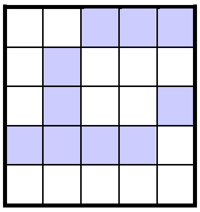
J: ciklikus léptetés jobbra

L: ciklikus léptetés lefelé,

F: ciklikus léptetés felfelé.

KÉP:


KÉP1:


KÉP2:


A. Minimum hány művelettel keletkezhetett a KÉP-ből KÉP1, illetve KÉP2?

B. Adj meg egy minimális műveletsort, amivel KÉP1 és KÉP2 a KÉP-ből keletkezhetett!

C. Ha egy minta az elsőből a C1: `JJJLLBJLBBF`, a C2: `JJJJLLFJJJF`, illetve a C3: `LLJJLLJJ` műveletsorral keletkezett, akkor mi lehet a minimális műveletsor, ami ugyanezt az eredményt adja?

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Hamming kód (16 pont)

Egy négybites üzenetet a $b_1b_2b_3b_4$ bitekből áll. Az üzenet továbbításakor legfeljebb 1 hiba keletkezhet, azaz lehet, hogy egy 1-es bit 0-ra, vagy egy 0-s bit 1-esre változik. Ennek felismeréséhez újabb 3 bitet számolunk ki az alábbi képletekkel:

$b_5=(b_2+b_3+b_4)$ mod 2; $b_6=(b_1+b_3+b_4)$ mod 2; $b_7=(b_1+b_2+b_4)$ mod 2.

Így a továbbított kód a $b_1b_2b_3b_4b_5b_6b_7$ bitekből fog állni.

A. Az 1100011, 1010110, 1111111, 0101010, 0110010 üzenetekből melyek helyesek és melyek hibásak?

B. Add meg a hibásakról, hogy melyik bitjük hibás, ha tudjuk, hogy pontosan 1 hibás bit van bennük, és indokold meg, hogy miért!

C. Add meg a kódját az 1001, illetve a 0101 üzenetnek!

2. feladat: Számjegyek vizsgálata (20 pont)

Az alábbi két eljárás a 0 és 9 közötti értékeket tartalmazó `N` elemű `X` vektor alapján határozza meg `k`, `v` és `h` értékét.

```
Alfa(N,X,k,v,h):
  i:=1; k:=i; v:=1; h:=1; a:=0;
  Ciklus j=1-től N-ig
    Ha (X(i+1)-X(i))*(X(j+1)-X(j))>0 akkor a:=a+1
    különben ha a>h akkor k:=i; v:=j; h:=a
    különben i:=j; a:=0
  Ciklus vége
Eljárás vége.
```

```
Béta(N,X,k,v,h):
  T:=(0,...0); k:=1; v:=1; h:=1
  Ciklus i=1-től N-ig
    Ha i-T(X(i))>h és T(X(i))>0
       akkor k:=T(X(i)); v:=i; h:=i-T(X(i))
    T(X(i)):=i
  Ciklus vége
Eljárás vége.
```

A. Mi lesz `k`, `v` és `h` értéke az egyes eljárások esetén, ha N=10, X=(1,3,5,6,5,4,3,2,3,4)?

B. Fogalmazd meg általánosan a két eljárás feladatát, azaz hogyan függ `k`, `v` és `h` értéke a bemenő adatoktól!

3. feladat: Táblázat (25 pont)

A táblázat típusnál egy függvényt használunk, amely kiszámolja egy elem címét egy tömbben, ha az a cím üres, akkor az elemet oda helyezi, ha pedig foglalt, akkor valahova a közelébe. Kezdetben a 10 elemű T vektor minden eleme `-1` értékű. Az alábbi két eljárást írtuk meg (nemnegatív A paramétert használva):

```
Beszúr(A):
  i:=A mod 10
  Ciklus amíg T(i)≠-1 és T(i)≠-2
    i:=(i+1) mod 10
  Ciklus vége
  T(i):=A
Eljárás vége.
```

```
Töröl(A):
  i:=A mod 10
  Ciklus amíg T(i)≠A
    i:=(i+1) mod 10
  Ciklus vége
  T(i):=-2
Eljárás vége.
```

A. Mi lesz a T vektor értéke az egyes eljáráshívások után, ha az eljárásokat a következő sorrendben hívjuk?

Beszúr(100); Beszúr(22); Beszúr(10); Beszúr(30); Beszúr(11); Töröl(10); Beszúr(21)

B. Milyen esetben lehet az első eljárás ciklusa végtelen?

C. Milyen esetben lehet a második eljárás ciklusa végtelen?

D. Hogyan kell kijavítani őket, hogy ne legyen végtelen ciklus?

4. feladat: Multihalmaz (24 pont)

Két állattenyésztő az állatait ábécésorrendben tartja nyilván, mindegyik állatból lehet több is (pl. kecske,ló,ló,ló,nyúl,nyúl). Az egyik állatai az N elemű A vektorban, a másiké pedig az M elemű B vektorban vannak.

```
Első(N,A,M,B,K,C):
  i:=1; j:=1; K:=0; B(M+1):="zzzzz"
  Ciklus amíg i≤N és j≤M+1
    Ha A(i)<B(j) akkor K:=K+1; C(K):=A(i); i:=i+1
    különben ha A(i)>B(j) akkor j:=j+1
    különben i:=i+1; j:=j+1
  Ciklus vége
Eljárás vége.
```

```
Második(N,A,M,B,K,C):
  i:=1; j:=1; K:=0; A(N+1):="zzzzz"; B(M+1):="zzzzz"
  Ciklus amíg i<N+1 vagy j<M+1
    Ha A(i)<B(j) akkor K:=K+1; C(K):=A(i); i:=i+1
    különben ha A(i)>B(j) akkor K:=K+1; C(K):=B(j); j:=j+1
    különben i:=i+1; j:=j+1
  Ciklus vége
Eljárás vége.
```

A. Mi lesz az egyes eljárások hatására K és a C vektor értéke, ha N=6, A=(kecske,ló,ló,ló,nyúl,nyúl), M=5, B=(kecske,kecske,ló,szamár,szamár)?

B. Mi lenne a fenti bemenetre az eredménye a második eljárásnak, ha a ciklusfeltétele (i≤N és j≤M) lenne?

C. Mi lenne az első eljárás eredménye N=5, A=(kecske,kecske,ló,szamár,szamár), M=6, B=(kecske,ló,ló,ló,nyúl,nyúl), ha a ciklusfeltételből a j≤M+1-et lehagynánk?

D. Fogalmazd meg szavakkal, hogy az egyes eljárások esetén a két vektor mely tulajdonságú elemei kerülnek be a C vektorba!

E. Milyen tulajdonságú A és B vektorra lesz a két eljárás eredménye ugyanaz?

5. feladat: Kép (15 pont)

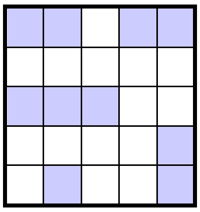
Egy 5*5-ös képre a következő műveleteket alkalmazhatjuk:

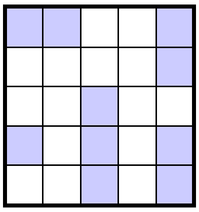
B: forgatás 90 fokkal balra (az óramutató járásával ellentétes irányba),

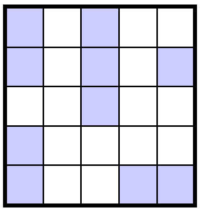
J: forgatás 90 fokkal jobbra (az óramutató járásával megegyező irányba),

K: középpontos tükrözés,

H: horizontális (vízszintes tengelyre) tükrözés, V: vertikális (függőleges tengelyre) tükrözés.


KÉP


KÉP1


KÉP2

A. Minimum hány művelettel keletkezhetett a KÉP-ből KÉP1 és KÉP2?

B. Adj meg egy minimális műveletsort, amivel KÉP1 és KÉP2 a KÉP-ből keletkezhetett!

C. Ha egy minta az elsőből a C1: `JJJHKBJVBBH`, a C2: `JJJJHVJHJH`, illetve a C3: `VBBVHJJJ` műveletsorral keletkezett, akkor mi lehet a minimális műveletsor, ami ugyanezt az eredményt adja?

2007. Második forduló

Ötödik-nyolcadik osztályosok

1. feladat: Japán kalendárium (20 pont)

A japán naptár 60 éves ciklusokat tartalmaz, az egyes éveket párosítják, s mindegyik párhoz valamilyen színt rendelnek (zöld, piros, sárga, fehér, fekete).

1,2,11,12, …,51,52: zöld évek

3,4,13,14,…,53,54: piros évek

5,6,15,16,…55,56: sárga évek

7,8,17,18,…57,58: fehér évek

9,10,19,20,…,59,60: fekete évek

Tudjuk, hogy 1984-ben indult az utolsó ciklus, amely 2043-ban fog véget érni (azaz 1984 és 1985 zöld év volt, 2043 pedig fekete év lesz).

Írj programot, amely megadja egy évről (`1800≤M≤ 2200`), hogy milyen színű!

2. feladat: Szilveszter (35 pont)

Készíts programot, amely megadja, hogy mennyi idő van még szilveszter éjfélig!

A program olvassa be, hogy melyik év (`1900≤év≤2100`), melyik hónap (`1≤hónap≤12`), melyik napján (`1≤nap≤a hónap napszáma`) vagyunk éppen, hány óra (`0≤óra≤23`), hány perc (`0≤perc≤59`), hány másodperc (`0≤másodperc≤59`) van most!

Ezután írja ki a szilveszter éjfélig hátralevő időt! Ha az idő valamelyik része 0 értékű, akkor azt nem szabad kiírni (lásd az első példát)!

Példák:

```
év=2007, hónap=3, nap=1, óra=0, perc=0; másodperc=0 → 10 hónap
év=2007, hónap=1, nap=13, óra=9, perc=59; másodperc=50 →
    11 hónap 18 nap 14 óra 10 másodperc
```

3. feladat: Mozaikszó (20 pont)

Különböző egyesületek, vállalatok gyakran választanak maguknak rövid nevet az elnevezésük néhány betűjéből (pl. BKV – Budapesti Közlekedési Vállalat). Az ilyen neveket hívjuk mozaikszónak. A mozaikszavak nem csak a szavak első betűjéből állhatnak (pl. MATÁV – Magyar Távközlési Rt.), sőt az sem biztos, hogy a szavak első néhány betűjéből. A mozaikszó betűi akkor is lehetnek nagybetűsek, ha az eredeti szóban az adott betűt kisbetűvel írják.

Készíts programot, amely egy szöveg és egy mozaikszó alapján megadja, hogy a mozaikszó a szöveg hányadik betűiből állhat! Feltehetjük, hogy biztosan van megoldás. Ha több megoldás is van, akkor bármelyik kiírható.

Példa:

`Budapesti Közlekedési Vállalat, BKV → 1 11 23`

`Magyar Távközlési Rt., MATÁV → 1 2 8 9 10 vagy 1 5 8 9 10`

Kilencedik-tizedik osztályosok

1. feladat: Hidak (16 pont)

Egy óceáni szigetország N szigetből áll, amelyek egy egyenes mentén helyezkednek el. A szigeteket hidakkal szeretnék összekötni. A hídépítő vállalatnak annyi pénze van, hogy az egyes tengerszakaszokra összesen K darab pillért építhet, amivel megoldhatja azt, hogy egyes hidakat több darabból építsék össze.

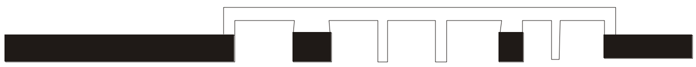
Készíts programot, amely megadja, hogy mely tengerszakaszokra hány pillért építsenek, ha azt szeretnék, hogy a leghosszabb híddarab a lehető legkisebb legyen!

A *HIDAK.BE* állomány első sorában a szigetek száma (1≤N≤1000) és a pillérek száma (1≤K≤100) van. A következő N sorban egy-egy sziget kezdő és végpozíciója található (0≤kezdőpozíció<végpozíció≤1 000 000). A pozíciókat az első sziget kezdetétől mérjük, azaz az első sziget kezdőpozíciója biztosan 0.

A *HIDAK.KI* állományba annyi sort kell írni, ahány tengerszakaszra helyezünk el pilléreket. Minden sorban a tengerszakasz sorszáma, valamint az oda építendő pillérek száma legyen, sorszám szerint növekvő sorrendben!

Példa:

```
HIDAK.BE      HIDAK.KI
4 3           2 2
0 20          3 1
25 28
42 44
52 58
```



2. feladat: Szövegkereső (15 pont)

Szövegek keresésekor megengedhető, hogy a keresőszóba a normál karakterek mellett ?, valamint * karaktert is tegyünk. A ? azt jelenti, hogy azon a pozíción tetszőleges karakter állhat, a * pedig tetszőleges számú karaktert helyettesíthet.

Készíts programot, amely kettő, legfeljebb egy * karaktert tartalmazó keresőszóról eldönti, hogy az elsővel megtalálható összes szó megtalálható-e a másodikkal is, és ha nem, akkor megad egy olyat, amit az elsővel megtalálunk, a másodikkal pedig nem!

A *RESZ.BE* állomány első sorában az első, a másodikban pedig a második keresőszó van, mindegyik csak nyomtatható karaktereket tartalmaz és legfeljebb 100 karakterből áll.

A *RESZ.KI* állomány első sorába egy, csak az első keresőszó által megtalálható halmazba tartozó szót kell írni (azaz olyat, amit a másodikkal nem találunk meg); egyébként pedig az IGEN szót!

Példa:

```
RESZ.BE       RESZ.KI
A?ma          Amma
Al?*
```

3. feladat: Hírek (16 pont)

Egy osztályban N tanuló van. Mindenki ismeri valahány osztálytársának telefonszámát (ez nem mindig kölcsönös). Azt mondjuk, hogy egy A tanuló egy B tanulónak el tud juttatni egy hírt, ha A fel tud hívni valakit, aki fel tud hívni egy további tanulót, …, s végül az utolsó fel tudja hívni B-t.

Készíts programot, amely megadja, hogy

A. hány olyan tanulópár van, akik üzenetet küldhetnek egymásnak;

B. hány olyan tanuló van, aki mindenkinek tud üzenetet küldeni!

A *HIR.BE* állomány első sorában a tanulók száma (1≤N≤100) van. A következő N sor mindegyikében N 'i' vagy 'n' betű van: Ha az i-edik sor j-edik karaktere 'i' betű, az azt jelenti, hogy az i-edik tanuló ismeri a j-edik tanuló telefonszámát.

A *HIR.KI* állomány első sorába az A feladat megoldását, a második sorába pedig a B feladat megoldását kell írni!

Példa:

HIR.BE	HIR.KI	
5	2	(az 1-3, illetve a 4-5)
i i i i i	2	(az 1. és a 3.)
n i n n n		
i n i n i		
n n n i i		
n n n i i		

4. feladat: Utazó (14 pont)

Két utazó nyaralása során különböző városokat látogat meg, mindegyikben eltöltenek néhány napot, s eközben szeretnének egymással minél többször találkozni.

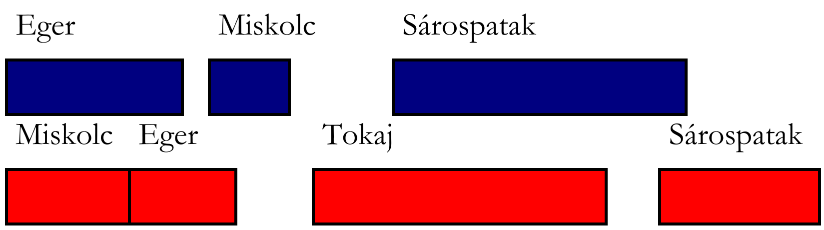
Készíts programot, amely megadja, hogy mely városokban találkozhattak utazásuk során!

Az *UTAZO.BE* állomány első sorában az első utazó által meglátogatott városok száma (1≤N≤100) van. A következő N sor mindegyike 2 számot és egy városnevet (A B C) tartalmaz, amelynek jelentése: az első utazó az A-adik naptól a B-edik napig a C városban tartózkodott (0≤A≤B≤10 000). Az ezt követő sorban van a második utazó által meglátogatott városok száma (1≤M≤100), amit M sorban követ az általa meglátogatott városok felsorolása, ugyanolyan formátumban, mint az első utazóé. Mindkét utazó adatai időben növekvő sorrendben vannak, egy várost csak egyszer látogatnak meg.

Az *UTAZO.KI* állomány első sorába azon városok K számát kell írni, amelyekben a két utazó találkozhatott, a következő K sorba pedig ezen városok nevét, a lehetséges találkozási sorrendben!

Példa:

UTAZO.BE	UTAZO.KI
3	2
1 6 Eger	Eger
8 10 Miskolc	Sárospatak
15 25 Sárospatak	
4	
1 4 Miskolc	
5 8 Eger	
12 22 Tokaj	
25 30 Sárospatak	



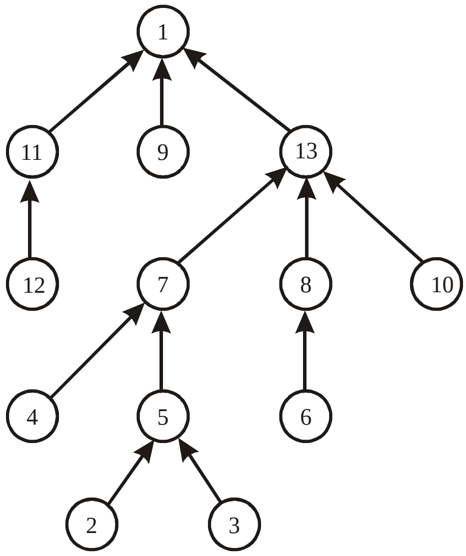
5. feladat: Jelentés (14 pont)

Egy nagyvállalatnál az alkalmazottak olyan szervezeti felépítésben dolgoznak, hogy mindenkinek (kivéve a vállalat igazgatóját, akinek nincs főnöke) pontosan egy közvetlen főnöke van. Ha valaki jelentést akar továbbítani az igazgatónak, akkor azt csak úgy teheti, hogy továbbítja a közvetlen főnökének, aki azt az Ő közvetlen főnökének küldi, és így tovább, amíg a jelentés el nem jut az igazgatóhoz. Minden jelentés továbbítása a közvetlen főnöknek egy órát vesz igénybe.

Készíts programot, amely kiszámítja, hogy kik azok az alkalmazottak, akiknek a jelentése pontosan a megadott óra alatt jut el az igazgatóhoz!

A *JELENT.BE* állomány első sorában a vállalat alkalmazottainak száma $(1 \le N \le 200)$ és a kérdésben szereplő időtartam $(1 \le K \le 200)$ van. A második sor pontosan N számot tartalmaz, az i-edik szám az i-edik alkalmazott közvetlen főnökének a sorszáma. Az igazgatót az 1 sorszám azonosítja, és így a második sorban az első szám 0, jelezve, hogy az igazgatónak nincs főnöke.

A *JELENT.KI* állomány első sorába azoknak az alkalmazottaknak az M számát kell írni, akiknek a jelentése pontosan K óra alatt jut el az igazgatóhoz! A második sor tartalmazza ezeknek az alkalmazottaknak a sorszámát egy-egy szóközzel elválasztva, növekvő sorrendben!



Példa:

```
JELENT.BE
13 2
0 5 5 7 7 8 13 13 1 13 1 11 1
JELENT.KI
4
7 8 10 12
```

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Jegyek (14 pont)

Egy osztályban N tanuló van. Dolgozatíráskor a tanulókat 0 és M pont között pontozzák. Lelkileg nem szerencsés, amikor valaki úgy érzi, hogy nagyon közel volt egy osztályzathoz, csak néhány ponton múlt, hogy nem érte azt el. Emiatt azt találták ki, hogy úgy határoznak meg ponthatárokat az egyes jegyekhez, hogy a ponthatáron ne legyen egyetlen diák sem, valamint a diákok minél messzebb legyenek ennek elérésétől, de azért mindenféle osztályzatot kiadhassanak.

Készíts programot, amely megadja a ponthatárokat!

A *JEGYEK.BE* állomány első sorában a tanulók száma $(1 \le N \le 50)$ és a maximálisan elérhető M pontszám $(2 * N \le M \le 1000)$ van. A következő N sorban egy-egy tanuló pontszáma található.

A *JEGYEK.KI* állomány öt sorába az egyes osztályzatok pontszámhatárát kell írni, az első sorba az egyes felső határát (az alsó határ 0 pont), a másodikba a kettesét, és így tovább! (Az 5-ös felső határa biztosan M pont.) Ha több megoldás is lenne, elég az egyiket kiírni. Ha a fenti feltételekkel nincs megoldás – a pontok alapján nem lehet ötféle osztályzatot adni (pl. csak 1 tanuló írt dolgozatot), akkor a JEGYEK.KI állományba egyetlen, 0-t tartalmazó sort kell kiírni!

Példa: (a példa ellenőrzése miatt a bemenetben a pontszámok növekvő sorrendben vannak, a verseny tesztelésekor azonban tetszőleges sorrendben lehetnek!!!)

JEGYEK.BE	JEGYEK.KI
10 100	15
8	36
9	47
16	79
18	100
23	
37	
48	
54	
60	
80	

2. feladat: Szövegkereső (16 pont)

Szövegek keresésekor megengedhető, hogy a keresőszóba a normál karakterek mellett ?, valamint * karaktert is tegyünk. A ? azt jelenti, hogy azon a pozíción tetszőleges karakter állhat, a * pedig tetszőleges számú karaktert helyettesíthet.

Készíts programot, amely kettő, legfeljebb egy * karaktert tartalmazó keresőszóról eldönti, hogy van-e olyan szó, amelyet mindkettővel megtalálhatunk, és ha van, akkor meg is ad egy ilyet!

A *KERES.BE* állomány első sorában az első, a másodikban pedig a második keresőszó van.

A *KERES.KI* állomány első sorába egy olyan, legfeljebb 200 karaktert tartalmazó szót kell írni, ami mindkét keresőszónak megfelel! Ha nincs ilyen szó, akkor a NEM szót kell kiírni!

Példa:

KERES.BE	KERES.KI
Alm*a	Almafa
A?ma*fa	

3. feladat: Rendezvény (15 pont)

Egy kultúrháznak két nagy előadóterme van, A és B. Egy napon sok előadást szeretnének tartani a két teremben. Az igazgató begyűjtötte az igényeket, azt, hogy ki mettől-meddig akar előadást tartani. Természetesen egy teremben egyszerre csak egy előadás tartható. Ha egy előadás az T időpontban ér véget, akkor a következő előadás legkorábban a T+1 időpontban kezdődhet.

Készíts programot, amely kiszámítja a legtöbb előadás számát, amelyek megtarthatók a két teremben! A programod adjon is meg egy beosztást a két teremre, amellyel elérhető, hogy a lehető legtöbb előadás legyen megtartva!

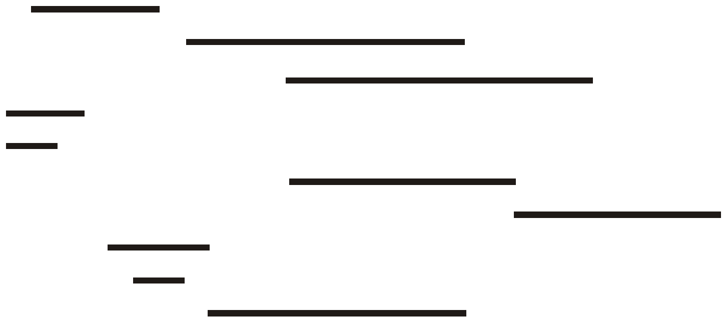
A *RENDEZ.BE* állomány első sorában az igényelt előadások száma van ($1 \le N \le 1000$). A további N sor mindegyike egy igényelt előadás kezdő időpontját és befejezési időpontját ($1 \le K < B \le 720$) tartalmazza.

A *RENDEZ.KI* állomány első sora első száma az A terembe beosztott előadások U száma, a második pedig a B terembe beosztott előadások V száma legyen! A második sor azon előadások sorszámát tartalmazza, amelyeket ebben a sorrendben az A teremben tartanak! A harmadik sor azon előadások sorszámát tartalmazza, amelyeket ebben a sorrendben az B teremben tartanak! Több megoldás esetén bármelyik megadható.

Példa:

RENDEZ.BE
```
10
3 8
9 20
13 25
2 5
2 4
13 22
22 30
6 10
7 9
10 20
```

RENDEZ.KI
```
3 4
5 8 6
4 9 10 7
```



4. feladat: Hálózat (15 pont)

Egy számítógépes hálózat csomópontokból és bizonyos csomópont-párokat összekötő egyirányú adatátvitelt biztosító közvetlen vonalakból épül fel. Adott A csomópontból egy másik B csomópontba lehet adatot továbbítani, ha van olyan $A=p_1, p_2, \ldots, p_k=B$ csomópont-sorozat, hogy minden i-re (i=1,…,k-1) p_i –ből van közvetlen vonal p_{i+1} –be.

Készíts programot, amely kiszámítja, hogy melyek azok a Q csomópontok, amelyekbe lehet adatot továbbítani adott K csomópontból, de Q-ból nem lehet adatot továbbítani K-ba!

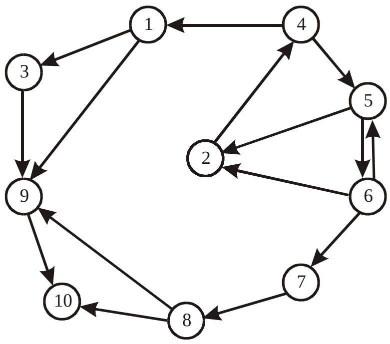
A *HALOZAT.BE* állomány első sorában a csomópontok száma (2≤N≤150), a közvetlen vonalak száma és a kijelölt K csomópont van. A további M sor mindegyike egy számpárt tartalmaz (1≤U,V≤N), ami azt jelenti, hogy az U csomópontból közvetlen vonalon lehet adatot továbbítani a V csomópontba.

A *HALOZAT.KI* állomány első sorába azon Q csomópontok számát kell írni, amelyekbe lehet adatot továbbítani a K csomópontból, de Q-ból nem lehet adatot továbbítani K-ba! A második sor tartalmazza ezeket a csomópontokat tetszőleges sorrendben!

Példa:

HALOZAT.BE
```
10 15 5
4 5
2 4
4 1
5 2
5 6
6 5
6 2
6 7
1 3
3 9
1 9
7 8
8 9
9 10
8 10
```

HALOZAT.KI
```
6
1 7 3 8 9 10
```



5. feladat: Játék (15 pont)

Tekintsük azt az egyszemélyes játékot, amelyet egy N sorból és M oszlopból álló négyzetrácsos táblán lehet játszani! A tábla véletlenszerűen kiválasztott mezőin gyöngyöket helyeznek el. A táblán lehetnek csapda mezők, amelyekre nem lehet lépni. A játék célja az, hogy a játékos egy bábut mozgatva a tábla mezőin a lehető legtöbb gyöngyöt gyűjtse be. A játékszabály a következő:

- Kezdetben a bábu a tábla (1,1) koordinátájú bal felső sarkában áll.
- Egy lépésben a bábut csak szomszédos mezőre lehet mozgatni, vagy jobbra, vagy lefelé.
- Csapda mezőre nem lehet lépni.
- A játék akkor ér véget, ha a bábu a tábla (`N,M`) koordinátájú jobb alsó mezőjére, a célmezőre kerül.
- A játékban szerzett pontszám azokon a mezőkön található gyöngyök számának összege, amelyekre a bábuval lépett a versenyző. Az (1, 1) nem csapdamező és az ott lévő gyöngyök is a játékosé lesznek.

Készíts programot, amely kiszámít egy olyan játékmenetet, amely a legtöbb pontot eredményezi!

A `JATEK.BE` állomány első sora a tábla sorainak és oszlopainak számát tartalmazza `(1≤N, M≤150)`. Az állomány következő `N` sora a kezdeti játékállást tartalmazza. Minden sorban pontosan `M` pozitív egész szám van. Ha `j`-edik szám `-1`, akkor ott csapda mező van, egyébként azt adja meg, hogy az adott sorban a `j`-edik mezőn hány gyöngy van. Minden szám értéke legfeljebb `500`.

A `JATEK.KI` állomány első sorába a szabályos játékkal elérhető legnagyobb pontszám értékét kell írni! Ha a célmező nem érhető el, akkor az első és egyetlen sorba a `-1` értéket kell írni! Ha el lehet jutni a célmezőre, akkor a második sor pontosan `N+M-2` karaktert tartalmazzon, egy olyan szabályos lépéssorozatot, amellyel elérhető a maximális pontszám! A jobbra lépés jele a `'J'`, a lefelé lépés jele az `'L'` karakter. A karakterek között nem lehet szóköz, és az utolsó karakter után nem lehet szóköz! Több megoldás esetén bármelyik megadható.

Példa: (a bemeneti állományban vastagon szedve jelöljük a megoldás szerinti utat)

JATEK.BE	JATEK.KI
5 6	17
1 2 3 4 0 1	JJLLLLJJJ
2 -1 **2** 1 -1 3	
-1 0 **6** 0 0 0	
4 1 **0** -1 1 -1	
0 0 **1 2 0 0**	

2007. Harmadik forduló

Ötödik-nyolcadik osztályosok

1. feladat: Négyzetek (18 pont)

Van N darab egységnyi méretű négyzetlapunk. KxK-as négyzeteket kell összerakni belőlük, először a lehető legnagyobbat, utána a maradékból egyre kisebbeket,...

Írj programot, amely beolvassa a négyzetlapok számát (`1≤N≤10 000`), majd megadja, hogy a fenti elven mekkora négyzetek rakhatók ki belőlük!

Példa:

```
N=72        ⇒8*8-as négyzet, 2*2-es négyzet, 2*2-es négyzet
Magyarázat: 72=64+4+4.
```

2. feladat: Bűvös négyzet (30 pont)

A bűvös négyzetek N*N számot tartalmaznak, négyzetes elrendezésben. Minden bűvös négyzetre igaz, hogy minden sor és minden oszlop összege is ugyanaz a szám. Kaptunk egy bűvös négyzetet, amiben lehet, hogy egyetlen számot elrontottak.

Írj programot, amely beolvassa a bűvös négyzet méretét ($3 \le N \le 5$) majd a bűvös négyzet számait, majd megadja, hogy a bűvös négyzet helyesen van-e kitöltve! Ha nem helyes, akkor azt is meg kell adnia, hogy hol van a hiba és milyen számot kellene oda írni!

Példa:

```
N=3          ⇒    Hibás a bűvös négyzet
2 3 2             Hibás hely: 2. sor 1. oszlop
3 1 4             Helyes értéke: 2
3 3 1
```

3. feladat: Nyelvek (27 pont)

Egy programozási versenyen minden versenyző választhat egy programozási nyelvet, amin dolgozni fog.

Készíts programot a következő feladat megoldására! A programod olvassa be a választható nyelvek számát ($1 \le M \le 10$) és a versenyen induló tanulók számát ($1 \le N \le 100$), majd a választható nyelveket, s legvégül az egyes tanulók által választott nyelveket! Ezután adja meg, hogy mely tanulók választottak illegális nyelvet (olyat, ami nem szerepelt a felsoroltak között), mely nyelveket nem választotta senki, s melyik választott nyelvet hányan választották!

Példa:

```
Nyelvek száma: 3       ⇒  Illegális nyelv: 3. versenyző
Versenyzők száma: 5       Nem választott nyelv: Logo
Választható nyelvek:      Választott nyelvek:
    Pascal                        Pascal: 3 versenyző
    Logo                          C++: 1 versenyző
    C++
Választott nyelvek:
    Pascal
    Pascal
    Delphi
    C++
    Pascal
```

Kilencedik-tizedik osztályosok

1. feladat: Madarak (20 pont)

Egyes madarak a fészkelő helyüktől adott távolságra saját területet, ún. territóriumot tartanak. Ha két madár territóriuma átfedő, akkor ott összeverekedhetnek egymással.

Készíts programot, amely megadja

A. a senki mással nem verekedő madarakat;

B. azt a legveszélyesebb helyet, ahol a legtöbb verekedés lehet és ezen a helyen a verekedésben résztvevő madarak számát;

C. azon helyek számát, amely nem tartozik egyetlen madár territóriumához sem!

A MADARAK.BE állomány első sorában a madarak száma ($1 \le M \le 32$) és a négyzet alakú terület mérete ($1 \le N \le 100$) van. A következő M sorban egy-egy madarat leíró 3 szám szerepel. Az első két szám a madár fészkének helye ($1 \le X, Y \le N$), a harmadik pedig a territórium mérete ($1 \le R \le 10$), ami azt jelenti, hogy a territórium az ($X-R, Y-R$) ponttól az ($X+R, Y+R$) pontig tart (az N*N-es területről kilógó részekkel nem kell foglalkozni).

A MADARAK.KI állomány három sorába a három részfeladat megoldását kell írni. Ha valamelyik részfeladatra nincs megoldás, egy üres sort akkor is ki kell írni! Az első sorba a senki mással nem verekedő madarak sorszáma kerüljön! A második sorba a legveszélyesebb hely verekedésszámát és a sor- és oszlopindexét kell írni (ha több megoldás van, akkor bármelyik kiírható)! A harmadik sorba a territóriumhoz nem tartozó helyek száma kerüljön!

Példa: (az ábrán látható, hogy melyik hely melyik madár territóriumához tartozik)

			5	5	5		4	4	4
	1	1	15	5	5		4	4	4
	1	13	135	35	35	3			
	1	13	13	3	3	3			
2	2	23	23	23	23	23	2		
2	2	23	23	23	23	23	2		
2	2	23	23	23	23	23	2		
2	2	2	2	2	2	2	2		
2	2	2	2	2	2	2	2		
2	2	2	2	2	2	2	2		

MADARAK.BE	MADARAK.KI
5 10	4
3 3 1	3 3 4
10 3 5	26
5 5 2	
1 9 1	
2 5 1	

2. feladat: Vízirendőr (15 pont)

Egy folyó mentén N település helyezkedik el, melyek közül néhányon vízirendőrséget szeretnének alapítani. Egy rendőrcsónak a folyó folyásirányában A, folyásiránnyal szemben pedig B kilométert tud megtenni óránként.

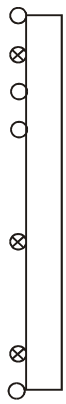
Készíts programot, amely megadja azt a minimális számú települést, ahol vízirendőrséget kell létrehozni úgy, hogy a rendőrségekről bármely település K órán belül elérhető legyen!

A VIZI.BE állomány első sorában a települések száma (1≤N≤1000), a felfelé és a lefelé 1 óra alatt megtehető távolság (1≤A,B≤100), valamint az időtartam (1≤K≤10) van. A következő N-1 sorban folyásirányban haladva az adott települések távolsága szerepel az első településtől mérve.

A VIZI.KI állomány első sorába az alapítandó vízirendőrségek M számát kell írni, a második sorba pedig az M vízirendőrség helyének sorszámát! Több megoldás esetén bármelyik megadható.

Példa:

VIZI.BE	VIZI.KI
7 20 10 1	3
10	2 5 7
20	
30	
60	
90	
100	



3. feladat: Jegy (20 pont)

Egy jegyiroda nagyszabású koncertre árul jegyeket. Összesen *M* ülőhelyre lehet jegyeket igényelni, pontosabban minden igénylő két egymás melletti jegyet igényelhet, és meg kell adnia, hogy ezért mennyit fizetne. A jegyiroda a beérkezett igények közül ki akarja választani a legtöbb bevételt eredményező igényeket, amelyeket természetesen ki is tud elégíteni.

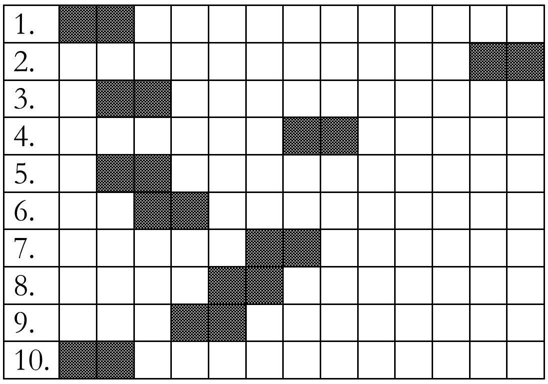
Készíts programot, amely kiszámítja az elérhető legnagyobb jegybevételt és meg is adja, hogy melyik igények kielégítése esetén érhető el a legnagyobb bevétel!

A JEGY.BE állomány első sora az ülőhelyek számát (1≤M≤6000) és az igények számát tartalmazza (1≤N≤30 000). A további *N* sor mindegyike egy igényt leíró két egész számot tartalmaz. Az első szám az igényelt két ülőhely első székének *s* sorszáma (1≤s<M), a második szám az az *f* pénzösszeg (1≤f<500), amit az igénylő a két jegyért fizetne.

A JEGY.KI állomány első sorába az elérhető legnagyobb bevételt kell írni! A második sorba azoknak az igényeknek a sorszáma kerüljön (tetszőleges sorrendben), amelyek esetén a bevétel a legnagyobb lesz! Több megoldás esetén bármelyik megadható.

```
JEGY.BE
20 10
1 21
12 23
2 12
7 43
2 32
3 24
6 12
5 33
4 32
1 12
```

```
JEGY.KI
144
2 4 8 6 1
```



4. feladat: Ismeretség (20 pont)

Egy Internetes fórumon a tagok megadhatják, hogy kit ismernek, az ismeretség azonban itt nem biztos, hogy kölcsönös. Az ismeretség lehet közvetlen (A ismeri Z-t), illetve közvetett (A ismeri B-t, aki ismeri C-t, … s az így felépített lánc utolsó tagja ismeri Z-t).

Készíts programot, amely egy adott tagra megadja azon közvetlen vagy közvetett ismerőseit, akikkel az ismeretség nem kölcsönös!

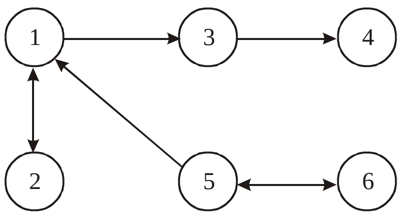
Az ISMER.BE állomány első sorában a tagok száma (1≤N≤100), az ismeretségek száma (1≤M≤10 000) és az adott tag sorszáma (1≤A≤N) van. A következő M sor mindegyike két egész számot tartalmaz (1≤X≠Y≤N), aminek jelentése: X ismeri Y-t.

Az ISMER.KI állomány első sorába A olyan közvetlen ismerősei K számát kell írni, akik nem kölcsönös ismerősök! A második sorban a K ilyen ismerős sorszáma legyen! Az állomány harmadik sorába A olyan közvetett ismerősei L számát kell írni, akik nem kölcsönös ismerősök! A negyedik sorban az L ilyen ismerős sorszáma legyen!

Példa:

```
ISMER.BE
6 7 1
1 2
2 1
1 3
3 4
5 6
6 5
5 1
```

```
ISMER.KI
1
3
1
4
```



Tizenegyedik-tizenharmadik osztályosok

1. feladat: Madár (15 pont)

Egyes madarak a fészkelő helyüktől adott távolságra saját területet, ún. territóriumot tartanak. Ha két madár territóriuma átfedő, akkor ott összeverekedhetnek egymással.

Készíts programot, amely megadja

A. a senki mással nem verekedő madarakat;

B. a legtöbb másik madárral verekedő madarat;

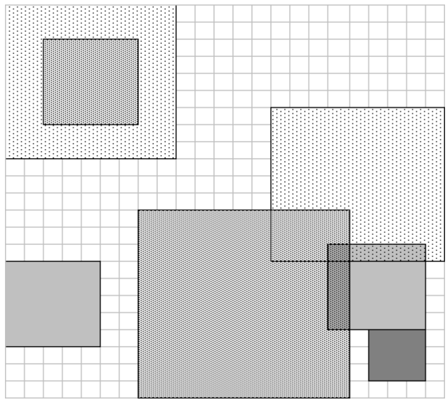
C. azon madarak számát, amelyek territóriuma része valamely más madár territóriumának!

A MADAR.BE állomány első sorában a madarak száma (`1≤M≤1000`) és egy, a fészkeket tartalmazó négyzet alakú terület mérete (`1≤N≤10 000`) van. A következő M sorban egy-egy madarat leíró 3 szám szerepel. Az első két szám a madár fészkének helye (`1≤X,Y≤N`), a harmadik pedig a territórium mérete (`1≤R≤100`), ami azt jelenti, hogy a territórium az (`X-R,Y-R`) ponttól az (`X+R,Y+R`) pontig tart.

A MADAR.KI állomány három sorába a három részfeladat megoldását kell írni. Ha valamelyik részfeladatra nincs megoldás, egy üres sort akkor is ki kell írni! Az első a senki mással nem verekedő madarak sorszáma kerüljön! A második sorba a legtöbb másik madárral verekedő madár sorszámát kell írni (ha több megoldás van, akkor bármelyik kiírható)! A harmadik sorba azon madarak száma kerüljön, amelyek territóriuma teljes egészében része valamely más madár territóriumának!

<u>Példa:</u>

MADAR.BE	MADAR.KI
7 100	6 7
7 7 4	3
7 7 2	1
20 15 5	
14 21 4	
19 22 2	
20 5 2	
23 23 1	



<u>2. feladat</u>: Málna (15 pont)

Egy gyümölcslevet gyártó üzem naponta K kg málnát tud feldolgozni. N termelő ajánlotta fel a málnáját, az is lehet, hogy eltérő árakon. Mivel a málna romlandó, ezért az árusok naponta egységesen F forintot engednek az árból. Az üzem a lehető leghamarabb fel akarja dolgozni a beérkezett málnát, de ezen belül azért arra törekszik, hogy a lehető legkevesebb pénzt kelljen fizetnie a termelőknek. Feltehető, hogy az ár az engedményekkel sem megy le 0 forintra.

Készíts programot, amely megadja, hogy az üzem melyik napon melyik termelőtől vegyen málnát, hogy a kiadása a lehető legkisebb legyen!

A MALNA.BE állomány első sorában az üzem napi kapacitása (`100≤K≤1000`), a termelők száma (`1≤N≤5000`), valamint a napi engedmény (`1≤F≤100`) szerepel. A következő N sor mindegyikében két egész szám van, az adott termelő által felajánlott málna mennyisége (`1≤mennyiség≤200`), illetve kilónkénti ára (`500≤ár≤1500`).

A MALNA.KI állomány első sorába az összes málna feldolgozásához szükséges napok M számát és az üzem által kifizetett teljes vételárat kell írni! A következő M sor mindegyikében egy-egy nap adatai szerepeljenek: azon termelők sorszáma és a vásárolt mennyisége, akiktől azon a napon vesz málnát az üzem!

<u>Példa:</u>

MALNA.BE	MALNA.KI
100 5 100	3 144000
40 800	3 20 4 70 2 10
120 600	2 100
20 500	2 10 5 30 1 40
70 500	
30 700	

<u>3. feladat</u>: DNS (15 pont)

A biológiai szekvenciák, különösen a DNS szekvenciák vizsgálata nagyon fontos kutatási terület. Minden DNS szekvencia leírható olyan karaktersorozattal, amely csak az A, C, G és T karaktereket tartalmazhatja. Két DNS szekvencia hasonlóságára különböző mértékeket használnak. Az egyik leggyakrabban alkalmazott mérték a következőt jelenti. Adott S1 és S2 szekvenciához keresnek olyan S szekvenciát, hogy mind S1, mind S2 előállítható S-ből karakterek beszúrásával, illetve átírásával. Mivel biológiailag nagyobb hasonlóságot jelent egy karakter átírása, mint egy beszúrás, ezért az átírást 1, a beszúrást 2 súllyal számítják. Tehát a hasonlóság vizsgálatánál olyan S szekvenciát keresnek, amelyből a lehető legkisebb összsúllyal előállítható S1 és S2. Ezt az értéket a két szekvencia hasonlósági értékének nevezik. Pl. az S1=ATGCGTTT és az S2=ATCCGCGTC esetén az S=ATCCGGTC szekvenciából S1 3 átírással, az S2 pedig egy beszúrással kapható, tehát a hasonlósági érték 5, mert nincs ennél jobb előállítás.

Készíts programot, amely kiszámítja két DNS szekvencia hasonlósági értékét, és meg is ad egy optimális előállítást!

A DNS.BE állomány első sorában az S1, a második sorában az S2 DNS szekvencia van. Mindkettő legfeljebb 100 karaktert tartalmaz.

A DNS.KI állomány első sorába azt a K egész számot kell írni, ami az S1 és S2 hasonlósági értéke! A második sor tartalmazza azt az S DNS szekvenciát, amelyből S1 és S2 előállítható pontosan K összsúlyú módosítással! A harmadik sorba egy olyan karaktersorozatot kell írni, amely azt adja meg, hogy az S1 hogyan állítható elő S-ből, a negyedikbe pedig olyat, amely az S2 előállítását adja! Az előállítások leírásában a '_' aláhúzás jel jelölje a beszúrást, az 'X' karakter pedig az átírást! Több megoldás esetén bármelyik megadható.

Példa:

DNS.BE	DNS.KI
ATGCGTTT	5
ATCCGCGTC	ATCCGGTC
	ATXCGXTX
	ATCCG_GTC

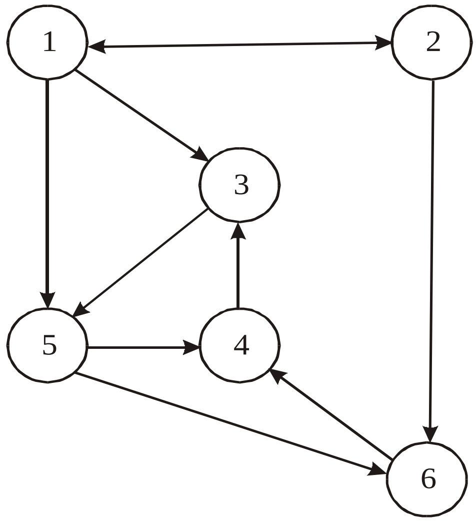
4. feladat: Vidámpark (15 pont)

Egy vidámpark *N* részlegből áll, a részlegeket az *1,…,N* számokkal azonosítják, az *1.* részleg a bejárat, az *N.* részleg a kijárat. Minden részlegbe ugyanazzal az egységes jeggyel lehet bemenni. A részlegeket egyirányú utak kötik össze. Ha egy részlegben elhasználunk egy jegyet, akkor el kell hagyni azt a részleget, tehát át kell menni egy másik részlegbe. Ha K darab jegyünk van, és mindet el akarjuk használni, akkor meg kell határoznunk egy olyan $R_1=1, \ldots, R_k=N$ sétát, amely a részlegeknek egy olyan felsorolása, amely a bejárattal kezdődik, a kijárattal végződik, és minden i-re ($1 \le i < K$) az R_i részlegből közvetlenül át lehet menni az R_{i+1} részlegbe. (Ugyanazon részleg természetesen többször is előfordulhat a sétában.)

Készíts programot, amely meghatároz egy olyan pontosan K részleget tartalmazó sétát, amely a bejárattal kezdődik, a kijárattal végződik!

A VIDAM.BE állomány első sora a részlegek számát ($2 \le N \le 100$), a a részlegeket összekötő közvetlen utak számát ($1 \le M \le 5000$), valamint a jegyek számát ($2 \le K \le 200$) tartalmazza. A további *M* sor mindegyike két egész számot tartalmaz ($1 \le U, V \le N,\ U \ne V$), ami azt jelenti, hogy az U részlegből van közvetlen út a V részlegbe.

A VIDAM.KI állomány egyetlen sort tartalmazzon! A sor az egyetlen 0 számot tartalmazza, ha nincs megoldás! Ha van megoldás, akkor pontosan K egész számot tartalmazzon, ami egy K hosszúságú séta a bejárattól a kijáratig!



Példa:

VIDAM.BE	VIDAM.KI
6 10 7	1 5 6 4 3 5 6
1 2	
2 1	
2 6	
1 5	
1 3	
3 5	
5 4	
5 6	
4 3	
6 4	

5. feladat: Labirintus (15 pont)

Tekintsük azt a labirintust, amely egy *MxN*-es négyzetrács, amelynek minden mezője lehet:

- Üres (0)
- Fal (1)
- Kapcsoló (2)
- Ajtó, amely vagy nyitva van (3), vagy zárva van (4)

Ha kapcsoló mezőre lépünk, akkor minden nyitott ajtó bezáródik, és minden zárt ajtó kinyílik. A labirintus (1,1) koordinátájú bal felső sarkából a lehető legkevesebb lépéssel el kell jutni a jobb alsó (M,N) koordinátájú mezőjére. Minden lépésben a szomszédos mezőre léphetünk balra, jobbra, lefelé vagy felfelé, feltéve, hogy az nem fal és nem zárt ajtó.

Készíts programot, amely kiszámítja, hogy legkevesebb hány lépésben lehet kijutni a labirintusból, és meg is ad egy kivezető utat!

A LABI.BE állomány első sorában a labirintus sorainak (2≤M≤100), valamint oszlopainak száma (2≤N≤100) van. A további *M* sor mindegyike *N* egész számot tartalmaz. Közülük az i-edik sor j-edik számí a labirintus (i,j) koordinátájú mezőjét adja meg, a fenti kódolás szerint.

A LABI.KI állomány első sor az egyetlen 0 számot tartalmazza, ha nem lehet kijutni a labirintusból, egyébként a kijutáshoz minimálisan szükséges lépések *K* számát! A következő sor pontosan *K* karaktert tartalmazzon (szóközök nélkül), amely a kijutást eredményező egy legrövidebb lépéssorozat! A balra lépés jele a 'B', a jobbra lépésé a 'J', a felfelé lépésé az 'F', a lefelé lépésé az 'L'. Több megoldás esetén bármelyik megadható.

Példa:

LABI.BE	LABI.KI
4 5	9
0 4 2 1 0	LJLFJJLLJ
2 2 2 3 0	
1 0 4 3 1	
0 0 1 0 0	

A verseny végeredménye:

I. korcsoport		
1.	Miglász Dániel	Bárdos László Gimnázium, Tatabánya
2.	Ács Kurucz Gábor	Révai Miklós Gimnázium, Győr
3.	Sebők Márton	Földes Ferenc Gimnázium, Miskolc
	Kovács Zsombor	Fazekas Mihály Gimnázium, Budapest
	Póta Kristóf	Dobó István Gimnázium, Eger
6.	Dankovics Attila	Veres Péter Gimnázium, Budapest
7.	Backhausz Tibor	Fazekas Mihály Gimnázium, Budapest
8.	Ali Dávid	III. Béla Gimnázium, Baja
9.	Erdős Gergely	Batthyány Lajos Gimnázium és Szakközépiskola, Nagykanizsa
	Buza Dániel István	Széchenyi István Gimnázium, Dunaújváros
	Szenczi Zoltán	Fazekas Mihály Gimnázium, Budapest
II. korcsoport		
1.	Csikota Balázs	Szent Benedek Általános Iskola és Gimnázium, Budapest
2.	Danner Gábor	Ságvári Endre Gimnázium, Szeged
3.	Danka Miklós András	Fazekas Mihály Gimnázium, Budapest
4.	Hegedűs Tamás	Földes Ferenc Gimnázium, Miskolc
5.	Mártonka Zoltán	Dráva Völgye Középiskola, Barcs
6.	Turi Zsolt	Neumann János Szakközépiskola, Budapest
7.	Deák Zsolt	Verseghy Ferenc Gimnázium, Szolnok
	Kriván Bálint	Berzsenyi Dániel Gimnázium, Budapest
9.	Éles András	Fazekas Mihály Gimnázium, Debrecen
10.	Gévay Gábor	Ságvári Endre Gimnázium, Szeged

III. korcsoport

1	Szalkai Balázs	Lovassy László Gimnázium, Veszprém
2	Vincze János	Fazekas Mihály Gimnázium, Debrecen
3	Nagy Gergely	Fazekas Mihály Gimnázium, Budapest
4	Eisenberger András	Fazekas Mihály Gimnázium, Budapest
5	Györök Péter	Táncsics Mihály Gimnázium, Kaposvár
6	Nagy Csaba	Fazekas Mihály Gimnázium, Budapest
7	Badics Alex	Eötvös József Gimnázium, Tata
8	Mészáros Balázs	Berzsenyi Dániel Gimnázium, Budapest
9	Jámbor Attila	Neumann János Középiskola és Kollégium, Eger
10	Poór Márk	Neumann János Középiskola és Kollégium, Eger
11	Varga Iván András	Bólyai János Gyak. Általános Iskola és Gimnázium, Szombathely
12	Korándi Dániel	Fazekas Mihály Gimnázium, Budapest
13	Králik Balázs	Szent Orsolya Római Katolikus Általános Iskola, Sopron
14	Csóka Győző	Fazekas Mihály Gimnázium, Debrecen
15	Korom-Vellás Judit	Radnóti Miklós Gimnázium, Szeged

2008. Első forduló

Ötödik-nyolcadik osztályosok

Számítógép nélküli feladatok

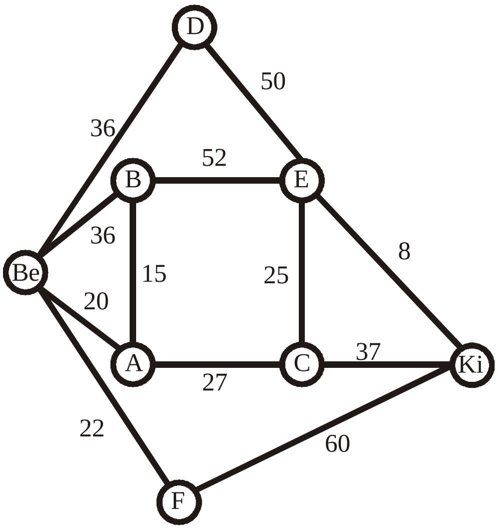
1. feladat: Képlet (20 pont)

Az alábbi algoritmus 4 számsorozat tagjait számolja ki

```
Számsorozatok:
  A:=1; B:=1; C:=1; D:=1; E:=1; F:=0
  Ciklus I=1-től 10-ig
    Ki: A,B,C,D
    A:=A+2; B:=B+A; F:=E; E:=C; C:=E+F; D:=D+I+1
  Ciklus vége
Eljárás vége.
```

Add meg az algoritmus által kiírt A, B, C, illetve D változó 10-10 értékét!

2. feladat: Park (23 pont)



A mellékelt ábrán látható parkban a bejárat (Be) és a kijárat (Ki) között utak vezetnek, amelyek különböző szobrokat (A..F) érintenek. A bejáratnál és a kijáratnál is van egy-egy szobor. Ismerjük minden út hosszát (az ábrán az utak mellé írva).

A. Add meg, hogy mennyi a bejárattól az egyes szobrokhoz vezető legrövidebb út hossza!

B. Add meg, hogy a bejárattól a kijáratig vezető legrövidebb út mely szobrokat érint!

C. Add meg a legtöbb szobrot érintő utat, amely a bejárattól a kijáratig vezet, de minden úton és minden szobornál csak egyszer járhatsz!

3. feladat: Kétirányból (17 pont)

Az alábbi algoritmusnak az N elemű X számsorozatban meg kellene keresni az első (A), az utolsó (B) és a középső (C) 0 érték helyét. Ha páros számú 0 van a sorozatban, akkor a két középső közül a kisebb sorszámot kell adni (pl. négy 0 esetén a másodikat tekintjük középsőnek). Ha a számsorozatban nincs 0, akkor mindhárom változónak 0-nak kellene lennie.

Az algoritmus azonban sajnos hibás, add meg, hogy mik a hibák benne! (Az algoritmus szerkezete jó, csupán figyelmetlenségből eredő elírások és kihagyások vannak benne.)

```
...
E:=1; U:=N; A:=0; B:=0; C:=0
Ciklus amíg E<U
  Ciklus amíg E≤U és X(E)≠0
    E:=E+1
  Ciklus vége
  Ha A=0 akkor A:=E

  Ha E≤U akkor E:=C
  Ciklus amíg E≤U és X(U)=0
    U:=U+1
  Ciklus vége
  Ha E≤U akkor B:=U
Ciklus vége
...
```

Számítógépes feladatok

1. feladat: Távolság (40 pont)

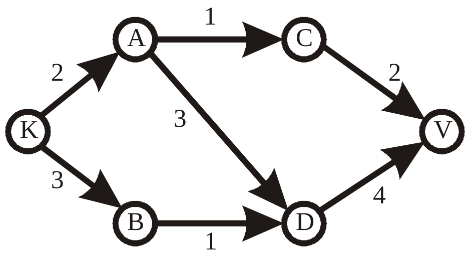
Készíts programot, amely beolvassa egy ember nevét csupa nagybetűvel, majd kiírja a nevet és azt, hogy milyen messze van egymástól a két, egymástól legmesszebb levő E-betű! Ha csak egy E-betű van a névben, akkor az eredmény legyen 0, ha pedig egy sincs, akkor pedig -1! A név legfeljebb 100 karakteres lehet.

Példa:

```
név: FERENC    ⇒  távolság: 2
```

Kilencedik-tizedik osztályosok

1. feladat: Ütemezés (17 pont)



A mellékelt ábra egy feladat részfeladatainak ütemezését mutatja. Az ábrán K betű jelöli a megoldás kezdetét, V pedig a végét. Az egyes részfeladatokat az ábécé betűivel jelöljük. Megadjuk minden részfeladatra, hogy valamely előző részfeladat után mennyi idővel lehet elvégezni (például az ábrán az A részfeladat a K kezdés után 2 időegységgel kezdhető, a C pedig az A elkezdése után 1 időegységgel). Egy részfeladat akkor kezdhető el, ha minden előzményét elvégeztük. Add meg a mellékelt ábra alapján, hogy

A. mikor kezdhető el az egyes részfeladatok legkorábban (a K a 0. időpontban kezdődik)?

B. melyek azok a részfeladatok, amelyek később kezdése nem befolyásolja a V részfeladat lehető legkorábbi elkezdését, és mennyivel lehet őket később kezdeni?

2. feladat: Gyorsabbra (21 pont)

Az alábbi algoritmus kiszámolja N tanuló érettségi eredménye százaléka (biztosan 0 és 100 közötti egész számok) alapján, hogy melyik osztályzatból mennyi lesz, valamint megadja a sikeres érettségit tett és a bukott tanulók számát.

Írd át gyorsabbra az alábbi algoritmust!

```
Valami:
  I:=0; II:=0; III:=0; IV:=0; V:=0; SIK:=0; BUK:=0
  Ciklus J=1-től N-ig
    Ha 0≤X(J) és X(J)≤20 akkor I:=I+1
    Ha 21≤X(J) és X(J)≤40 akkor II:=II+1
    Ha 41≤X(J) és X(J)≤60 akkor III:=III+1
    Ha 61≤X(J) és X(J)≤80 akkor IV:=IV+1
    Ha 81≤X(J) és X(J)≤100 akkor V:=V+1
    Ha 21≤X(J) és X(J)≤100 akkor SIK:=SIK+1
    Ha 0≤X(J) és X(J)≤20 akkor BUK:=BUK+1
  Ciklus vége
Eljárás vége.
```

3. feladat: Mik a hibák? (21 pont)

Multihalmaznak nevezzük az olyan halmazokat, amelyeknél az egyes elemek több példányban is előfordulhatnak. Például a (ló,egér,kutya,egér,egér,ló) multihalmazt a következőképpen írhatjuk le: ((ló,2), (egér,3), (kutya,1)). Egy másik multihalmaz: ((ló,3), (macska,2), (egér,1)). Multihalmazok összege az unióhoz hasonló, de a darabszámok összegét vesszük (a fenti két halmazra: ((ló,5), (egér,4), (kutya,1), (macska,2))). Javítsd ki az alábbi algoritmust, ami a két multihalmaz (az N állatfajtát tartalmazó X és az M állatfajtát tartalmazó Y) összegét adná meg, ha jó lenne!

```
Összeg(N,X,M,Y,K,Z):
  K:=N; Z:=X
  Ciklus I=1-től M-ig
    J:=1
    Ciklus amíg J≤K és Y(J).név≠X(I).név
      J:=J+1
    Ciklus vége
    Ha J>N akkor Z(J).db:=Z(J).DB+Y(I).db
        különben K:=N+1; Z(K):=X(I)
    Elágazás vége
  Ciklus vége
Eljárás vége.
```

4. feladat: Pakolás (17 pont)

Egymás mellé N darab ládát helyezünk el, különböző méretűeket. Tetszőleges üres láda megfogható és belehelyezhető valamely, tőle balra vagy jobbra elhelyezkedő nagyobb ládába, ha köztük nincs másik láda. Ha például balról jobbra haladva a ládák mérete rendre 1, 3, 2, akkor először a 2 méretűt rakjuk a 3 méretűbe, majd az 1 méretűt beletehetjük; de ha először az 1-est tesszük a 3-asba, akkor a 2-est már nem tehetjük bele. Az a cél, hogy a pakolás végén a ládákat a lehető legkevesebb ládába pakoljuk be.

Példa:

1 3 2 4 5 5 ⇒ 3 (azaz például az 5-ösbe tesszük a mellette levő 4-est, azután ebbe beletesszük a most már szomszédos 2-est, majd a 3-asba tesszük az 1-est, a másik 5-ös üresen marad)

Add meg, hogy az alábbi ládasorozatok esetén a ládák hány ládába pakolhatók!

A. 1 2 3 4 5 5 4 3 2 1

B. 1 3 5 2 4 2 3 5 4 1 2 4

C. 2 4 3 5 3 5 3 2 2 3 4 5 4 5

D. 3 2 5 1 4 3 4 2 5

5. feladat: Kifejezések (24 pont)

A matematikában megismert kifejezéseket viszonylag nehéz programmal kiszámítani a zárójelezés és a bonyolult kiszámítási szabályok miatt, ezért az informatikusok kitalálták a kifejezések ún. lengyel formáját. Ebben a formában a kifejezésben szereplő műveleti jeleket nem a paramétereik közé, hanem mögé kell írni. Például: X+Y*Z helyett X Y Z * +, (X+Y)*Z helyett X Y + Z * szerepel. Mint a példából látható, a kifejezés lengyel formájában nincs szükség zárójelek használatára.

Add meg az alábbi kifejezések lengyel formáját:

A. A*B+C*D+E/F

B. (X–Y)*(X+Z)

C. X+Y+Z*(A–B–C)

Add meg az alábbi lengyel formájú kifejezések hagyományos formáját:

D. X Y + A B + *

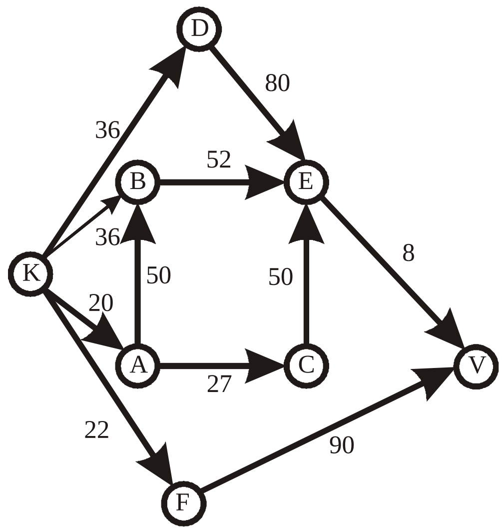
E. A B – C * C D + B * +

F. A B + C * A B + + A B + /

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Ütemezés (20 pont)

A mellékelt ábra egy feladat részfeladatainak ütemezését mutatja. Az ábrán K betű jelöli a megoldás kezdetét, V pedig a végét. Az egyes részfeladatokat az ábécé betűivel jelöljük. Megadjuk minden részfeladatra, hogy valamely előző részfeladat után mennyi idővel lehet elvégezni (például az ábrán az A részfeladat a K kezdés után 20 időegységgel kezdhető, a C pedig az A elkezdése után 27 időegységgel). Egy részfeladat akkor kezdhető el, ha minden előzményét elvégeztük. Add meg a mellékelt ábra alapján, hogy



A. mikor kezdhetők el az egyes részfeladatok legkorábban (a K a 0. időpontban kezdődik)?

B. melyek azok a részfeladatok, amelyek később kezdése nem befolyásolja a V részfeladat lehető legkorábbi elkezdését, és mennyivel lehet őket később kezdeni?

2. feladat: Gyorsabbra (18 pont)

Az alábbi algoritmus kiszámolja N tanuló osztálysorszáma (biztosan 5 és 12 közötti egész számok) alapján, hogy közülük hányan indulhatnak a Nemes Tihamér verseny 1. (I) és 2. (II) korcsoportjában, az OKTV-n (OK), a Közép-Európai Diákolimpián (CE), a Nemzetközi Diákolimpián (IO), illetve hányan érettségizhetnek (ER) normál időben informatikából.

Írd át gyorsabbra az alábbi algoritmust!

```
Valami:
  I:=0; II:=0; OK:=0; CE:=0; IO:=0; ER:=0
  Ciklus J=1-től N-ig
    Ha 5≤X(J) és X(J)≤8 akkor I:=I+1
    Ha X(J)=9 vagy X(J)=10 akkor II:=II+1
    Ha 11≤X(J) és X(J)≤12 akkor OK:=OK+1
    Ha X(J)=9 vagy X(J)=10 vagy X(J)=11 akkor CE:=CE+1
    Ha 11≤X(J) és X(J)≤12 akkor IO:=IO+1
    Ha X(J)=12 akkor ER:=ER+1
  Ciklus vége
Eljárás vége.
```

3. feladat: Mik a hibák? (21 pont)

Multihalmaznak nevezzük az olyan halmazokat, amelyeknél az egyes elemek több példányban is előfordulhatnak. Például a (ló,egér,kutya,egér,egér,ló) multihalmazt a következőképpen írhatjuk le: ((ló,2), (egér,3), (kutya,1)). Egy másik multihalmaz: ((ló,3), (macska,2), (egér,1)). Multihalmazok uniójakor az azonos elemeknél a darabszámok nagyobbikát választjuk (a fenti két halmazra: ((ló,3), (egér,3), (kutya,1), (macska,2))).

Add meg, hogy mik a hibák az alábbi algoritmusban, ami a két multihalmaz (az N állatfajtát tartalmazó X és az M állatfajtát tartalmazó Y) unióját adná meg, ha jó lenne!

```
Unió(N,X,M,Y,K,Z):
  K:=N; Z:=X
  Ciklus I=1-től M-ig
    J:=1
    Ciklus amíg J≤N és Y(J).név=X(I).név
      J:=J+1
    Ciklus vége
    Ha J≤K akkor Ha Y(I).db>X(J).db akkor Z(J).db:=Y(I).db
                             különben K:=K+1; X(K):=Y(I)
    Elágazások vége
  Ciklus vége
Eljárás vége.
```

4. feladat: Pakolás (17 pont)

Egymás mellé N darab ládát helyezünk el, különböző méretűeket. Tetszőleges üres láda megfogható és belehelyezhető valamely, tőle jobbra akárhol elhelyezkedő nagyobb ládába. Ha például balról jobbra haladva a ládák mérete rendre 1, 2, 3, akkor először a 2 méretűt rakjuk a 3 méretűbe, majd az 1 méretűt beletehetjük; de ha először az 1-est tesszük a 2-esbe, akkor azok együtt már nem mozgathatók. Az a cél, hogy a pakolás végén a ládákat a lehető legkevesebb ládába pakoljuk be.

Példa:

```
3 1 5 4     ⇒2 (azaz például a 3-as betehető az 5-ösbe, ezután az 1-es az
              5-ösben levő 3-asba, tehát marad az 5-ös és a 4-es)
```

Add meg, hogy az alábbi ládasorozatok esetén a ládák hány ládába pakolhatók!

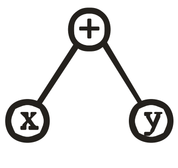
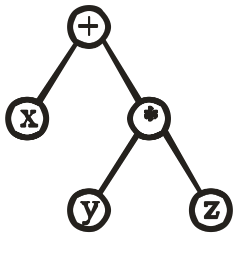
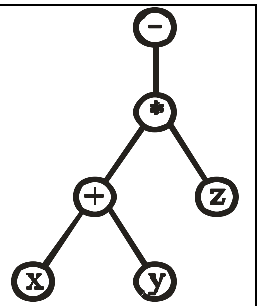
A. 1 2 3 4 5 5 4 3 2 1

B. 1 3 5 2 4 2 3 5 4 1 2 4

C. 2 4 2 4 3 5 3 5 2 3 2 3 4 5 4 5

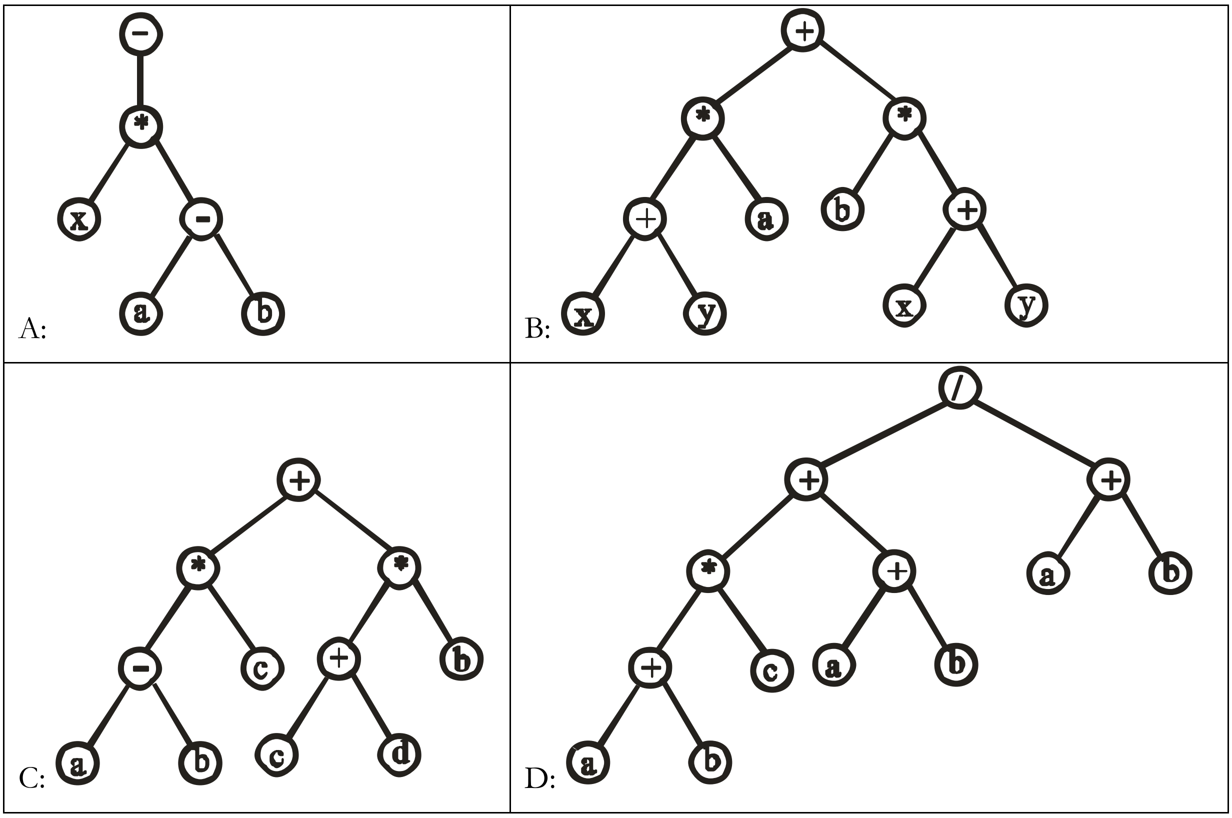
5. feladat: Kifejezésfa (24 pont)

Aritmetikai kifejezéseket az informatikában sokszor ábrázolnak fával. Például:

x+y:		x+y*z:		–(x+y)*z:	

Vannak olyan kifejezések, amelyek többféleképpen is felírhatók (pl. x+y+z), s vannak olyanok is, amelyek többféle felírásában más darabszámú művelet használható (pl. - - x ugyanaz, mint x, a második felírásban kettővel kevesebb műveleti jel kell.

Alakítsd át az alábbi kifejezésfákat úgy, hogy a kifejezések értéke ne változzon, de a lehető legkevesebb műveletet tartalmazzák, valamint add meg az eredeti fához tartozó „hagyományos” kifejezést és az átalakított fához tartozót is!



2008. Második forduló

Ötödik-nyolcadik osztályosok

1. feladat: Titkos kód (20 pont)

Egy legfeljebb N számjegyű titkos számkódot szeretnénk megfejteni, a kód nemnegatív egész szám. Az tudjuk csak róla, hogy mennyi az 5-tel (A), a 7-tel (B), illetve a 11-gyel (C) osztásának maradéka.

Írj programot, amely beolvassa a kód hosszát (1≤N≤4), valamint A és B és C értékét, majd megadja a legkisebb olyan számkódot, ami a maradékok alapján lehetséges, továbbá hogy hány ilyen legfeljebb N-jegyű szám van összesen!

Példa:

```
N=3, A=3, B=4, C=0      ⇒  A legkisebb ilyen szám: 88.
                            Összesen 3 ilyen szám van.
```

Megjegyzés: a továbbiak: 473, 858 (ezt nem kell kiírni)

2. feladat: Szótagolás (35 pont)

Készíts programot, amely kisbetűvel írt magyar szavak szótagolására képes! Mivel a magyar nyelv szabályai bonyolultak, ezért nem kell minden szóra helyesen működnie, az alábbi szigorító feltételeket kell figyelembe venni:

- csak egyszerű szavakat kell szótagolni, összetett szavakat nem;
- hosszú kétjegyű mássalhangzókat (pl., ssz) nem használunk;
- ha egymás mellé kerül két olyan karakter, ami kétjegyű mássalhangzónak értelmezhető (pl. cs), akkor azt mindig annak is kell értelmezni (azaz nem kell helyesen szótagolni az olyan szavakat, mint pl. a `kurucság`).

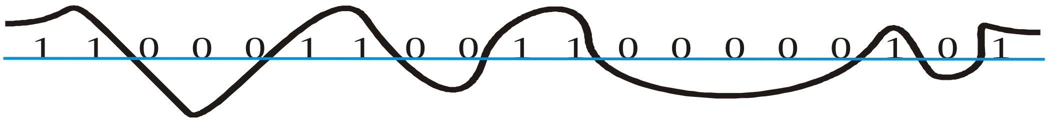
Példák:

`szavak`	⇒	`sza-vak`
`balta`	⇒	`bal-ta`
`keszeg`	⇒	`ke-szeg`
`jobbra`	⇒	`jobb-ra`
`könyvtár`	⇒	`könyv-tár`
`beosztott`	⇒	`be-osz-tott`

3. feladat: Sziget (20 pont)

Régen a polinéz szigetvilágot az őslakók szigetről szigetre hajózva népesítették be. Egyes hajókkal csak a közelebbi szigetekre tudtak eljutni, másokkal távolabbra is.

A feladatban a szigetek egy vonalban, egymás mellett helyezkednek el az ábrának megfelelően (1-esek jelzik a szárazföldet, 0-k pedig a tengert):



Bal, illetve jobb oldalról indulhatnak hajók, amely a szigetek között legfeljebb T darab 0-val leírt távolságot tudnak megtenni.

Készíts programot, amely az N adat ($1 \le N \le 1000$), a T távolság ($1 \le T \le 100$) és a szigetek elhelyezkedése ismeretében megadja, hogy az őslakók hány szigetet tudnak benépesíteni (balról, illetve jobbról indulva)!

Példa:

```
N=19, T=3, szigetek= 1 1 0 0 0 1 1 0 0 1 1 0 0 0 0 0 1 0 1
Eredmény: 3 sziget népesíthető be balról, 2 jobbról.
```

Magyarázat: a baloldali kiinduló sziget, innen elérhető a második, onnan pedig a harmadik sziget. A negyedikig azonban 5 egységet kellene hajózni, de csak 3 egységet tudnak. A jobbról induló a jobboldaliról csak egyetlen másik szigetet érhet el. Az eredménybe a kiinduló szigeteket is beleszámoljuk.

Kilencedik-tizedik osztályosok

1. feladat: Tagok (20 pont)

Egy titkos társaságban egyetlen főnök van. A társaság minden tagjának maximum két közvetlen beosztottja lehet, de ezek nem egyenrangúak (azaz más funkciója van az A- és más a B-típusú beosztottnak). A tagokat a sorszámukkal azonosítjuk. Egy új tag jelentkezik a társaságba, akit be kell osztani valakihez A- vagy B típusú beosztottnak.

Készíts programot, amely megadja, hogy az új tag hányféle helyre kerülhet:

A. a főnök közvetlen beosztottjaként;

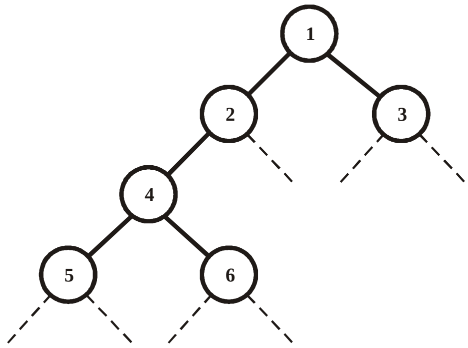
B. a főnök A típusú beosztottja bármelyik (nem csak közvetlen) beosztottjaként;

C. a főnök B típusú beosztottja bármelyik (nem csak közvetlen) beosztottjaként!

A *TAG.BE* állomány első sorában a tagok száma ($1 \le N \le 10\,000$) van. A következő N-1 sorban egy-egy tag leírása található: a sorszáma, a közvetlen főnöke sorszáma, valamint egy A vagy egy B betű, attól függően, hogy a főnökének milyen típusú beosztottja. A fő főnök az 1-es sorszámú, neki nincs leírása.

A *TAG.KI* állományba három sort kell írni, a három kérdésre adott választ!

Példa: (az ábrán az A típusú beosztottak a közvetlen főnöküktől balra, a B-típusúak pedig jobbra helyezkednek el)




```
TAG.BE          TAG.KI
6               0
2 1 A           5
4 2 A           2
5 4 A
6 4 B
3 1 B
```

2. feladat: Ismerősök (20 pont)

Egy közösségi portálra N ember jelentkezett be. Mindenki megadta, hogy kiket ismer, amit az ismerősük vissza is igazolt.

Készíts programot, amely megadja azon párokat

A. akiknek minden ismerősük közös (de van legalább 1);

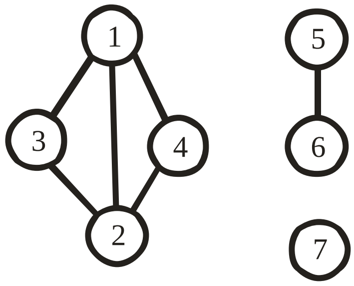
B. akiknek van közös ismerősük!

Az *ISMER.BE* állomány első sorában az emberek száma (1≤N≤100) és az ismeretségek száma (0≤M≤5000) van. A következő M sor mindegyike két egymást ismerő ember sorszámát tartalmazza (1≤i≠j≤N).

Az *ISMER.KI* állomány első sorába azon párok K számát kell írni, akiknek minden ismerősük közös, a következő K sorba pedig egy-egy ilyen pár sorszámát! A következő sorba azon párok L számát kell írni, akiknek van közös ismerősük, s az ezt követő L sorba pedig egy-egy ilyen pár sorszámát

Példa:

```
ISMER.BE        ISMER.KI
7 6             2
1 2             1 2        ismerőseik: 3,4
1 3             3 4        ismerőseik: 1,2
1 4             6
3 2             1 2
4 2             1 3
5 6             1 4
                2 3
                2 4
                3 4
```



3. feladat: Pakolás (20 pont)

Egymás mellé N darab ládát helyezünk el, különböző méretűeket. Tetszőleges üres láda megfogható és belehelyezhető valamely, tőle balra vagy jobbra elhelyezkedő nagyobb ládába, ha köztük nincs másik láda. Ha például balról jobbra haladva a ládák mérete rendre 1, 3, 2, akkor először a 2 méretűt rakjuk a 3 méretűbe, majd az 1 méretűt beletehetjük; de ha először az 1-est tesszük a 3-asba, akkor a 2-est már nem tehetjük bele. Az a cél, hogy a pakolás végén a ládákat a lehető legkevesebb ládába pakoljuk be.

Készíts programot, amely kiszámítja, hogy legkevesebb hány ládába lehet összepakolni a bemeneti ládasort!

A PAKOL.BE állomány első sorában a ládák száma (1≤N≤10 000) van. A második sor pontosan N különböző pozitív egész számot tartalmaz, a ládák méreteit balról jobbra sorrendben. A legnagyobb láda mérete N.

A PAKOL.KI állomány első és egyetlen sorába egy egész számot kell írni, ahány ládába a ládasor bepakolható!

Példa:

PAKOL.BE	PAKOL.KI
8	2
3 4 7 2 1 5 8 6	

4. feladat: Vitorlás (15 pont)

Egy vitorlás versenyen N futamot rendeznek, melyek mindegyikében az első K helyezettet értékelik. Az első helyezett K, a második K-1, a harmadik K-2, … pontot kap. Az összetett versenyben mindenkinek az L legmagasabb pontszámát veszik figyelembe. A versenyen M versenyző vett részt. A helyezést ezen pontszámok összege alapján csökkenő sorrendben határozzák meg. Ha két versenyzőnek ugyanannyi pontja lenne, akkor az kerül előbbre, akinek több első helyezése van; ha ugyanannyi első helyezésük van, akkor a második helyezések száma dönt, … és így tovább. Ha két versenyző ebben is egyforma, akkor a sorrendjük tetszőleges lehet.

Készíts programot, amely megadja azok névsorát, akik mindegyik futamon az első K hely valamelyikén végeztek, valamint add meg a verseny végeredményét!

A VITORLAS.BE állomány első sorában a futamok száma ($0<N\leq100$), a helyezések száma ($3\leq K\leq10$), a figyelembe veendő pontszámok száma ($2\leq L\leq N$) és a versenyzők száma ($1\leq M\leq1000$) van. Ezt követi N sor az egyes futamok sorrendjével. Minden sorban K versenyző sorszáma van, helyezésük szerint csökkenő sorrendben.

A VITORLAS.KI állomány első sorába azon versenyzők X számát kell írni, ahányan mindegyik futamban az első K helyezés valamelyikét érték el, a következő sorba pedig az X darab ilyen versenyzők sorszámát, tetszőleges sorrendben! A harmadikba az összes, pontot szerzett versenyző Y számát kell írni, a következő Y sorba pedig ezek sorszámát és összpontszámát, pontszám szerint csökkenő sorrendben!

Példa:

VITORLAS.BE	VITORLAS.KI	
5 5 3 15	2	
1 2 3 4 5	3 4	
2 4 6 8 3	9	
3 6 9 12 4	4 12	
5 4 3 2 1	1 11	
1 4 5 2 3	2 11	
	3 11	
	5 9	
	6 7	
	9 3	de az utolsó két helyen jó lenne:
	8 2	12 2
	12 2	8 2

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Tagok (15 pont)

Egy titkos társaságban egyetlen főnök van. A társaság minden tagjának maximum ***K*** közvetlen beosztottja lehet, de ezek nem egyenrangúak, fontos az is, hogy egy beosztott a közvetlen főnökének hányadik közvetlen beosztottja. A tagokat a sorszámukkal azonosítjuk. Egy új tag jelentkezik a társaságba, akit be kell osztani valakinek a közvetlen beosztottjaként.

Készíts programot, amely megadja, hogy az új tag hányféle helyre kerülhet:

A. a társaság főnökének közvetlen beosztottjaként;

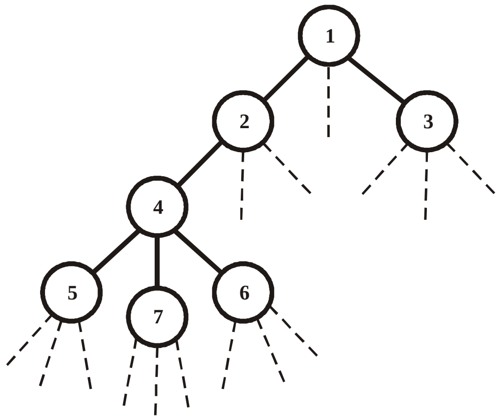
B. olyan tagként, akinek főnöke (nem feltétlenül közvetlen) lesz a társaság főnökének I-edik közvetlen beosztottja!

A *TAGOK.BE* állomány első sorában a tagok száma (1≤N≤10 000) és a közvetlen beosztottak maximális száma (1≤K≤100) van. A következő N-1 sorban egy-egy tag leírása található: a sorszáma, a közvetlen főnöke sorszáma, valamint az, hogy a közvetlen főnökének hányadik beosztottja. A társaság főnöke az 1-es sorszámú, neki nincs leírása, mert nem beosztottja senkinek.

A *TAGOK.KI* állományba két sort kell írni, a két kérdésre adott választ! Mivel a társaság főnökének is K közvetlen beosztottja lehet, ezért a második kérdésre K darab egész szám a válasz.

Példa:

TAGOK.BE	TAGOK.KI
6 3	1
2 1 1	11 0 3
4 2 1	
5 4 1	
6 4 3	
3 1 3	
7 4 2	



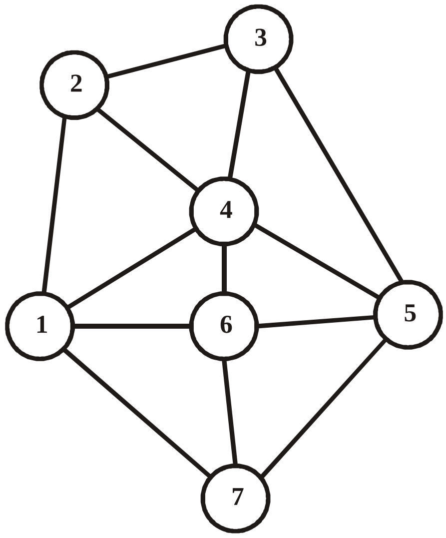
2. feladat: Város (15 pont)

Egy konvex sokszög alakú országban N város található. Az országot háromszög alakú megyékre osztják, ahol a háromszögek csúcsaiban vannak a városok.

Készíts programot, amely megadja, hogy hány város van az ország határán; illetve minden városra megadja, hogy hány másik várossal szomszédos (azaz közös háromszögben van),!

A *VAROS.BE* állomány első sorában a városok száma (3≤N≤1000) és a háromszögek száma (1≤M≤10 000) van. A következő M sorban egy-egy háromszög három csúcsán levő város sorszáma van.

A *VAROS.KI* állomány első sorába az országhatáron levő városok számát kell kiírni! A második sorba N számot kell írni, az i-edik szám az i-edik város szomszédainak száma legyen!



Példa:

VAROS.BE	VAROS.KI
7 7	5
1 6 7	4 3 3 5 4 4 3
6 7 5	
4 6 5	
4 6 1	
1 4 2	
2 3 4	
3 4 5	

3. feladat: Konténer (15 pont)

Egy raktárban egyetlen sorban egymás mellett van N darab kocka alakú konténer. Mindegyik konténer egy konténerhelyet foglal el a méretétől függetlenül. A raktár teljesen tele van és a raktárosnak helyet kell biztosítani újonnan érkező konténerek számára. Helyet csak úgy tud biztosítani, ha konténereket egymásra rak. A raktár biztonsági előírása szerint konténer csak nálánál nagyobb méretű konténerre rakható rá, de ennek betartásával akárhány konténer rakható egymásra. A raktáros dolgát nehezíti, hogy az átpakolást olyan robottal végezheti, amely bármely konténert fel tud venni, de csak balról jobbra haladva tud szállítani.

Készíts programot, amely kiszámítja, hogy legjobb esetben hány konténerhely szabadítható fel konténerek egymásra rakásával!

A *KONTENER.BE* állomány első sorában a konténerek száma (1≤N≤10 000) és a legnagyobb konténer méret (1≤K≤1000) van. A második sor pontosan N egész számot tartalmaz, a konténerek méretét balról jobbra haladó sorrendben.

A *KONTENER.KI* állomány első és egyetlen sorába a felszabadítható konténerhelyek maximálás számát kell írni!

Példa:

KONTENER.BE	KONTENER.KI
10 20	5
12 2 13 12 20 6 10 4 5 2	

4. feladat: Pakolás (15 pont)

Kamionnal kell elszállítani tárgyakat. Ismerjük a kamion kapacitását, tehát azt a súlyt, amelynél több nem rakható a kamionra, és ismerjük az elszállítandó tárgyak súlyát. Az a cél, hogy a kamiont úgy pakoljuk meg tárgyakkal, hogy az összsúly a lehető legnagyobb legyen.

Készíts programot, amely kiszámítja, hogy mekkora az a legnagyobb összsúly, amit a kamionnal elszállíthatunk! A program adja meg, hogy mely tárgyak kamionra rakásával érhető ez el.

A *PAKOL.BE* állomány első sorában a tárgyak száma (1≤N≤100) és a kamion kapacitása (1≤K≤600) van. A második sor pontosan N pozitív egész számot tartalmaz. Az i-edik szám az i-edik tárgy súlya, ami nem nagyobb, mint a kamion K kapacitása.

A *PAKOL.KI* állomány első sorába a kamionnal elszállítható legnagyobb S összsúlyt kell írni! A második sorba az S összsúlyt adó pakolásban szereplő tárgyak M számát kell írni! A harmadik sorba a kamionra pakolt M tárgy sorszámát kell írni tetszőleges sorrendben! Több megoldás esetén bármelyik megadható.

Példa:

PAKOL.BE	*PAKOL*.KI
6 20	19
18 12 4 7 10 5	3
	3 6 5

5. feladat: Kimérés (15 pont)

Egy gazda három tejeskannában gyűjti össze a napi tejet, amit három boltba szállít. Mindegyik boltba ugyanannyi liter tejet szállít, ezért átöntögetésekkel el kell érnie, hogy mindhárom kannában ugyanannyi tej legyen. Az áttöltögetések elvégzéséhez van két mérőpohara, az egyik A literes, a másik B literes. Tehát egy lépésben a gazda egyik kannából egy másik kannába tud áttölteni vagy A, vagy B liter tejet.

Készíts programot, amely kiszámítja, hogy legkevesebb hány áttöltést kell elvégezni, hogy mindhárom kannában ugyanannyi tej legyen! A program azt is adja meg, hogy áttöltések milyen sorozatával lehet ezt elérni.

A KIMER.BE állomány első sora a három kannában lévő tej mennyiségét (0≤T1,T2,T3≤100) tartalmazza, T1+T2+T3 osztható hárommal. A második sor két egész számot tartalmaz, a két mérőpohár űrtartalmát (1≤A,B≤100).

A KIMER.KI állomány első sorába a legkevesebb áttöltések M számát kell írni, amellyel elérhető, hogy mindhárom kannában ugyanannyi tej legyen! A további M sor mindegyike egy-egy áttöltést adjon meg három egész számmal: X Y Z, ami azt jelenti, hogy az X sorszámú (1≤X≤3) kannából az Y sorszámú (1≤Y≤3) kannába Z (Z=A vagy Z=B) liter tejet töltünk át! Ha nincs megoldás, akkor az első és egyetlen sor a -1 számot tartalmazza! Több megoldás esetén bármelyik megadható.

Példa:

KIMER.BE	KIMER.KI
7 3 8 2 5	3 3 1 2 1 2 5 2 1 2

2008. Harmadik forduló

Ötödik-nyolcadik osztályosok

1. feladat: Szél (24 pont)

A vitorlás versenyek rendezői megmérték N napon át Balatonfüreden a szélsebességet. Vitorlás versenyt szélcsendben nem lehet rendezni, de nagyobb viharban sem célszerű. Egy versenyen naponta egy fordulót rendeznek, s legalább K fordulóra van szükség.

Írj programot, amely beolvassa a napok számát (1≤N≤1000), a versenynapok számát (1≤K≤10) és az egyes napok szélsebességét (0≤sebesség≤200)!

Ezekből számolja ki,

A. hogy hány napon volt szélcsend;

B. hány napon volt kis szél (kisebb a szélsebesség, mint 10 km/óra);

C. hány napon volt vihar (nagyobb a szélsebesség, mint 100 km/óra);

D. melyik napon kezdhető meg a verseny (elkezdhető, ha volt olyan egymás utáni K nap, ahol a szélsebesség legalább 10 és legfeljebb 100 km/óra)!

Példa:

```
Bemenet:    N=10, K=3
            sebességek: 50,40,0,5,0,80,70,90,100,120
Kimenet:    Szélcsend: 2 nap
            Kis szél: 1 nap
            Vihar: 1 nap
            Versenyidő: 6. naptól
```

2. feladat: Vasút (25 pont)

Egy vasútvonal mellett N állomás található, ismerjük mindegyik távolságát a kiinduló állomástól. Egy utazó indul a kiinduló állomásról. Minél több állomáson meg kell állnia, de azzal a feltétellel, hogy ha egy helyen megállt, akkor a következő megállása legalább K kilométerre kell legyen!

Írj programot, amely beolvassa az állomások számát (2≤N≤100) és a K kilométert (1≤K≤100), valamint az egyes állomások távolságát a kiinduló állomástól (0=T(1)<T(2)<…<T(N), T(N)≥K)!

Ezekből számolja ki, hogy az utazó maximum hány állomáson állhat meg, s melyek lehetnek ezek az állomások! A kiinduló és a végállomáson mindenképpen meg kell állnia!

Példa:

Bemenet:	Kimenet:
N=5, K=10 T=(0,5,13,24,28)	3 állomáson állhat meg 1., 3., 5. állomás

3. feladat: Olimpia (26 pont)

Az Olimpiai Játékokon M ország vesz részt (1≤M≤100), N versenyszámban versenyeznek a résztvevők (1≤N≤1000). Minden versenyszámban 1 arany-, 1 ezüst-, valamint 1 vagy 2 bronzérmet adnak ki (kieséses versenyek esetén a döntőbe jutásért küzdők közül mindkét vesztes bronzérmet kap). Az országokat 1 és M közötti sorszámukkal azonosítjuk.

Készíts programot, amely az eredmények ismeretében előállítja az olimpia éremtáblázatát! Az éremtáblázat aranyérmek száma szerint csökkenő sorrendű legyen. Azonos aranyérem szám esetén a több ezüst-, azonos ezüstérem szám esetén a több bronzérem döntsön! Ha mindhárom éremből ugyanannyi van, akkor a kisebb sorszámú legyen elöl!

Példa:

Bemenet:	Kimenet:
M=5, N=3	2. ország: 1 A, 2 E, 2 B
1. szám: A – 2, E – 3, B – 2	1. ország: 1 A
2. szám: A – 1, E – 2, B – 3	5. ország: 1 A
3. szám: A – 5, E – 2, B – 2,3	3. ország: 1 E, 2 B

Kilencedik-tizedik osztályosok

1. feladat: Címlet (15 pont)

Egy országban N-féle pénzjegy címlet van (mindből tetszőleges darabszámú).

Készíts programot, amely megadja, hogy mely 1 és M közötti pénzösszegeket nem lehet a használható címletekkel pontosan kifizetni!

A CIMLET.BE állomány első sorában a pénzjegyek száma (1≤N≤100) és a maximális összeg értéke (1≤M≤1 000 000) van. A második sorban N szám van, az N címlet értéke, érték szerint növekvő sorrendben.

A CIMLET.KI állomány első sorába a nem kifizethető összegek számát kell írni! A második sorban a nem kifizethető összegek szerepeljenek, növekvő sorrendben!

Példa:

CIMLET.BE	CIMLET.KI
2 20	10
5 6	1 2 3 4 7 8 9 13 14 19

2. feladat: Vitorlás (20 pont)

Egy vitorlás versenyen K futamot rendeznek, naponta legfeljebb 1 futamot. Nem rendezhető futam, ha nagyon kicsi a szélsebesség, vagy ha nagy vihar van. Ha ilyen nap fordul elő, akkor szünnapot tartanak. Ismerjük N napra a szélsebesség előrejelzést. Az a cél, hogy a versenyt a lehető legkevesebb szünnappal rendezzék meg.

Készíts programot, amely megadja, hogy mely napokon kezdődhet a verseny úgy, hogy a lehető legkevesebb szünnap legyen!

A VITORLAS.BE állomány első sorában a napok száma (1≤N≤10 000), a futamok száma (1≤K≤100), valamint a legkisebb és a legnagyobb szélsebesség van, amikor még versenyt lehet rendezni (0≤sebesség≤200). A következő N sorban egy-egy napra a szélsebesség előrejelzése található (0≤sebesség≤200 km/óra).

A VITORLAS.KI állomány első sorába a lehetséges verseny kezdőnapok M számát kell írni, a második sorba pedig az M kezdőnap sorszámát! Ha nincs K napon megfelelő szélsebesség, akkor az eredmény egyetlen sorába 0-t kell írni!

Példa:

VITORLAS.BE	VITORLAS.KI
8 4 10 100	3
0	2 3 4
30	
30	
80	
110	
70	
60	
40	

3. feladat: Délkert (20 pont)

A DélKert szövetkezetben N termelő termel gyümölcsöt, amit a szövetkezet két hűtőházban gyűjt össze. Az i-edik termelő a leszedett gyümölcsöt az A hűtőházba a_i, a B hűtőházba b_i idő alatt tudja beszállítani. A hűtőházak kapacitása korlátozott, az A hűtőház N1, a B pedig N2 termelőtől tud gyümölcsöt fogadni. Az a cél, hogy minden termelőtől a lehető leghamarabb hűtőházba kerüljön a leszedett gyümölcs.

Készíts programot amely kiszámítja, hogy az egyes termelőknek melyik hűtőházba kell szállítnia a leszedett gyümölcsöt, hogy az összes termelőtől a gyümölcs a lehető legkorábban hűtőházba kerüljön!

A DELKERT.BE állomány első sorában a termelők száma (1≤N≤10000), az A hűtőház N1, és a B hűtőház N2 kapacitása van (N1+N2≥N). A második és a harmadok sor pontosan N pozitív egész számot tartalmaz. A második sorban az i-edik szám azt adja meg, hogy az i-edik termelő mennyi idő alatt tudja a gyümölcsöt az A hűtőházba szállítani. A harmadik sorban az i-edik szám azt adja meg, hogy az i-edik termelő mennyi idő alatt tudja a gyümölcsöt az B hűtőházba szállítani. A második és a harmadik sorban minden szám értéke legfeljebb 1000 lehet

A DELKERT.KI állományba három sort kell írni! Az első sorba azt a legkisebb K számot kell írni, amelyre teljesül, hogy minden termelő legfeljebb K idő alatt el tudja juttatni a gyümölcsét valamelyik hűtőházba, ha alkalmas beosztás szerint szállítanak! A második sorba azoknak a termelőknek a sorszámát kell kiírni, akik az A, a harmadik sorba pedig azokét, akik a B hűtőházba szállítják a gyümölcsöt! A sorokba a számokat tetszőleges sorrendben ki lehet írni. Ha több megoldás is van, bármelyik megadható.

Példa:

DELKERT.BE	DELKERT.KI
10 4 7	6
2 8 9 2 3 2 4 3 6 5	4 5 6 8
6 3 2 7 6 9 3 8 5 2	1 2 3 7 9 10

4. feladat: Falak (20 pont)

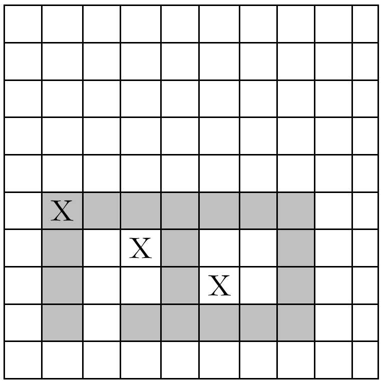
Egy négyzet alakú területre falakat helyezünk, amelyek 1 egység széles téglalapok. A téglalapok oldalai párhuzamosak a négyzet oldalaival. Arra vagyunk kíváncsiak, hogy bizonyos pontokból kiindulva eljuthatunk-e a négyzet szélére úgy, hogy falon nem megyünk át.

Készíts programot, amely kiszámítja, hogy az adott pontokból ki lehet-e jutni a négyzet szélére!

A FALAK.BE állomány első sorában a négyzet oldalhossza (1≤H≤1000), a falak száma (0≤L≤10 000) és a kiindulási pontok száma (1≤K≤5) van. A következő L sor mindegyike 4 számot tartalmaz, a fal bal alsó sarkának koordinátáit (1≤x, y≤H) és a fal oldalhosszait (1≤x+dx, y+dy≤H; dx és dy közül az egyik biztosan 1). Az utolsó K sorban a K kezdőpont koordinátái vannak (1≤kx, ky≤H).

A FALAK.KI állományba K sort kell írni! Ha az I-edik pontból ki lehet jutni, akkor az I-edik sorba az IGEN szó kerüljön, egyébként pedig a NEM! Ha a kezdőpont fal belsejében van, akkor is a NEM szót kell kiírni.!

Példa:



FALAK.BE	FALAK.KI
10 5 3	IGEN
2 2 1 4	NEM
4 2 5 1	NEM
3 5 6 1	
5 3 1 2	
8 3 1 2	
4 4	
6 3	
2 5	

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Robot (15 pont)

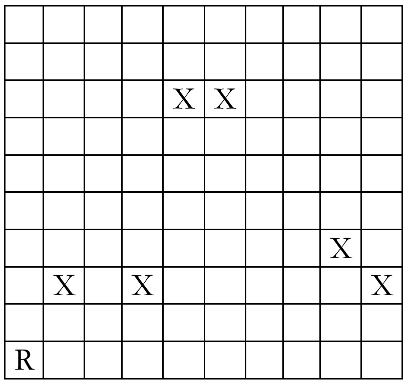
Egy gyárban a munkagépek négyzetrácsos elrendezésben vannak. A futószalagon érkező tárgyakat egy robotnak kell elszállítania a rendeltetési helyére. A robot a (0,0) mezőről indul, a tárgyakat érkezési sorrendjükben veheti le a futószalagról és egyszerre legfeljebb 3 tárgyat szállíthat. Ha több tárgyat szállít, akkor azokat tetszőleges sorrendben adhatja le a rendeltetési helyre. A robot a munkagépek felett mozoghat, egy lépésben szomszédos mezőre léphet egyet: balra, jobbra, felfelé, vagy lefelé. Egy lépése egy időegységet igényel. Miután leadta az egy menetben szállított tárgyakat, vissza kell térnie a kiindulási helyére, a (0,0) mezőre.

Készíts programot, amely kiszámítja, hogy legkevesebb mennyi idő alatt tudja a robot elszállítani az összes tárgyat, és meg is ad egy szállítási ütemezést!

A ROBOT.BE állomány első sorában a tárgyak száma (≤N≤10 000) van. A következő N sor mindegyikében két pozitív egész szám van (1≤X,Y≤1000), egy tárgy rendeltetési helyének koordinátái. Ugyanarra a helyre több tárgy is érkezhet.

A ROBOT.KI állomány első sorába azt a legkisebb M számot kell írni, amely alatt a robot az összes tárgyat el tudja szállítani a rendeltetési helyére! A második sorba egy számsorozatot kell írni, amely megadja, hogy a robot egy-egy menetben hány tárgyat szállít! A számsorozat minden eleme 1,2, vagy 3 lehet!

Példa:



ROBOT.BE	ROBOT.KI
6	54
1 2	3 3
3 2	
4 7	
8 3	
5 7	
9 2	

2. feladat: Szolga (15 pont)

Egy számítógépes hálózat N csomópontot tartalmaz. Azt mondjuk, hogy az Y csomópont közvetlen szomszédja az X csomópontnak, ha össze vannak kötve kétirányú adatátvitelt biztosító közvetlen vonallal. (Tehát, ha Y szomszédja X-nek, akkor X is szomszédja Y-nak.) Van K darab csomópont, amelyek névfeloldó szolgáltatást tudnak adni. Ha egy csomópontban lévő gép névfeloldást kíván, akkor a kérését el kell juttatnia valamelyik kiszolgálóhoz. Ha valamelyik közvetlen szomszédja kiszolgáló, akkor a kérését ennek továbbítja, amelyik azt meg is válaszolja. Egyébként a kérést valamelyik közvetlen szomszédjának kell átadni, aki azt továbbítja, és így tovább, amíg a kérés valamelyik kiszolgálóhoz nem ér, aki megválaszolja, és visszaküldi a választ ugyanazon útvonalon, amelyiken érkezett. A válaszidőt az határozza meg, hogy hány csomóponton keresztül jut el a kérés a kiszolgálóhoz.

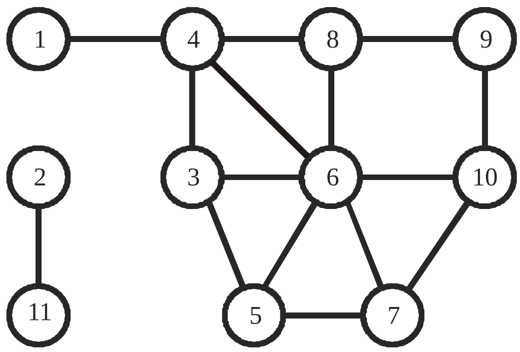
Készíts programot, amely minden csomópontra kiszámítja, hogy a csomópont melyik közvetlen szomszédjának küldje a kérést, ha az a cél, hogy a legrövidebb időn belül megkapja a választ!

A SZOLGA.BE állomány első sorában a csomópontok száma `(1≤N≤10 000)` és a kiszolgálók száma `(1≤K≤1000)` van. A második sor a `K` kiszolgáló sorszámait tartalmazza. A következő `N` sor írja le a hálózatot. Az állomány `i+2`-edik sorában azok a csomópontok vannak felsorolva és `0`-val zárva, amelyek az `i` csomópont közvetlen szomszédjai. A hálózat legfeljebb `100 000` közvetlen vonalat tartalmaz.

A SZOLGA.KI állomány első sorába azt az `M` számot kell írni, amelyre teljesül, hogy bármely csomópont kérése megválaszolható úgy, hogy legfeljebb `M` csomóponton keresztül jut el a kérés valamely kiszolgálóhoz (beleértve a kiszolgálót, de nem számítva a kérést küldőt)! A következő `N` sor mindegyikébe két számot kell írni, az `i+1`-edik sorban az első szám a legkevesebb csomópont száma, amelyen keresztülhalad az i-edik csomópont kérése! A második szám pedig annak a csomópontnak a sorszáma legyen, amelyiknek az `i` csomópont a kérését továbbítja! A kiszolgálók esetén a `0 i` számpár legyen kiírva! Ha nincs olyan útvonal, amelyen az i-edik csomópont eljuthatna valamely kiszolgálóhoz, akkor a `0 0` számpárt kell kiírni! Több megoldás esetén bármelyik megadható.

Példa:

SZOLGA.BE	SZOLGA.KI
11 3	3
1 3 5	0 1
4 0	0 0
11 0	0 3
4 6 5 0	1 1
1 3 6 8 0	0 5
7 3 6 0	1 3
4 3 5 7 10 8 0	1 5
5 6 10 0	2 4
4 6 9 0	3 8
8 10 0	2 6
9 6 7 0	0 0
2 0	



3. feladat: Intervallumok (15 pont)

Adott zárt intervallumoknak egy halmaza és egy `K` szám. Azt mondjuk, hogy az `[a1,b1]` és az `[a2,b2]` zárt intervallumoknak van közös része, ha `a1≤a2≤b1`, vagy `a2≤a1≤b2`.

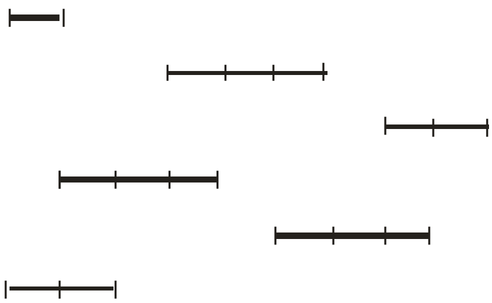
Készíts programot, amely kiszámítja azt a legszűkebb `[A,B]` zárt intervallumot, amelynek legalább `K` bemeneti intervallummal van közös része!

Az INTER.BE állomány első sorában az intervallumok száma `(1≤N≤50 000)`, és a `K` szám `(1≤K≤N)` van. A következő `N` sor mindegyikében egy intervallum bal és jobb végpontja van, mindegyik legfeljebb `3600` lehet. A bal végpont mindig kisebb, mint a jobb végpont.

Az INTER.KI állomány első és egyetlen sorába azt az `A` és `B` (`A<B`) számpárt kell írni, amelyre teljesül, hogy az `[A,B]` zárt intervallummal legalább `K` bemeneti intervallumnak van közös része, és `B-A` a lehető legkisebb! Ha több ilyen `(A,B)` számpár létezne, akkor azt kell kiírni, amelyikre az `A` a legkisebb!

Példa:

```
INTER.BE        INTER.KI
6 4             2 4
1 2
4 7
8 10
2 5
6 9
1 3
```



4. feladat: Lapok (15 pont)

Egy négyzet alakú fehér területre színes téglalapokat helyezünk, amelyek akár át is fedhetik egymást. A téglalapok oldalai párhuzamosak a fehér négyzet oldalaival. Arra vagyunk kíváncsiak, hogy a végén hány egyszínű összefüggő terület alakul ki.

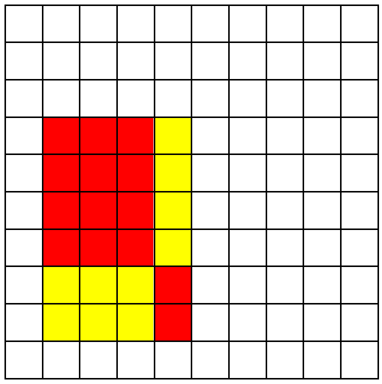
Készíts programot, amely kiszámítja a lerakott lapok alapján, hogy hány egyszínű összefüggő terület lett!

A LAPOK.BE állomány első sorában a négyzet oldalhossza (1≤H≤1000) és a színes lapok száma (0≤L≤10 000) van. A következő L sor mindegyike 5 számot tartalmaz, a színes négyzet bal alsó sarkának x és y koordinátáit (1≤x, y≤H), a téglalap oldalhosszait (1≤x+dx, y+dy≤H), valamint a színének kódját (1≤szín≤1000), a lerakás sorrendjében.

A LAPOK.KI állományba egyetlen sort kell írni, a lapok lerakása utáni egyszínű összefüggő területek számát!

Példa:

```
LAPOK.BE        LAPOK.KI
10 4            5
2 2 3 2 1
5 2 1 2 2
2 4 4 4 2
5 4 1 4 1
```



5. feladat: Hírek (15 pont)

Egy iskola tanulóiról tudjuk, hogy ha valaki érdekes hírt kap, akkor kiknek adja tovább.

Készíts programot, amely kiszámítja, hogy ki legyen az a kiválasztandó K tanuló, akikhez eljuttatva egy hírt, a legtöbb tanulóhoz eljut a hír!

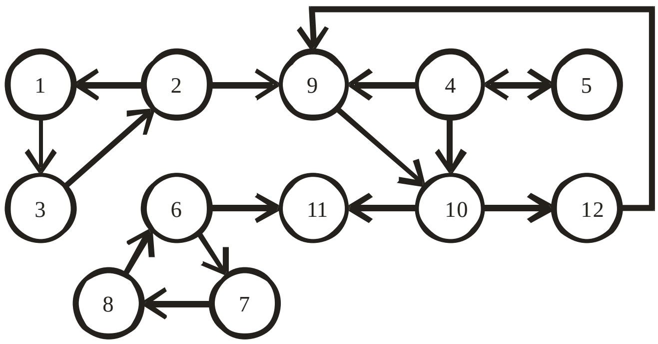
A HIREK.BE állomány első sora a tanulók számát (1≤N≤10 000), és a kiválasztandó tanulók számát (1≤K≤2) tartalmazza. A következő N sor írja le, hogy az egyes tanulók kiknek adják tovább a hírt. Közülük az i-edik sorban azoknak a tanulóknak a sorszáma van felsorolva és 0-val zárva, akiknek az i-edik tanuló továbbadja a hírt. Az utolsó N sorban legfeljebb 100 000 nem nulla szám van.

A HIREK.KI állomány első sora azon tanulók M számát tartalmazza, akikhez eljut a hír (beleértve a kiválasztott tanulókat is), ha a második sorban megadott tanulókhoz juttatjuk el! A második sor pontosan K kiválasztott tanuló sorszámát tartalmazza! Több megoldás esetén bármelyik megadható.

Példa:

```
HIREK.BE        HIREK.KI
12 2            10
3 0         2 7
9 1 0
2 0
```

```
5 9 10 0
4 0
7 11 0
8 0
6 0
10 0
11 12 0
0
9 0
```



A verseny végeredménye:

I. korcsoport

1.	Kővágó Zoltán	Szent István Gimnázium, Budapest
	Kovács Gábor Ferenc	Árpád Gimnázium, Tatabánya
	Tilk Bence	Balassi Bálint Általános Iskola, Eger
	Adrián Patrik	Fazekas Mihály Gimnázium, Debrecen
5.	Erdős Gergely	Batthyány Lajos Gimnázium, Nagykanizsa
6.	Marussy Kristóf	Szent István Gimnázium, Budapest
7.	Weisz Gellért	Fazekas Mihály Gimnázium, Budapest
	Szenczi Zoltán	Fazekas Mihály Gimnázium, Budapest
9.	Major Attila	Radnóti Miklós Gimnázium, Szeged
10.	Manninger Mátyás	Juhász Gyula Általános Iskola, Vác

II. korcsoport

1.	Turi Zsolt	Neumann János Számítástechnikai Szakközépiskola, Budapest
2.	Éles András	Fazekas Mihály Gimnázium, Debrecen
3.	Wagner Zsolt	Fazekas Mihály Gimnázium, Budapest
4.	Hunyady Márton	Bencés Gimnázium, Pannonhalma
5.	Dankovics Attila	Veres Péter Gimnázium, Budapest
	Rutai Richárd	Szent István Gimnázium, Budapest
7.	Hevele István	Orbán Balázs Gimnázium, Székelykeresztúr
8.	Miglász Dániel	Bárdos László Gimnázium, Tatabánya
	Papp Pál András	Szent István Gimnázium, Budapest
10.	Ács Kurucz Gábor	Révai Miklós Gimnázium, Győr
	Mezei Zsolt	Bolyai Farkas Elméleti Líceum, Marosvásárhely

III. korcsoport

1.	Szalkai Balázs	Lovassy László Gimnázium, Veszprém
2.	Nagy Gergely Gábor	Fazekas Mihály Gimnázium, Budapest
3.	Danka Miklós András	Fazekas Mihály Gimnázium, Budapest
4.	Eisenberger András	Fazekas Mihály Gimnázium, Budapest
5.	Hadházi Dániel	Neumann János Középiskola és Kollégium, Eger
6.	Korándi Dániel	Fazekas Mihály Gimnázium, Budapest
7.	Badics Alex	Eötvös József Gimnázium, Tata
8.	Szárnyas Gábor	Bólyai János Gyak. Általános Iskola és Gimnázium, Szombathely
9.	Mészáros András	Révai Miklós Gimnázium, Győr
10.	Szebeni Szilveszter	Illyés Gyula Gimnázium és Szakközépiskola, Budaörs
11.	Pósfai Gergely	Ságvári Endre Gimnázium, Szeged
12.	Danner Gábor	Ságvári Endre Gimnázium, Szeged
13.	Gévay Gábor	Ságvári Endre Gimnázium, Szeged
14.	Halász Gábor	Boronkay György Műszaki Szakközépiskola, Vác
15.	Korom-Vellás Judit	Radnóti Miklós Gimnázium, Szeged

2009. Első forduló

Ötödik-nyolcadik osztályosok

1. feladat: Vércsoport (18 pont)

Egy játékban két kártyát (X, Y) húzunk egy-egy pakliból. Mindkét pakliban találhatók A-betűt, B-betűt és 0-s számjegyet tartalmazó kártyák. Add meg, hogy az alábbi algoritmus alapján milyen esetekben hány pontot kaphatunk:

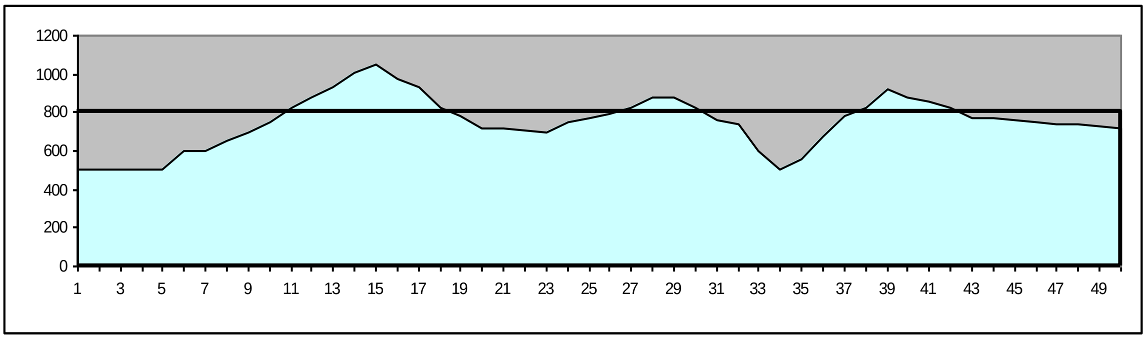
```
V:=X="A" vagy Y="A"
W:=X="B" vagy Y="B"
Ha V és W akkor Pont:=3
különben Ha V akkor Pont:=2
különben Ha W akkor Pont:=1
különben Pont:=0
```

Megjegyzés: Ez az algoritmus az egyik legfontosabb emberi vércsoportrendszer öröklésmenetét mutatja be.

X	Y	Pont
A	A	
A	B	
A	0	
B	A	
B	B	
B	0	
0	A	
0	B	
0	0	

2. feladat: Hegy (30 pont)

Egy hegymászó az útja során N helyen (N>2) feljegyezte, hogy milyen tengerszint feletti magasságban jár.




```
N=50, a mért értékek:
500,500,500,500,500,600,600,650,700,750,820,880,930,1010,1050,
980,930,830,780,720,720,710,700,750,770,790,820,880,880,820,
760,740,600,500,560,670,780,820,920,880,860,820,770,770,760,
750,740,740,730,720.
```

Add meg, hogy az alábbi programrészletekben a fenti értékek alapján mi lesz az értéke a DB, illetve az E feladatban az M változónak! Fogalmazd meg általánosan is, hogy a DB és az M változó mit tartalmaz!

```
A: DB:=0
   Ciklus i=1-től N-ig
     Ha X(i)>800 akkor DB:=DB+1
   Ciklus vége
```

```
B: DB:=0
   Ciklus i=2-től N-ig
     Ha X(i)>800 és X(i-1)<=800 akkor DB:=DB+1
   Ciklus vége
```

```
C: DB:=0
   Ciklus i=2-től N-1-ig
     Ha X(i)>X(i-1) és X(i)>X(i+1) akkor DB:=DB+1
   Ciklus vége
D: DB:=0
   Ciklus i=2-től N-1-ig
     Ha X(i)<X(i-1) és X(i)<X(i+1) akkor DB:=DB+1
   Ciklus vége
E: M:=0
   Ciklus i=2-től N-ig
     Ha X(i)-X(i-1)>M akkor M:=X(i)-X(i-1)
   Ciklus vége
```

3. feladat: Római szám (20 pont)

Az alábbi eljárás 1 és 99 közötti egész számokat ad meg római számokkal. Segítségként nézzük meg a római számok táblázatát:

1 = I	11 = XI	21 = XXI	60 = LX
2 = II	12 = XII	22 = XXII	70 = LXX
3 = III	13 = XIII	28 = XXVIII	74 = LXXIV
4 = IV	14 = XIV	29 = XXIX	80 = LXXX
5 = V	15 = XV	30 = XXX	89 = LXXXIX
6 = VI	16 = XVI	40 = XL	90 = XC
7 = VII	17 = XVII	49 = XLIX	99 = XCIX
8 = VIII	18 = XVIII	50 = L	
9 = IX	19 = XIX		
10 = X	20 = XX		

```
Római(N):
  R:=’’
  Ha N≥90 akkor R:=R+’XC’; N:=N-90      {1}
  Ha N≥50 akkor R:=R+’L’; N:=N-50   {2}
  Ha N≥40 akkor R:=R+’XL’; N:=N-40      {3}
  Ha N≥10 akkor R:=R+’X’; N:=N-10   {4}
  Ha N≥10 akkor R:=R+’X’; N:=N-10   {5}
  Ha N≥10 akkor R:=R+’X’; N:=N-10   {6}
  Ha N=9 akkor R:=R+’IX’; N:=0      {7}
  Ha N=4 akkor R:=R+’I’; N:=5       {8}
  Ha N>4 akkor R:=R+’V’; N:=N-5     {9}
  Ha N≥1 akkor R:=R+’I’; N:=N-1     {10}
  Ha N≥1 akkor R:=R+’I’; N:=N-1     {11}
  Ha N≥1 akkor R:=R+’I’; N:=N-1     {12}
Eljárás vége.
```

Add meg, hogy az alábbi számok esetén mely elágazáságakon levő utasítások hajtódnak végre (pl. 12 esetén a 4., 10. és 11. ág): N= 1, 3, 7, 24, 45, 62, 89

Számítógépes feladat – VÁLASZTHATÓ

4. feladat: Hőmérséklet (32 pont)

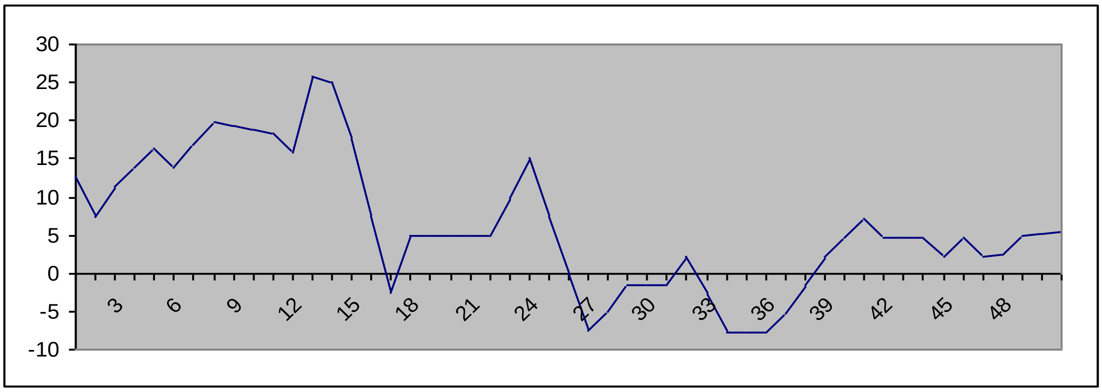
Az elmúlt N nap (1≤N≤50) hőmérséklete alapján számold meg:

A. hányszor fagyott,

B. hány fagyos időszak volt,

C. mennyi volt a leghidegebb!

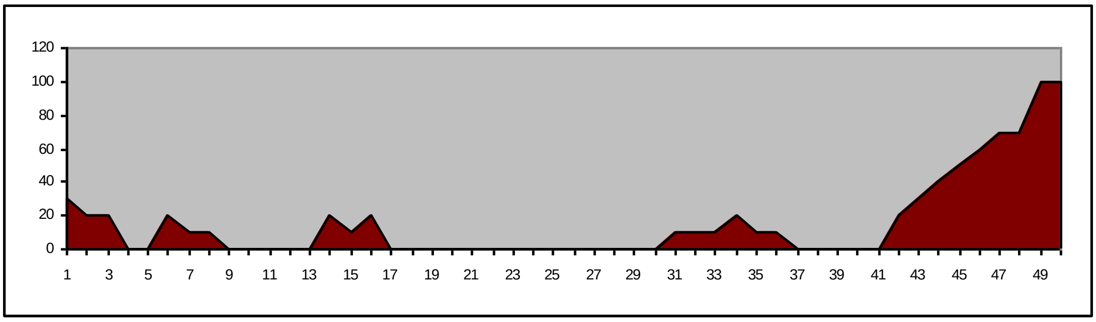
Akkor fagy, ha a hőmérséklet <0 fok. (Az ábrán 3 fagyos időszak látható; 12 nap fagyott – a 26. napon 0 fok volt, ami még nem fagy; a leghidegebb -7 fok volt.)



Számítógép nélküli feladat – VÁLASZTHATÓ

4. feladat: Szigetek (32 pont)

Európából repültünk Amerikába, s útközben N helyen (N>2) mértük a tengerszint feletti magasságot. Tenger esetén ez 0, Európában, Amerikában, illetve a közbülső szigeteken pedig >0. Az alábbi algoritmusrészlet a szigetek kezdetét (K – az első nem 0 helye) és végét (V – az utolsó nem 0 helye) határozná meg, ha jó lenne. (Az ábrán három sziget van, az első sziget a 6. méréstől a 8. mérésig tart.)




```
E:=az első 0 helye
U:=az utolsó 0 helye
DB:=0
Ciklus i=E+1-től N-1-ig
  Ha X(i)>0 vagy X(i-1)=0 akkor DB:=DB+1; K(DB):=i
  Ha X(i-1)>0 és X(i)=0 akkor DB:=DB+1; V(DB):=i
Ciklus vége
```

A. Add meg, hogy mik a hibák benne!

B. Új utasítások beszúrása nélkül javítsd ki az algoritmusrészletet!

Kilencedik-tizedik osztályosok

1. feladat: Kakastaréj (16 pont)

Egy játékban egy-egy kártyát (A, B, X, Y) húzunk négy kártyacsomagból. Az A és B kártyát olyan csomagból húzzuk, amelyikben R és r betűt tartalmazó kártyák vannak, az X és Y kártyát pedig olyan csomagból húzzuk, amelyikben P és p betűt tartalmazó kártyák vannak. Add meg, hogy az alábbi algoritmus alapján milyen esetekben mit kapunk eredményként:

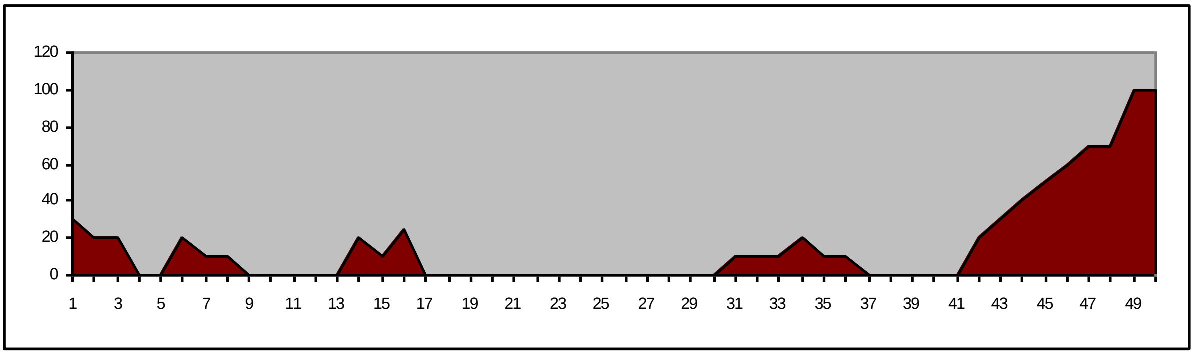
```
V:=A="R" vagy B="R"
W:=X="P" vagy Y="P"
Ha V és W akkor Taréj:="Dió"
különben Ha V akkor Taréj:="Rózsa"
különben Ha W akkor Taréj:="Borsó"
különben Taréj:="Egyszerű"
```

<u>Megjegyzés</u>: Ez az algoritmus a kétgénes domináns-recesszív öröklésmenetet (pl. a kakastaréj alakjának öröklődése) mutatja be.

A	B	X	Y	Taréj
A	B	X	Y	
R	R	P	P	
R	R	P	p	
R	R	p	P	
R	R	p	p	
R	r	P	P	
R	r	P	p	
R	r	p	P	
R	r	p	p	
r	R	P	P	
r	R	P	p	
r	R	p	P	
r	R	p	p	
r	r	P	P	
r	r	P	p	
r	r	p	P	

<u>2. feladat: Szigetek (34 pont)</u>

Európából repültünk Amerikába, s útközben N helyen (N>2) mértük a felszín tengerszint feletti magasságát. Tenger esetén ez 0, Európában, Amerikában, illetve a közbülső szigeteken pedig >0.



N=50, a mért adatok: 30,20,20,0,0,20,10,10,0,0,0,0,0,20,10,25, 0,0,0,0,0,0,0,0,0,0,0,0,0,0,10,10,10,20,10,10,0,0,0,0,0,20,30, 40,50, 60,70,70,100,100

Nézzük a következő programrészletet!

```
E:=1
Ciklus amíg X(E)>0
  E:=E+1
Ciklus vége

U:=N
Ciklus amíg X(U)>0
  U:=U-1
Ciklus vége

DB:=0
Ciklus i=E+1-től U-1-ig
  Ha X(i)>0 és X(i-1)=0 akkor DB:=DB+1; K(DB):=i
  Ha X(i)>0 és X(i+1)=0 akkor V(DB):=i
Ciklus vége
```

A. Mi lesz a fenti értékekre az E értéke? Fogalmazd meg általánosan is!

B. Mi lesz a fenti értékekre az U értéke? Fogalmazd meg általánosan is!

C. Mi lesz a fenti értékekre a DB,K,V értéke? Fogalmazd meg általánosan is!

D. Mi lesz az M és az S értéke, ha az első program harmadik ciklusát az alábbiakra cseréljük? Fogalmazd meg általánosan is!

```
DB:=0; M:=0
Ciklus i=E+1-től U-1-ig
  Ha X(i)>0 és X(i-1)=0 akkor DB:=DB+1; K(DB):=i
  Ha X(i)>M akkor M:=X(i); S:=DB
  Ha X(i)>0 és X(i+1)=0 akkor V(DB):=i
Ciklus vége
```

E. Mi lesz az M és az S értéke, ha az első program harmadik ciklusát az alábbiakra cseréljük? Fogalmazd meg általánosan is!

```
DB:=0; M:=0
Ciklus i=E+1-től U-1-ig
  Ha X(i)>0 és X(i-1)=0 akkor DB:=DB+1; K(DB):=i
  Ha X(i)>0 és X(i+1)=0
     akkor V(DB):=i
          Ha V(DB)-K(DB)+1>M akkor M:=V(DB)-K(DB)+1; S:=DB
            Elágazás vége
  Elágazás vége
Ciklus vége
```

3. feladat: Mik a hibák? (16 pont)

Egy könyvtár könyveiről az alábbi formájú táblázatokat tárolja (egy könyvnek csak egy szerzője van):

```
kulcs: (kód,szó) párok sorozata (KulcsDB darab)
könyv: (kód,szerző,cím) hármasok sorozata (KönyvDB darab)
```

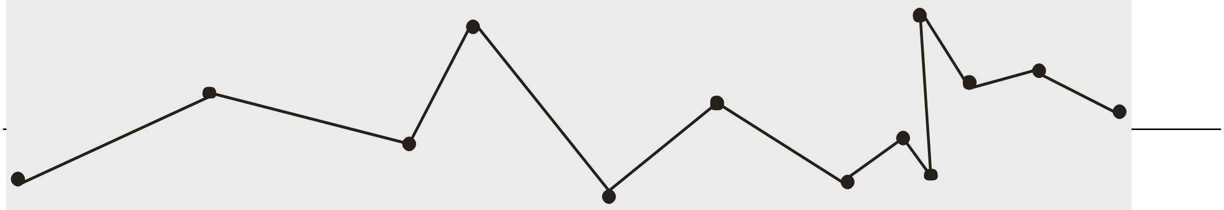
Az alábbi program az SZ vektor első DB elemében adná meg azon kulcsszavakat, amelyek pontosan egyetlen szerző könyveire illenek, ha jól működne! Add meg, mik benne a hibák!

```
kulcsszavak:
  DB:=0
  Ciklus i=1-től KulcsDB-ig
    i:=1
    Ciklus amíg j≤KönyvDB vagy kulcs(i).kód≠könyv(i).kód
      j:=j+1
    Ciklus vége
    Ha j>KönyvDB akkor {az i-edik kulcsszó illik a j-edik könyvre}
       k:=i+1
       Ciklus amíg k<KulcsDB és (kulcs(i).kód≠könyv(k).kód
                   vagy könyv(k).szerző=könyv(j).szerző)
         k:=k+1
       Ciklus vége
       Ha k>KönyvDB akkor {más szerző könyvére nem illik}
          DB:=DB+1; SZ(i):=kulcs(i).szó
       Elágazás vége
    Elágazás vége
  Ciklus vége
Eljárás vége.
```

4. feladat: Benzinkút (16 pont)

A Kukutyin Piripócs útvonalon N benzinkút található. Ismerjük az egyes benzinkutak távolságát, valamint azt, hogy tele tankkal az autónk hány kilométert tud megtenni (K). Számold ki az alábbi távolságok esetén, hogy minimum hány helyen kell tankolnunk, s mondd is meg, hogy mely benzinkutaknál! (Kukutyin az 1-es sorszámú, itt mindenképpen tankolnunk kell, mert az autó üres benzintankkal nem indul, Piripócson már nincs benzinkút!)

Add meg, hogy minimum hány helyen kell tankolni, s add meg azt is, hogy hol!



Megjegyzés: az ábrán az első pont Kukutyin, az utolsó (13.) pont pedig Piripócs.

A. N=12, K=350, a távolságok: 280,260,60,230,100,130,70,40,120,80,60,100.

B. N=12, K=360, a távolságok: 280,260,60,230,100,130,70,40,120,80,60,100.

5. feladat: Rekurzió (18 pont)

Egy x sorozatra az alábbi függvényeket értelmezzük (biztosan csak értelmes paraméterekre hívjuk meg őket):

`első(x)` – az x sorozat első elemét adja meg (`első((1 2 3)) = 1`)

`elsőnélküli(x)` – az x sorozatból az első elemet elhagyja
(`elsőnélküli((1 2 3)) = (2 3) )`

`elsőnek(y,x)` – az x sorozat elejére beszúrja y elemet
(`elsőnek(4,(1 2 3)) = (4 1 2 3)`

`üres?(x)` – igaz, ha az x sorozatnak nincs eleme

`üres` – egy üres sorozatot ad

Mit számolnak ki az alábbi rekurzív függvények az `egyik((1 2 3 4 5),3)`, `másik((1 2 3 4 5),3)`, `három((1 2 3 4 5),3,2)` hívások esetén?

```
egyik(x,y): ha y=0 akkor egyik:=üres
különben egyik:=elsőnek(első(x),egyik(elsőnélküli(x),y-1)).
```

```
másik(x,y): ha hossz(x)=y akkor másik:=x
különben másik:=másik(elsőnélküli(x),y).
```

```
három(x,y,z): ha y>1 akkor
                 három:=három(elsőnélküli(x),y-1,z)
  különben ha z=0 akkor három:=üres
  különben három:=elsőnek(első(x),
                        három(elsőnélküli(x),y,z-1)).
```

Fogalmazd meg a három függvény feladatát általánosan is (őket is csak értelmes paraméterekre hívjuk meg)!

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Kutyaszín (16 pont)

Egy játékban egy-egy kártyát (A, B, C, D) húzunk négy kártyacsomagból. Az A és B kártyát olyan csomagból húzzuk, amelyikben F és v betűt tartalmazó kártyák vannak, az C és D kártyát pedig olyan csomagból húzzuk, amelyikben G és g betűt tartalmazó kártyák vannak. Add meg, hogy az alábbi algoritmus alapján milyen esetekben mi lesz a `Szín` változó értéke!

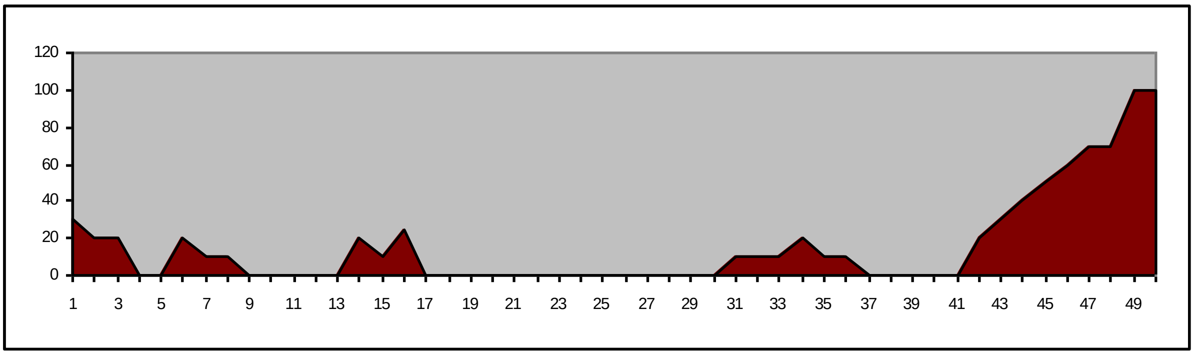
```
V:=A=”F” vagy B=”F”
W:=C=”G” vagy D=”G”
X:=C=”G” és D=”G”
Ha V és X akkor Szín:=”Világos”
különben Ha V és W akkor Szín:=”Sötét”
különben Ha V akkor Szín:=”Fekete”
különben Szín:=”Vörös”
```

Megjegyzés: Ez az algoritmus néhány genetikailag meghatározott jellegnek, például egyes kutyafajták (ír farkaskutya) színének öröklésmenetét mutatja be.

A	B	C	D	Szín
F	v	G	G	
v	F	G	G	
F	F	G	G	
v	v	G	G	
F	v	G	g	
v	F	G	g	
F	F	G	g	
v	v	G	g	
F	v	g	G	
v	F	g	G	
F	F	g	G	
v	v	g	G	
F	v	g	g	
v	F	g	g	
F	F	g	g	
v	v	g	g	

2. feladat: Szigetek (27 pont)

Európából repültünk Amerikába, s útközben N helyen (N>2) mértük a felszín tengerszint feletti magasságát. Tenger esetén ez 0, Európában, Amerikában, illetve a közbülső szigeteken pedig >0.




```
N=50, a mért adatok: 30,20,20,0,0,20,10,10,0,0,0,0,0,20,10,25,
0,0,0,0,0,0,0,0,0,0,0,0,0,0,10,10,10,20,10,10,0,0,0,0,0,20,30,
40,50, 60,70,70,100,100
```

Az alábbi algoritmusrészletet írtuk:

```
E:=1; U:=N
Ciklus amíg X(E)>0
  E:=E+1
Ciklus vége
Ciklus amíg X(U)>0
  U:=U-1
Ciklus vége

DB:=0
Ciklus i=E+1-től U-1-ig
  Ha X(i)>0 és X(i-1)=0 akkor DB:=DB+1; K(DB):=i
  Ha X(i)>0 és X(i+1)=0 akkor V(DB):=i
Ciklus vége
```

A. Mi lesz a fenti értékekre az E értéke? Fogalmazd meg általánosan is!

B. Mi lesz a fenti értékekre az U értéke? Fogalmazd meg általánosan is!

C. Mi lesz a fenti értékekre a DB,K,V értéke? Fogalmazd meg általánosan is!

D. Mi lesz az M és az S értéke, ha az első program harmadik ciklusát az alábbiakra cseréljük? Fogalmazd meg általánosan is!

```
DB:=0; M:=0
Ciklus i=E+1-től U-1-ig
  Ha X(i)>0 és X(i-1)=0 akkor DB:=DB+1; K(DB):=i
  Ha X(i)>X(M) akkor M:=X(i); S:=DB
  Ha X(i)>0 és X(i+1)=0 akkor V(DB):=i
Ciklus vége
```

E. Mi lesz az M és az S értéke, ha az első program harmadik ciklusát az alábbiakra cseréljük? Fogalmazd meg általánosan is!

```
DB:=0; M:=N
Ciklus i=E+1-től U-1-ig
  Ha X(i)>0 és X(i-1)=0 akkor DB:=DB+1; K(DB):=i
  Ha X(i)>0 és X(i+1)=0
    akkor V(DB):=i
          Ha V(DB)-K(DB)+1<M akkor M:=V(DB)-K(DB)+1; S:=DB
          Elágazás vége
  Elágazás vége
Ciklus vége
```

<u>3. feladat</u>: Mik a hibák? (16 pont)

Egy könyvtár könyveiről az alábbi formájú táblázatokat tárolja (a többszerzős könyveket minden szerzővel külön felsorolják, valamint nincs két egyforma című könyv):

```
kulcs: (kód,szó) párok sorozata (KulcsDB darab)
könyv: (kód,szerző,cím) hármasok sorozata (KönyvDB darab)
```

Az alábbi program megadná azon kulcsszavakat, amelyek pontosan egyetlen könyvre illenek, ha jól működne! Add meg, mik benne a hibák!

```
kulcsszavak:
  DB:=1
  Ciklus i=1-től KulcsDB-ig
    j:=1
    Ciklus amíg j≤KönyvDB és kulcs(j).kód=könyv(j).kód
      j:=j+1
    Ciklus vége
    Ha j<KönyvDB akkor {az i-edik kulcsszó illik a j-edik könyvre}
       k:=j+1
       Ciklus amíg k≤KönyvDB és (kulcs(i).kód≠könyv(k).kód
                            vagy könyv(k).cím=könyv(j).cím)
         j:=j+1
       Ciklus vége
       Ha k≤KulcsDB akkor {más című könyvre nem illik}
          DB:=DB+1; SZ(DB):=kulcs(k).szó
       Elágazás vége
    Elágazás vége
  Ciklus vége
Eljárás vége.
```

<u>4. feladat</u>: Pakolás (23 pont)

Egy fazekas műhelyében sorban várakoznak az kiégetésre váró tárgyak. Minden tárgyról tudjuk, hogy mennyi az a legkevesebb idő, ami a kiégetéséhez kell (`Idő(i)`). Az égetésre váró tárgyakat az érkezésük sorrendjében kell kiégetni. Egyszerre több tárgyat is rakhatunk a kemencébe, azonban legfeljebb annyit, amennyi a kemence adott kapacitása (`K`). Az égetési idő egy menetben mindig a kemencébe rakott tárgyak minimális égetési idejének a maximuma kell legyen.

<u>Példa</u>:

```
7 tárgy, 3 fér be, idők: 10,8,20,25,30,12,40

Égetési idő: 75

Egy lehetséges égetési sorrend: 1-2, 3-4, 5-6-7.
```

A. Add meg, hogy az alábbi adatok alapján mennyi lesz a minimális égetési idő és adj is meg egy égetési sorrendet!

A1. 5 tárgy, 2 fér be, idők: 15,25,15,25,15

A2. 5 tárgy, 2 fér be, idők: 20,25,15,25,20

A3. 7 tárgy, 3 fér be, idők: 15,10,20,15,30,25,20

B. Fogalmazd meg képlettel (`Opt(i)=`...), hogy mennyi az első *i* tárgy kiégetésének lehető legkisebb összideje!

Példa:

Ha K=1 lenne, akkor $Opt(i)=\sum_{j=1}^{i} Idő(j)$ lenne.

5. feladat: Rekurzió (18 pont)

Egy x sorozatra az alábbi függvényeket értelmezzük (biztosan csak értelmes paraméterekre hívjuk meg őket):

`első(x)` – az x sorozat első elemét adja meg (`első((1 2 3)) = 1`)

`elsőnélküli(x)` – az x sorozatból az első elemet elhagyja
(`elsőnélküli((1 2 3)) = (2 3)`)

`elsőnek(y,x)` – az x sorozat elejére beszúrja y elemet
(`elsőnek(4,(1 2 3)) = (4 1 2 3)`)

`üres?(x)` – igaz, ha az x sorozatnak nincs eleme

`üres` – egy üres sorozatot ad

Mit számolnak ki az alábbi rekurzív függvények az `egyik((1 2 3 4 5),3)`, `másik((1 2 3 4 5),3)`, `három((1 2 3 4 5),3,4)` hívások esetén?

```
egyik(x,y): ha y=0 akkor egyik:=üres
  különben egyik:=elsőnek(első(x),egyik(elsőnélküli(x),y-1)).
```

```
másik(x,y): ha hossz(x)=y akkor másik:=x
  különben másik:=másik(elsőnélküli(x),y).
```

```
három(x,y,z): ha y>1 akkor
                   három:=három(elsőnélküli(x),y-1,z-1)
  különben ha z=0 akkor három:=üres
  különben három:=elsőnek(első(x),
                          három(elsőnélküli(x),y,z-1)).
```

Fogalmazd meg a három függvény feladatát általánosan is (őket is csak értelmes paraméterekre hívjuk meg)!

2009. Második forduló

Ötödik-nyolcadik osztályosok

1. feladat: Törtek (32 pont)

A legtöbb programozási nyelven csak egész, illetve valós számokat használhatunk, törteket `számláló/nevező` formában nem. Emiatt az alábbi műveletekre nincs a nyelvekben utasítás:

$$\frac{1}{2}+\frac{1}{6}=\frac{2}{3} \quad \frac{1}{2}-\frac{1}{6}=\frac{1}{3} \quad \frac{1}{2}*\frac{1}{6}=\frac{1}{12} \quad \frac{1}{2}:\frac{1}{6}=\frac{3}{1}$$

Készíts programot (`tort.pas`,...), amely beolvassa két tört számlálóját és nevezőjét, majd kiírja az összegüket, a különbségüket, a szorzatukat és a hányadosukat tört alakban, egyszerűsítve! Csak pozitív számokkal dolgozunk, még a különbség is csak 0 vagy pozitív lehet. A 0 számot 0/1 alakban írjuk.

Példa:

```
Bemenet:                        Kimenet:
Első szám számlálója? 1         Összeg számlálója: 2
Első szám nevezője? 2           Összeg nevezője: 3
Második szám számlálója? 1      Különbség számlálója: 1
Második szám nevezője? 6        Különbség nevezője: 3
                                Szorzat számlálója: 1
                                Szorzat nevezője: 12
                                Hányados számlálója: 3
                                Hányados nevezője: 1
```

2. feladat: Hangrend (24 pont)

Egy magyar szó magas hangrendű, ha csak magas magánhangzók (e, é, i, í, ö, ő, ü, ű) vannak benne. Mély hangrendű, ha csak mély magánhangzókat (a, á, o, ó, u, ú) tartalmaz. Vegyes hangrendű pedig akkor, ha van benne magas és mély hangrendű magánhangzó is.

Írj programot, amely beolvas egy magyar szót, majd kiírja, hogy milyen hangrendű!

Példa:

```
Bemenet: almafa     Kimenet: mély hangrendű
```

3. feladat: Lövészverseny (19 pont)

Egy lövészversenyen a versenyzők egymás után lőnek, ismerjük az eredményeiket a szereplésük sorrendjében.

Készíts programot, amely beolvassa a versenyzők számát (1≤N≤100), az N db eredményt, majd megadja az alábbiakat:

A. Azokat a versenyzőket, akik a verseny valamely időszakában álltak az első helyen!

B. Azokat a versenyzőket, akik a verseny valamely időszakában álltak az utolsó helyen!

Példa:

```
Bemenet: N=6, eredmények: 594, 596, 582, 599, 590, 590
Kimenet:    Elsők: 1 2 4
            Utolsók: 1 3
```

Kilencedik-tizedik osztályosok

1. feladat: Rendszergazda (20 pont)

Egy vállalat 2 rendszergazdát alkalmaz. Mindkettő megmondta, hogy a következő N nap alatt mettől meddig szeretne szabadságra menni. Biztonságosnak azokat az időintervallumokat nevezzük, amikor mindkét rendszergazda dolgozik, veszélyesnek pedig azokat, amikor mindketten szabadságon vannak.

Készíts programot, amely megadja a biztonságos, illetve a veszélyes időintervallumokat!

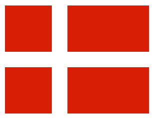
A `rendszer.be` állomány első sorában a napok száma van (1≤N≤1000). A második sorban az első rendszergazda tervezett szabadságai száma (0≤K≤N), a következő K sorban pedig az egyes szabadságai első és utolsó napjának sorszáma van. Az ezt követő sorban a második rendszergazda szabadságai száma (0≤L≤N) van, amit L sorban követ a szabadságok első és utolsó napjának sorszáma.

A `rendszer.ki` állomány első sorába a biztonságos időintervallumok B számát kell írni! A következő B sor mindegyikében egy-egy biztonságos időintervallum első és utolsó napjának sorszáma legyen! Ezt kövesse a veszélyes intervallumok V számát tartalmazó sor, majd pedig V darab sorban egy-egy veszélyes időintervallum első és utolsó napja!

Példa:

rendszer.be	rendszer.ki
50	4
3	1 4
5 10	11 11
40 45	25 39
15 20	46 50
1	1
12 24	15 20

2. feladat: Tömörítés (20 pont)



Geometrikus elemekből álló képeket (pl. a mellékelt dán zászlót) úgy tömöríthetünk, hogy minden egyes sorát azonos színű pontokból álló szakaszokra bontjuk (a példában az első sorban az 1. és a 3. pozíció között P, a 4. és a 4 pozíció között F, az 5. és a 10. pozíció között pedig P színű pontok vannak).

Írj programot, amely egy képet a fenti „futamhossz” eljárással tömörít!

A `tomor.be` állomány első sorában a kép sorai száma ($1 \le N \le 100$) és oszlopai száma ($1 \le M \le 100$) van. A következő N sor mindegyike M betűt tartalmaz, egy-egy szóközzel elválasztva. Az i-edik sor j-edik oszlopában a kép i-edik sora j-edik oszlopában levő képpont színét leíró betű található.

A `tomor.ki` állomány első sorába a kép sorai és oszlopai számát kell írni! A következő sorok a kódolt képet tartalmazzák, soronként, növekvő sorrendben. Minden sort A B párokkal írunk le, ami azt jelenti, hogy a kép adott sorában A darab B betű volt (pl. a minta első sorában 3 P betű, 1 F betű, majd 6 P betű szerepel).

Példa: (a dán zászló)

tomor.be	tomor.ki
7 10	7 10
P P P F P P P P P P	3 P 1 F 6 P
P P P F P P P P P P	3 P 1 F 6 P
P P P F P P P P P P	3 P 1 F 6 P
F F F F F F F F F F	10 F
P P P F P P P P P P	3 P 1 F 6 P
P P P F P P P P P P	3 P 1 F 6 P
P P P F P P P P P P	3 P 1 F 6 P

3. feladat: Huszár (15 pont)

Egy sakktáblára elhelyezünk egy huszárt. A sakktábla 8*8-as négyzet. A huszár „lóugrásban” lép, azaz vagy vízszintes irányban lép egyet és függőlegesen kettőt, vagy pedig fordítva.

Készíts programot, amely egy adott pozícióra elhelyezett huszár esetén megadja, hogy a huszár legfeljebb N lépés alatt mely pozíciókra juthat el!

A `huszar.be` állomány első sorában a huszár sorindexe ($1 \le Sor \le 8$), oszlopindexe ($1 \le Oszlop \le 8$), valamint a lépések maximális száma ($1 \le N \le 6$) van.

A `huszar.ki` állományba 8 sort kell írni, mindegyikben 8 karakter legyen egy-egy szóközzel elválasztva! Az i-edik sor j-edik oszlopába *-ot kell kiírni, ha a huszár ezt a pozíciót legfeljebb N lépés alatt elérheti, egyébként pedig szóköz!

Példa:

huszar.be

```
 1 1 2
```

huszar.ki

```
* * *
  **
**  *
 * *
* *
```

4. feladat: Olimpiai láng (20 pont)

Az olimpiai lángot egy kiindulási városból a cél városba kell eljuttatni. A két város távolsága `K` kilométer. A szervezők meghirdették, hogy olyan futók jelentkezését várják, akik pontosan `H` kilométert futnak az olimpiai lánggal. Sok futó jelentkezett, mindegyik megadta, hogy hányadik kilométertől vállalja a futást. A szervezők ki akarják választani a jelentkezők közül a lehető legkevesebb futót, akik végigviszik a lángot. Ha egy futó az `x` kilométertől fut, akkor minden olyan futó át tudja venni tőle a lángot, aki olyan `z` kilométertől vállalja a futást, hogy `z≤x+H`.

Írj programot, amely kiszámítja, hogy legkevesebb hány futó kell ahhoz, hogy a láng eljusson a cél városig!

A `lang.be` állomány első sorában a két város távolsága (`10≤K≤10 000`), a jelentkezett futók száma (`2≤N≤30 000`) és a lefutandó kilométer (`1≤H≤100`) van. A további `N` sor mindegyikében egy egész szám van (`0≤x≤K`) ami azt jelenti, hogy egy futó az `x`-edik kilométertől vállalja a láng továbbítását. Feltételezhetjük, hogy a láng eljuttatható a cél városig a jelentkezett futókkal.

A `lang.ki` állomány első sorába a láng célba juttatásához minimálisan szükséges futók `M` számát kell írni! A második sor pontosan `M` számot tartalmazzon, azon futók sorszámait, akik teljesítik a feladatot! A felsorolásban a `j`-edik futó a `j+1`-edik futónak adja át a lángot! Több megoldás esetén bármelyik megadható.

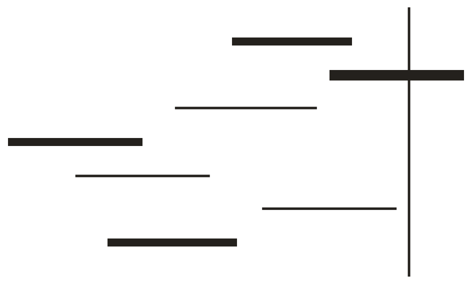
Példa:

lang.be

```
30 7 10
17
24
13
0
5
19
7
```

lang.ki

```
4
4 7 1 2
```



Tizenegyedik-tizenharmadik osztályosok

1. feladat: Rendszergazda (14 pont)

Egy vállalat 2 rendszergazdát alkalmaz. Mindkettő megmondta, hogy a következő évben mettől meddig szeretne szabadságra menni. Biztonságosnak azokat az időintervallumokat nevezzük, amikor mindkét rendszergazda dolgozik, veszélyesnek pedig azokat, amikor mindketten szabadságon vannak.

Készíts programot, amely megadja a biztonságos, illetve a veszélyes időintervallumokat!

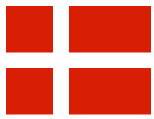
A `gazda.be` állomány első sorában az év van (1990≤EV≤2020). A második sorban az első rendszergazda tervezett szabadságai száma (0≤K≤100), a következő K sorban pedig az egyes szabadságai első és utolsó hónapjának és napjának sorszáma van. Az ezt követő sorban a második rendszergazda szabadságai száma (0≤L≤100) van, amit L sorban követ a szabadságok első hónapjának és napjának sorszáma, majd az utolsó hónapjának és napjának sorszáma.

A `gazda.ki` állomány első sorába a biztonságos időintervallumok B számát kell írni! A következő B sor mindegyikében egy-egy biztonságos időintervallum első és utolsó hónapjának és napjának sorszáma legyen! Ezt kövesse a veszélyes intervallumok V számát tartalmazó sor, majd pedig V darab sorban egy-egy veszélyes időintervallum első és utolsó hónapja és napja!

Példa:

gazda.be	gazda.ki
2009	4
3	1 1 1 4
1 5 1 10	1 11 3 11
12 4 12 5	3 25 12 3
3 15 3 20	12 6 12 31
1	1
3 12 3 24	3 15 3 20

2. feladat: Zászló (15 pont)



Geometrikus elemekből álló képeket (pl. a mellékelt dán zászlót) úgy tömöríthetünk, hogy minden egyes sorában csupán azt írjuk le, hogy mi változott az előző sorához képest. Ehhez az első sort azonos színű pontokból álló szakaszokra bontjuk (a példában az első sorban az 1. és a 3. pozíció között P, a 4. és a 4. pozíció között F, az 5. és a 10. pozíció között pedig P színű pontok vannak). Ezután csak azon sorokkal foglalkozunk, amelyek az előzőtől különböznek. Itt csak az előzőtől különböző részeket vizsgáljuk (a példában a 4. és 5. sor 1-3., illetve 5-10. pozíciója), amelyeket az első sorral megegyező módon kódolunk.

Írj programot, amely egy képet a fenti „változás”-tömörítési eljárással tömörít!

A `zaszlo.be` állomány első sorában a kép sorai (1≤N≤100) és oszlopai száma (1≤M≤100) van. A következő N sor mindegyike M betűt tartalmaz, egy-egy szóközzel elválasztva. Az i-edik sor j-edik oszlopában a kép i-edik sora j-edik oszlopában levő képpont színét leíró nagybetű található.

A `zaszlo.ki` állomány első sorába a kép sorai és oszlopai számát kell írni! A következő sorok a kódolt képet tartalmazzák, soronként, azon belül pedig oszloponként növekvő sorrendben. Ezekben három szám A, B, C és egy betű D van; jelentése: az A-adik sorban a B-edik pozíciótól a C-edik pozícióig D betű szerepelt a képen.

Példa: (a dán zászló)

zaszlo.be	zaszlo.ki
7 10	7 10
P P P F P P P P P P	1 1 3 P
P P P F P P P P P P	1 4 4 F
P P P F P P P P P P	1 5 10 P
F F F F F F F F F F	4 1 3 F
P P P F P P P P P P	4 5 10 F
P P P F P P P P P P	5 1 3 P
P P P F P P P P P P	5 5 10 P

3. feladat: Sakk (15 pont)

Egy sakktáblára elhelyezünk egy huszárt. A sakktábla 8*8-as négyzet. A huszár „lóugrásban” lép, azaz vagy vízszintes irányban lép egyet és függőlegesen kettőt, vagy pedig fordítva.

Készíts programot, amely egy adott pozícióra elhelyezett huszár esetén megadja, hogy a huszár mely pozíciókra minimum hány lépésben juthat el!

A `sakk.be` állomány első sorában a huszár sorindexe (`1≤Sor≤8`) és oszlopindexe (`1≤Oszlop≤8`) van.

A `sakk.ki` állományba 8 sort kell írni, mindegyikben 8 szám legyen! Az i-edik sor j-edik oszlopában az a lépésszám legyen, ahány lépésben az adott mező elérhető a kiinduló mezőről!

Példa:

sakk.be	sakk.ki
3 2	1 2 1 4 3 2 3 4
	2 3 2 1 2 3 4 3
	3 0 3 2 3 2 3 4
	2 3 2 1 2 3 4 3
	1 2 1 4 3 2 3 4
	2 3 2 3 2 3 4 3
	3 2 3 2 3 4 3 4
	4 3 4 3 4 3 4 5

4. feladat: Staféta (15 pont)

Az olimpiai lángot egy kiindulási városból a cél városba kell eljuttatni. A két város távolsága `K` kilométer. Sok futó jelentkezett, mindegyikről tudjuk, hogy hányadik kilométertől hányadik kilométerig vállalja a futást. Ha egy futó az `x` kilométertől az `y` kilométerig vállalja a futást, akkor minden olyan futó át tudja venni tőle a lángot, aki olyan `z` kilométertől vállalja a futást, hogy `x≤z≤y`.

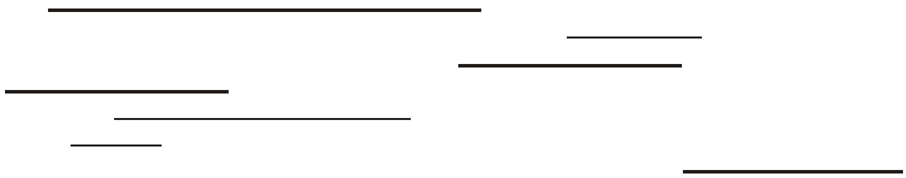
Írj programot, amely kiszámítja, hogy legkevesebb hány futó kell ahhoz, hogy a láng eljusson a cél városig.

A `stafeta.be` állomány első sorában a két város távolsága (`10≤K≤1000`) és a jelentkezett futók száma (`2≤N≤20 000`) van. A további `N` sor mindegyikében két egész szám van (`0≤I<E≤K`) ami azt jelenti, hogy egy futó az `I`-edik kilométertől az `E`-edik kilométerig vállalja a láng továbbítását. Feltételezhetjük, hogy a láng eljuttatható a cél városig a jelentkezett futókkal.

A `stafeta.ki` állomány első sorába a láng célba juttatásához minimálisan szükséges futók `M` számát kell írni! A második sor pontosan `M` számot tartalmazzon, azon futók sorszámait, akik teljesítik a feladatot! A felsorolásban a `j`-edik futó a `j+1`-edik futónak adja át a lángot! Több megoldás esetén bármelyik megadható.

Példa:

stafeta.be	stafeta.ki
40 7	4
2 21	4 1 3 7
25 35	
20 34	
0 10	
5 18	
3 7	
34 40	



5. feladat: Munka (16 pont)

Egy vállalkozó két azonos munkagépet üzemeltet, amelyeken speciális alkatrészeket gyárt. Sok megrendelést kapott alkatrészek gyártására. A megrendelésben különböző alkatrészek szerepelnek, de ismert, hogy az egyes alkatrészek legyártása mennyi időt igényel (percben kifejezve). A gépek folyamatosan dolgoznak. A vállalkozó el akarja osztani az alkatrészeket a két gép között, hogy a lehető legkorábban befejeződjön a legyártásuk, tehát ha az első gép a neki kiosztott alkatrészeket `T1`, a második gép `T2` idő alatt gyártja le, akkor `max(T1,T2)` a lehető legkisebb legyen.

Készíts programot, amely kiszámítja, hogy legkevesebb mennyi idő alatt tudja a két gép legyártani az összes alkatrészt, és meg is ad egy megfelelő beosztást!

A `munka.be` állomány első sorában az alkatrészek száma (2≤N≤2000) van. A második sor pontosan `N` egész számot tartalmaz, a legyártandó alkatrészek gyártási idejét, ami `1` és `50` közötti érték.

A `munka.ki` állomány első sora azt a legkisebb `T` időt tartalmazza, amely alatt a két gép le tudja gyártani az összes alkatrészt! A második sor azon alkatrészek sorszámát tartalmazza (tetszőleges sorrendben, amelyeket az első, a harmadik sor pedig azokat, amelyeket a második gép gyárt le! Több megoldás esetén bármelyik megadható.

Példa:

```
munka.be                      munka.ki
9                             54
7 12 5 21 6 9 31 4 12         2 3 5 7
                              1 4 6 8 9
```

2009. Harmadik forduló

Ötödik-nyolcadik osztályosok

1. feladat: Szmogriadó (24 pont)

A városban naponta mérik a levegő szennyezettségét (szénmonoxid-tartalmát a határérték százalékában). A mérőműszer időnként elromlik, és ilyenkor 0 százalékot mér, mindaddig, amíg meg nem javítják (javítás után újra 0%-nál többet mér). Szmogriadót akkor rendelnek el, amikor a szennyezettség 100% fölé emelkedik, s a szmogriadó mindaddig tart, amíg újra 100% alatt nem lesz a szennyezettség.

Írj programot, amely beolvassa napok számát (1≤N≤100) és az egyes napokon mért szennyezettséget (`0≤szennyezettség≤500`), majd megadja

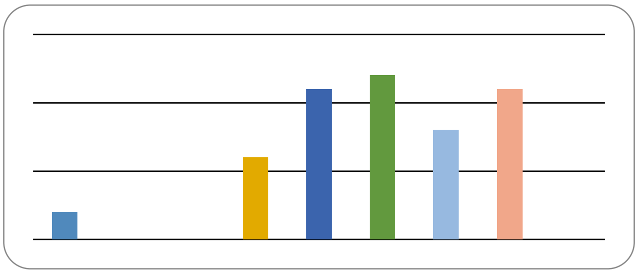
A. a legszennyezettebb napot;

B. a mérés közbeni javítások számát;

C. azon napokat, amikor szmogriadót kellett elrendelni!

Példa:

```
Bemenet:              Kimenet:
Napok száma?9         Legszennyezettebb nap: 6
1. nap?20             Javítások száma: 1
2. nap?0              Szmogriadók száma: 2
3. nap?0              Szmogriadók napjai: 5 8
4. nap?60
5. nap?110
6. nap?120
7. nap?80
8. nap?110
9. nap?0
```



2. feladat: Nevek (24 pont)

Írj programot, amely beolvassa N ember (1≤N≤20) keresztnevét, majd a képernyő K sorába (1≤K≤10) hasábokba rendezve írja ki őket (azaz a K+1. névre kezdje el a második hasábot)! Minden hasáb szélessége a benne levő leghosszabb keresztnév hossza legyen, s a hasábok között egyetlen szóköz legyen elválasztásra! Az N név biztosan elfér a képernyő K sorában.

Példa:

```
Bemenet:
Nevek száma?8
Sorok száma?3
1. név?András
2. név?Sándor
3. név?Krisztina
4. név?Éva
5. név?Csilla
6. név?Nóra
7. név?Eszter
8. név?Anikó
```

```
Kimenet:
András    Éva    Eszter
Sándor    Csilla Anikó
Krisztina Nóra
```

3. feladat: Kémek (27 pont)

Egy N tagú kémszervezetben mindenkinek több közvetlen beosztottja lehet, s mindenkinek ismerjük a közvetlen felettesét. Egyetlen tagnak nincs felettese, ő a főnök.

Írj programot, amely beolvassa a tagok számát (1≤N≤100) és a tag-párokat (kinek, ki a közvetlen főnöke), majd

A. megadja a főnököt;

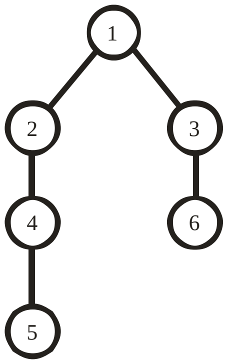
B. megad egy tagot, akinek a legtöbb közvetlen beosztottja van;

C. megad egy tagot, aki a „legmesszebb" van a főnöktől!

Példa:

```
Bemenet:
Létszám?6
1. pár beosztottja?2
Ki a felettese?1
2. pár beosztottja?3
Ki a felettese?1
3. pár beosztottja?6
Ki a felettese?3
4. pár beosztottja?5
Ki a felettese?4
5. pár beosztottja?4
Ki a felettese?2
```

```
Kimenet:
A főnök: 1
Legtöbb közvetlen beosztott: 1
Legtávolabb a főnöktől: 5
```



Kilencedik-tizedik osztályosok

1. feladat: Játék (20 pont)

Tekintsük azt az egyszemélyes játékot, amelyet M sorból és N oszlopból álló téglalap alakú táblán játszanak. Először a tábla mezőire véletlenszerűen gyöngyöket helyeznek el. A játékosnak egy bábut kell eljuttatnia a tábla (1,1) koordinátájú bal felső sarkából az (M,N) koordinátájú jobb alsó sarába. Egy lépésben a bábút szomszédos mezőre léptetheti, vagy jobbra, vagy lefelé. Miután a bábu eljutott a jobb alsó sarokba, a játékos kiválaszt két különböző mezőt, amelyeken a bábuja áthaladt (beleértve a (1,1) és (M,N) mezőket) az útvonalon, és a pontszáma a két kiválasztott mezőn lévő gyöngyök összege lesz.

Készíts programot, amely kiszámítja a játékkal elérhető legnagyobb pontszámot, és meg is ad lépéssort, amely a legnagyobb pontszámot eredményezi!

A JATEK.BE állomány első sorában a tábla mérete van (1≤M,N≤200). A következő M sor mindegyikében N nem-negatív egész szám van. Az állomány (i+1)-edik sorának j-edik száma a tábla (i,j) koordinátájú mezőjén lévő gyöngyök száma, ami nem nagyobb, mint 10 000.

A JATEK.KI állomány első sorába az elérhető legnagyobb pontszámot kell írni! A második sorba négy egész számot kell írni, a kiválasztott két mező koordinátáit, a haladás sorrendjében! A harmadik sorba pontosan M+N-2 karaktert kell írni, ami egy nyerő útvonalat ír le, amivel elérhető a maximális pontszám! A lefelé lépés jele az 'L', a jobbra lépés jele pedig a 'J' karakter. Több megoldás esetén bármelyik megadható.

Példa:

JATEK.BE	JATEK.KI
4 5	13
1 2 3 4 1	2 4 2 5
1 3 3 6 7	LJJJJLL
1 1 9 1 **1**	
1 2 2 1 **1**	

2. feladat: Kísérlet (15 pont)

Biológusok különleges sejttenyészetet vizsgálnak. A kísérlet során N sejt keletkezett. Minden sejtre feljegyezték azt az időpontot, amikor keletkezett, és amikor elpusztult.

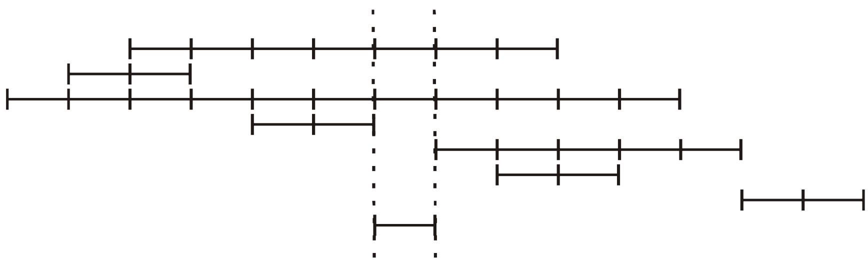
Készíts programot, amely kiszámítja azt a legrövidebb időintervallumot, amely alatt legalább K sejt megfigyelhető volt a kísérletben!

A KISERLET.BE állomány első sorában a kérdéses élő sejtek száma (1≤K≤N), a kísérlet időtartama (1≤M≤20 000) és a sejtek száma (1≤N≤50 000) van. A következő N sor mindegyikében egy sejt keletkezésének és elpusztulásának ideje van (1≤KEL≤ELP≤M) van.

A KISERLET.KI állomány első sorába az (A,B) számpárt kell írni: A B, amelyre teljesül, hogy A≤B és B-A a legkisebb olyan érték, hogy az [A,B] időintervallumban legalább K sejt megfigyelhető volt! Egy sejt megfigyelhető volt az [A,B] intervallumban, ha E keletkezési ideje E≤B és P elpusztulási ideje A≤P . Ha több ilyen A,B pár létezne, akkor azt kell kiírni, amelyiknek az A-érték a legkisebb!

Példa:

KISERLET.BE	KISERLET.KI
5 15 8	7 8
3 10	
2 4	
1 12	
5 7	
8 13	
9 11	
13 15	
7 8	



3. feladat: Rendezvény (20 pont)

Egy rendezvény szervezőinek N előadást kell elhelyezni termekben. Minden előadáshoz adott annak A kezdési és B befejezési időpontja.

Készíts programot, amely kiszámítja, hogy legkevesebb hány terem szükséges ahhoz, hogy minden előadást meg lehessen tartani!

A TERMEK.BE állomány első sorában az előadások száma van (1≤N≤200). A további N sor mindegyike két egész számot tartalmaz, az első szám az előadás kezdő ideje, a második a befejezési ideje. Az időpontok értékei nem nagyobbak, mint 500. Az előadások a kezdő időpontjuk szerint nem-csökkenő sorrendben vannak.

A TERMEK.KI állomány első sorába az előadások beosztásához minimálisan szükséges termek T számát kell írni! A további T sor azt adja meg, hogy az előadásokat mely termekbe osztottuk be! Az állomány i+1-edik sorában azoknak az előadásoknak a sorszámai vannak felsorolva időrendi sorrendben, amelyeket az i-edik terembe osztottunk be! Több megoldás esetén bármelyik megadható.

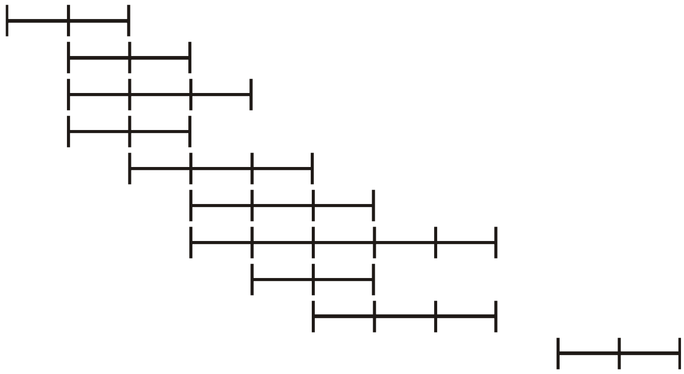
<u>Példa:</u>

TERMEK.BE

```
10
1 3
2 4
2 5
2 4
3 6
4 7
4 9
5 7
6 9
10 12
```

TERMEK.KI

```
4
1 5 9
2 6
3 8
4 7 10
```



<u>4. feladat</u>: Kémek (20 pont)

Egy kémszervezetben mindenkinek legfeljebb 2 közvetlen beosztottja lehet, s mindenkinek ismerjük a közvetlen felettesét. Egyetlen tagnak nincs felettese, ő a főnök. A szervezetbe egy ügynököt szeretnénk beépíteni, a következő két feltétellel:

- a lehető legközelebb legyen a főnökhöz,
- ha több legközelebbi hely is van, akkor azt kell választani közvetlen főnöknek, akinek a legtöbb – nem csak közvetlen – beosztottja van!

Készíts programot, amely megadja, hogy ki legyen az ügynök közvetlen főnöke!

A KEM.BE állomány első sorában a tagok száma (1≤N≤10 000) van. A következő N-1 sor mindegyike két számot tartalmaz (1≤X≠Y≤N), ami azt jelenti, hogy X-nek Y a közvetlen felettese.

A KEM.KI állomány egyetlen sorába annak a tagnak a sorszámát kell írni, aki a feltételeknek megfelelően az új ügynök közvetlen felettese lesz! Ha több megoldás is van, akkor a legkisebb sorszámút kell kiírni!

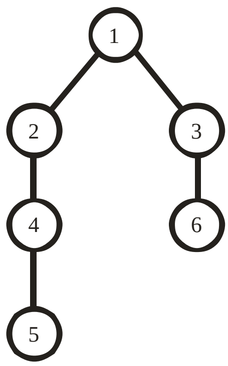
<u>Példa:</u>

KEM.BE

```
6
2 1
3 1
4 2
5 4
6 3
```

KEM.KI

```
2
```



Tizenegyedik-tizenharmadik osztályosok

1. feladat: Sejtek (15 pont)

Biológusok különleges sejttenyészetet vizsgálnak. A kísérlet során N sejt keletkezett. Minden sejtre feljegyezték azt az időpontot, amikor keletkezett, és azt, amikor elpusztult. Meg akarják határozni azt a legszűkebb [E,U] időintervallumot, melyre teljesül, hogy létezett legalább K sejt, amelyek keletkezési ideje nagyobb, vagy egyenlő, mint E, és elpusztulási ideje kisebb, vagy egyenlő, mint U.

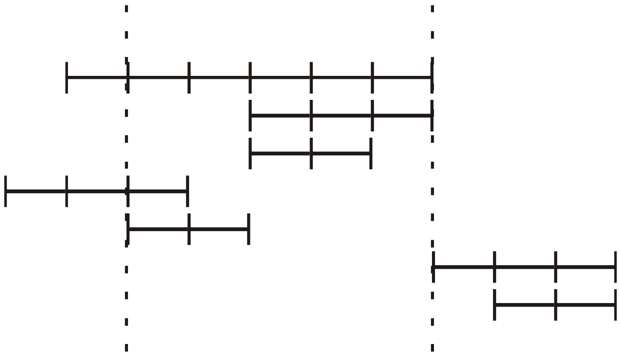
Készíts programot, amely megoldja a problémát!

A SEJTEK.BE állomány első sorában a kérdéses élő sejtek száma (1≤K≤N), a kísérlet időtartama (1≤M≤30 000), és a sejtek száma (1≤N≤50 000) van. A következő N sor mindegyikében egy sejt keletkezésének e, és elpusztulásának p ideje van (1≤e≤p≤M).

A SEJTEK.KI állomány első sorába két egész számot kell írni: E,U, amelyre teljesül, hogy U-E a legkisebb olyan érték, hogy létezik legalább K sejt, amelyek keletkezési ideje ≥E és elpusztulási ideje ≤U! Ha több ilyen E U pár létezne, akkor azt kell kiírni, amelyikre az E-érték a legkisebb!

Példa:

SEJTEK.BE	SEJTEK.KI
3 12 7	3 8
2 8	
5 8	
5 7	
1 4	
3 5	
8 11	
9 11	



2. feladat: Mérőkannák (15 pont)

Egy gazdának két kannája van, az egyik A literes, a másik pedig B. Szeretne kimérni pontosan L liter vizet. Az alábbi műveleteket lehet végezni a kimérés során:

1) Az A-literes kanna teletöltése
2) A B-literes kanna teletöltése
3) Az A-literes kanna kiürítése
4) Az B-literes kanna kiürítése
5) Áttöltés az A-literesből a B-literesbe (amíg az tele nem lesz, ill. van A-ban)
6) Áttöltés a B-literesből az A-literesbe (amíg az tele nem lesz, ill. van B-ben)

Készíts programot, amely meghatároz egy legkevesebb lépésből álló kimérés, aminek az eredményeként az A-literes kannában L liter lesz!

A KANNAK.BE állomány első sora a két kanna űrtartalmát (0<A,B≤200) tartalmazza. A második sor a kimérendő mennyiséget (0<L≤A) tartalmazza.

A KANNAK.KI állomány első sorába a kimérés lépéseinek minimális M számát kell írni! A második sor pontosan M egész számot tartalmazzon, a kimérés lépéseit! Ha nincs megoldás, akkor az első és egyetlen sor a 0 számot tartalmazza! Több megoldás esetén bármelyik megadható.

Példa:

KANNAK.BE	KANNAK.KI
9 4	8
6	1 5 4 5 4 5 1 5

3. feladat: Koncert (15 pont)

Az év koncertjére jegyet lehet igényelni. Minden igénylő pontosan két ülőhelyet jelölhet meg igényként.

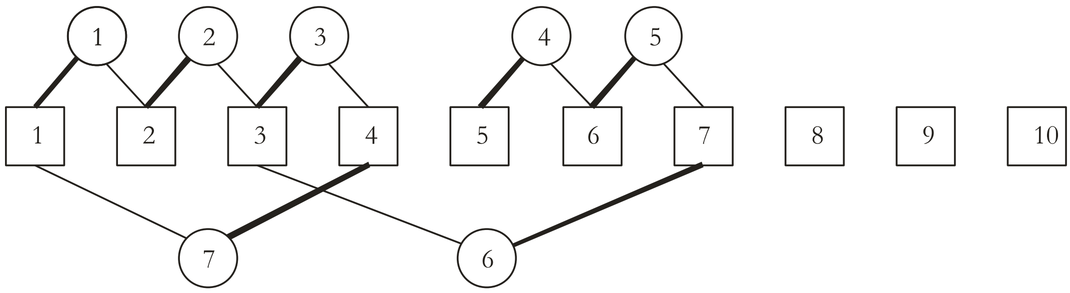
Készíts programot, amely kiszámítja, hogy legjobb esetben hány igény elégíthető ki, és meg is ad megfelelő jegykiosztást!

A KONCERT.BE állomány első sorában az ülőhelyek száma (1≤M≤30 000), és az igények száma (1≤N≤60 000) van. A további N sor mindegyike egy igényt tartalmaz (1≤u≠v≤M).

A KINCERT.KI állomány első sorába a legtöbb kielégíthető igény K számát kell írni! A következő K sorba kell kiírni egy megfelelő jegykiosztást! A sorban az első szám egy igény sorszáma, a második pedig annak a széknek a sorszáma legyen, amelyet az igénylő kap! Több megoldás esetén bármelyik megadható.

Példa:

KONCERT.BE	KONCERT.KI
10 7	7
1 2	1 1
2 3	2 2
3 4	3 3
5 6	7 4
6 7	4 5
3 7	5 6
1 4	6 7



4. feladat: Ügynökök (15 pont)

Egy kémszervezetben mindenkinek több beosztottja lehet, s mindenkinek ismerjük a közvetlen felettesét. Egyetlen tagnak nincs felettese, ő a nagyfőnök. A szervezetben minimális számú ügynököt szeretnénk lecserélni úgy, a következő két feltétel teljesüljön:

- olyan tag cserélhető le, akinek legfeljebb 2 közvetlen beosztottja van,
- a lecserélt ügynökök a szervezet legalább 50%-ának legyenek közvetett vagy közvetlen főnökei, saját magukat is beleértve (azaz egy lecserélt ügynök beosztottjait már nem érdemes lecserélni)!

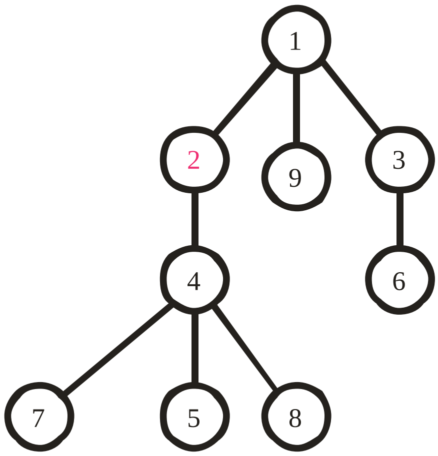
Készíts programot, amely megadja, hogy hány ügynököt kell lecserélni!

A UGYNOK.BE állomány első sorában a tagok száma (1≤N≤10 000) van. A következő N-1 sor mindegyike két számot tartalmaz (1≤X≠Y≤N) egy szóközzel elválasztva, ami azt jelenti, hogy X-nek Y a közvetlen felettese.

A UGYNOK.KI állomány egyetlen sorába a lecserélendő ügynökök minimális számát kell írni!

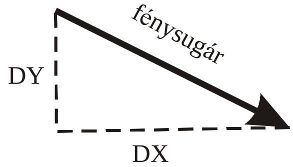
Példa:

```
UGYNOK.BE          UGYNOK.KI
9                  1
7 4
8 4
9 1
2 1
3 1
4 2
5 4
6 3
```



5. feladat: Házak (15 pont)

Egy koordinátarendszerben az x-tengely mentén téglalap alakú házak helyezkednek el. A Nap balról, fentről süt rájuk, a házakhoz párhuzamos fénysugarak érkeznek. A fénysugarak irányát a mellékelt ábra szerint adjuk meg.



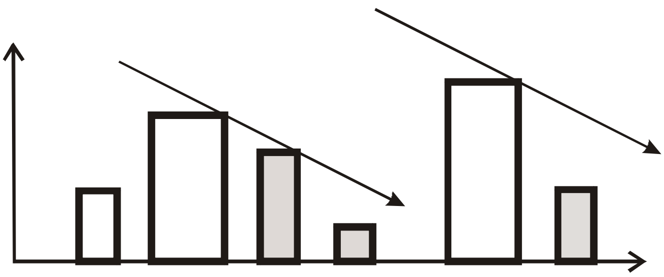
Készíts programot, amely megadja azokat a házakat, amelyeknek legalább egyetlen pontjára süt a nap!

A HAZAK.BE állomány első sorában a házak száma (1≤N≤10 000), valamint a napfény iránya (0<DX, DY) van. A következő N sorban az egyes épületek bal alsó sarkának x-koordinátája (0<x≤100 000), szélessége (0<szélesség≤100) és magassága (0<magasság≤1000) van (a bal alsó sarok y-koordinátája biztosan 0). Az épületek x-koordináta szerint növekvő sorrendben vannak.

A HAZAK.KI állomány első sorába a napsütötte házak számát, a második sorába pedig ezen házak sorszámát kell írni, növekvő sorrendben!

Példa:

```
HAZAK.BE           HAZAK.KI
6 2 1              3
10 5 10            1 2 5
20 10 20
35 5 15
45 5 5
60 10 25
75 5 10
```



A verseny végeredménye:

I. korcsoport

1	Weisz Gellért	Fazekas Mihály Gimnázium, Budapest
2	Varnyú József	Fazekas Mihály Gimnázium, Debrecen
	Leitereg András	Barcsay Jenő Általános Iskola, Szentendre
4	Simig Dániel	Dobó Katalin Gimnázium, Esztergom
	Szarka Gábor	Szent István Gimnázium, Budapest
6	Kiss Anna	Bolyai Farkas Elméleti Líceum, Marosvásárhely
7	Győri Mihály	Fazekas Mihály Gimnázium, Debrecen

8	Dankovics Viktor	Veres Péter Gimnázium, Budapest
	Palkó András	Vörösmarty Mihály Gimnázium, Szentgotthárd
10	Ruszki Bálint	Károlyi István 12 évfolyamos Gimnázium, Budapest
II. korcsoport		
1	Adrián Patrik	Fazekas Mihály Gimnázium, Debrecen
2	Weisz Ágoston	Fazekas Mihály Gimnázium, Budapest
3	Dankovics Attila	Veres Péter Gimnázium, Budapest
4	Danyluk Tamás	Földes Ferenc Gimnázium, Miskolc
5	Nagy Dániel	Szent István Gimnázium, Budapest
	Kovács Zsombor	Fazekas Mihály Gimnázium, Budapest
	Erdős Gergely	Batthyány Lajos Gimnázium és Szakközépiskola, Nagykanizsa
8	Fehér Péter	Batthyány Lajos Gimnázium és Szakközépiskola, Nagykanizsa
9	Kovács Gábor Ferenc	Árpád Gimnázium, Tatabánya
	Gergely Dániel	Balassi Bálint Nyolcévfolyamos Gimnázium, Budapest
	Sulyok András Attila	Református Gimnázium, Szentendre
III. korcsoport		
1	Danka Miklós András	Fazekas Mihály Gimnázium, Budapest
2	Danner Gábor	Ságvári Endre Gimnázium, Szeged
3	Hadházi Dániel	Neumann János Középiskola és Kollégium, Eger
4	Gévay Gábor	Ságvári Endre Gimnázium, Szeged
5	Varga László	Fazekas Mihály Gimnázium, Debrecen
6	Mészáros András	Révai Miklós Gimnázium, Győr
7	Éles András	Fazekas Mihály Gimnázium, Debrecen
8	Badics Alex	Eötvös József Gimnázium, Tata
9	Kalló Bernát	Fazekas Mihály Gimnázium, Budapest
10	Grósz Dániel	Fazekas Mihály Gimnázium, Budapest
11	Csikota Balázs	Szent Benedek Általános Iskola és Gimnázium, Budapest
12	Nádor István	Babits Mihály Gimnázium és Szakközépiskola, Pécs
13	Hunyady Márton	Bencés Gimnázium, Pannonhalma
14	Fábián András	Radnóti Miklós Gimnázium, Szeged
15	Wagner Zsolt	Fazekas Mihály Gimnázium, Budapest

II.

Megoldások, értékelések

Nemes Tihamér
Nemzetközi Informatikai Tanulmányi Verseny

2005. Első forduló

Ötödik-nyolcadik osztályosok

1. feladat: Képletek (27 pont)

A. Első: X=3, Y=2 — 2+2 pont
Második: X=-3, Y=-2 — 2+2 pont
Harmadik: X=6, Y=4 — 2+2 pont

B. tetszőleges X=Y példa helyes — 2 pont
X=0, Y=0 vagy tetszőleges X=-Y példa helyes — 2 pont
X=0, Y=0 — 2 pont

C. (X,Y) új értéke:=(Y,X) — 3 pont
(X,Y) új értéke:=(-Y,-X) — 3 pont
(X,Y) új értéke:=(2*Y,2*X) — 3 pont

2. feladat: Számkitaláló (24 pont)

A. Alfa: 1,8,2,6 — 1+1+1+1 pont
Béta: 1,1,2,41+1+1+1 pont
Gamma: 1,8,1,4 — 1+1+1+1 pont

B. Alfa: az N számjegyei összege — 4 pont
Béta: az N számjegyei száma — 4 pont
Gamma: az N legnagyobb értékű számjegye — 4 pont

3. feladat: Átrendezés (24 pont)

A D részfeladatra a leírttal azonos értelmű megfogalmazások is elfogadhatók.

A. X=(4,1,2,3,7,5,6) — 7*1 pont
B. X=(1,2,3,4,5,6,7) — 7*1 pont
C. {*}: Y-nál nagyobb értékű elemet keres — 2 pont
{**}: Y-nál kisebb értékű elemet keres — 2 pont
D. Az Y-nal egyenlő elemek a helyükön maradnak — 2 pont
Az Y-nál kisebbek mindegyike az Y-nál nagyobbak elé kerül — 2 pont
(hátulról előre teszi az Y-nál kisebbeket, elölről hátra teszi az Y-nál nagyobbakat)
Az elmozdított elemek sorrendje megfordul a korábbihoz képest — 2 pont
(azaz pl. a B kérdés esetén a 7 megelőzte az 5-öt, mindkettő hátrább került és a sorrendjük megfordult, …)

4. feladat: Robot (25 pont)

0	1	1	2	7	7
0	1	2	**0**	**0**	**0**
0	1	3	3	3	5
2	**0**	3	3	4	5
2	3	4	4	4	5

Helyes számonként 1-1 pont
A vastagon szedett 0-kért (R vagy X helyére kerülnek) nem jár pont.

Kilencedik-tizedik osztályosok

1. feladat: Mátrixelem (20 pont)

A. ha az elem nagyobb vagy egyenlő minden szomszédjánál 4 pont

B. első: i=7, j=3 2 pont

második: i=5, j=7 2 pont

C. első: mindig a legnagyobb növekedés irányába lép 4 pont

második: először jobbra, majd lefelé, majd balra, végül felfelé próbálkozik, s az első lehetséges irányba lép, amerre az érték növekszik 4 pont

D. Ne legyen a maximumom kívül olyan hely, ahonnan nem lehet tovább magasabb szomszéd felé lépni. 4 pont

2. feladat: Keresés (18 pont)

A. Első: S=5 2 pont

Második: S=4 2 pont

Harmadik: S=5 2 pont

B. Első: T-ben legalább egy 0 van 2 pont

Második: T-ben legalább kettő 0 van 2 pont

Harmadik: T-ben legalább K darab 0 van 2 pont

C. S:=T utolsó 0 értékű eleme sorszáma 2 pont

S:=T második 0 értékű eleme sorszáma 2 pont

S:=T K-adik 0 értékű eleme sorszáma 2 pont

3. feladat: Lista (22 pont)

Beszúr(Albert): FEJ=5 1 pont
ÜRES=9 1 pont
5. sor=Alajos,7 2 pont
7. sor=Albert,8 2 pont

Töröl(Zoli): FEJ=5 1 pont
ÜRES=10 1 pont
6. sor=Pista,0 2 pont
10. sor=Zoli,9 2 pont

Beszúr(Demeter): FEJ=5 1 pont
ÜRES=9 1 pont
8. sor=Barnabás,10 2 pont
10. sor=Demeter,4 2 pont

Beszúr(Aladár): FEJ=9 1 pont
ÜRES=1 1 pont
9. sor=Aladár,5 2 pont

4. feladat: Újság (18 pont)

Az elérhető összeg egyértelmű, az alább leírttól különböző hirdetésekkel is elérhető, így minden olyan megoldás elfogadható, amelyben ugyanannyi forint jön ki, s mindegyik hirdetés jó napon jelenik meg.

A. 4800 forint 2 pont

Hirdetés sorszámok: 6, 5, 9, 4, 1, 3, - 4 pont

B. 6000 forint 2 pont

Hirdetés sorszámok: 2, 1, 5, 6, 9, 8, 7 4 pont

C. 4200 forint 2 pont

Hirdetés sorszámok: 7, 5, 9, 8, 10, 3, 6 4 pont

5. feladat: Kocka (22 pont)

Értékelés:

M(1)	M(2)	M(3)	M(4)	M(5)	M(6)	M(7)	M(8)	M(9)	M(10)
10	17	23	27	11	30	19	14	28	23

10*2 pont

A legmagasabb építhető torony magassága: 30 2 pont

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Balaton (16 pont)

A. első: i=2, j=11 2 pont

második: i=6, j=2 2 pont

harmadik: i=9, j=5 2 pont

negyedik: i=6, j=13 2 pont

B. első: a legfelső 0-k közül a legbaloldalibb (~legészakibb pont) 1+1 pont

második: a legbaloldalibb 0-k közül a legfelső (~legnyugatibb pont) 1+1 pont

harmadik: a legalsó 0-k közül a legjobboldalibb (~legdélibb pont) 1+1 pont

negyedik:a legjobboldalibb 0-k közül a legalsó (~legkeletibb pont) 1+1 pont

2. feladat: Mit csinál? (20 pont)

A. 1,2,3,4,5,6,7 2 pont

B. 1,2,3,4,5,6,7 2 pont

C. Nagyság szerint növekvő sorrendbe rendezi a beolvasott elemeket 4 pont

D. Helyesen működik, ha bármely elemre igaz, hogy a bemenetben legfeljebb K-1 darab nála nagyobb elem volt előtte 4 pont

E. Az első K elemet beolvassa és sorba rendezi 2 pont

F. Az összes többi elemre: a tárolt K elemből a legkisebbet kiírja 2 pont
beolvassa a következő elemet 1 pont
majd azt beilleszti a nagyság szerinti helyére 2 pont

G. Az utoljára maradt K elemet kiírja 1 pont

3. feladat: Lista (24 pont)

Beszúr(Albert): FEJ=5 2 pont
ÜRES=9 1 pont
5. sor=Alajos,7 2 pont
7. sor=Albert,8 2 pont

Töröl(Zoli): FEJ=5 2 pont
ÜRES=10 1 pont
6. sor=Pista,0 2 pont
10. sor=Zoli,9 2 pont

Töröl(Alajos): FEJ=7 2 pont
ÜRES=5 1 pont
5. sor=Alajos,10 2 pont

Beszúr(Aladár): FEJ=5 2 pont
ÜRES=10 1 pont
5. sor=Aladár,7 2 pont

<u>4. feladat</u>: Újság (18 pont)

Az elérhető összeg egyértelmű, az alább leírttól különböző hirdetésekkel is elérhető, így minden olyan megoldás elfogadható, amelyben ugyanannyi forint jön ki, s mindegyik hirdetés jó napon jelenik meg.

A. 4800 forint 2 pont

Hirdetés sorszámok: 1: 5. nap, 3: 6. nap, 4: 4. nap, 5: 2. nap, 6: 1. nap, 9: 3. nap 4 pont

B. 6000 forint 2 pont

Hirdetés sorszámok: 1: 2. nap, 2: 1. nap, 5: 3. nap, 6: 4. nap, 7: 7. nap, 8: 6. nap, 9: 5. nap 4 pont

C. 4200 forint 2 pont

Hirdetés sorszámok: 3: 6. nap, 5: 2. nap, 6: 7. nap, 7: 1. nap, 8: 4. nap, 9: 3. nap, 10: 5. nap 4 pont

<u>5. feladat</u>: Kocka (22 pont)

Értékelés:

M(1)	M(2)	M(3)	M(4)	M(5)	M(6)	M(7)	M(8)	M(9)	M(10)
1	2	3	4	1	5	2	1	4	2

10*2 pont

A legtöbb kockából álló torony kockaszáma: 5 2 pont

2005. Második forduló

Ötödik-nyolcadik osztályosok

<u>1. feladat</u>: Számok franciául (27 pont)

Tároljuk egy-egy tömbben az egyes és a tízes helyiértékre írandó szövegeket! Ezután a képzési szabályok szerinti elágazással megoldható a feladat.

```
egyes: tömb(0..16,szöveg)=('z,ro','un','deux','trois',
'quatre','cinq','six','sept','huit','neuf','dix','onze', 'dou-
ze','treize','quatorze','quinze','seize')
```

```
tizes: tömb(1..9,szöveg)=('dix','vingt','trente', 'quaran-
te','cinquante','soixante','soixante',
'quatre-vingt','quatre-vingt')
```

```
Kiír(N):
  Ha n<17 akkor Ki: egyes(n)
  különben ha n<20 akkor Ki: tizes(1),'-',egyes(e)
  különben ha e=0 akkor
      Ki: tizes(t)
      Ha t=7 vagy t=9 akkor Ki: '-dix'
      különben ha t=8 akkor Ki: 's'
  különben ha n<70 akkor
      Ki: tizes(t)
      Ha e=1 akkor Ki: ' et un' különben Ki: '-',egyes(e)
  különben ha n∈(71..76,91..96) akkor
      Ki: tizes(t),'-',egyes(10+e))
  különben ha n∈ (81..89) akkor Ki: tizes(t),'-',egyes(e)
  különben {n∈(77..79,97..99)}
      Ki: tizes(t),'-',tizes(1),'-',egyes(e)
Eljárás vége.
```

2. feladat: Születésnap (28 pont)

Tároljuk az egyes hónapok napszámát egy konstans tömbben! Számítsuk át a születésnapokat éven belüli sorszámra! A megoldásban ügyelni kell arra is, hogy a legközelebbi születésnapos lehet az aktuális évben, de lehet a következő évben is!

```
hó: tömb(0..12,egész)=(0,31,28,31,30,31,30,31,31,30,31,30,30)

Születésnap(N,sz,li,lj):
  Ciklus i=1-től 12-ig
    hó(i):=hó(i)+hó(i-1)
  Ciklus vége
  Ciklus i=1-től N-ig
    sz(i).nap:=sz(i).nap+hó(sz(i).hó-1)
  Ciklus vége

  tav:=maxint; li:=0; lj:=0
  Ciklus i=1-től N-1-ig
    Ciklus j=i+1-től N-ig
      Ha |sz(i).nap-sz(j).nap|<183
          akkor kul:=|sz(i).nap-sz(j).nap|
       különben kul:=365-|sz(i).nap-sz(j).nap|
      Ha kul<tav akkor tav:=kul; li:=i; lj:=j
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

3. feladat: Szövegjavítás (20 pont)

Le kell másolni a bemenő szöveget a következő szabályokkal:

- Az írásjel előtti szóközöket ki kell szedni az eredményből!
- Írásjel után kell tenni pontosan egy szóközt!
- Egymás mögé egynél több szóközt nem szabad tenni!

```
Írásjelek:=(',','.',';','!','?',':')
```

```
Javítás(a,b):
  b:=a(1); i:=2
  Ciklus amíg i≤hossz(a)
    Ha a(i)∈Írásjelek akkor
        Ciklus amíg b(hossz(b))=' '
          b:=utolsónélküli(b)
        Ciklus vége
        b:=b+a(i); i:=i+1
        Ciklus amíg i≤hossz(a) és a(i)=' '
          i:=i+1
        Ciklus vége
        b:=b+' '
    különben ha a(i)≠' ' vagy b(hossz(b))≠' ' akkor
        b:=b+a(i); i:=i+1
    különben i:=i+1
  Ciklus vége
Eljárás vége.
```

Kilencedik-tizedik osztályosok

1. feladat: Gazda (10 pont)

Mivel a nyárfák között lehet szilvafa, ezért minden úton eggyel kevesebb szilvafa lehet, mint ahány nyárfa van. Emiatt az a célszerű, ha minél hosszabb utakon választunk egymás melletti szilvafákat.

Rendezzük tehát hossz szerint csökkenő sorrendbe az utakat, majd eszerint sorrendben válasszunk nyárfákat!

```
Gazda(N,K,t,db):
  Ciklus i=2-től K-ig
    j:=i-1; x:=t(i)
    Ciklus amíg j>0 és x>t(j)
      t(j+1):=t(j); j:=j-1
    Ciklus vége
    t(j+1):=x
  Ciklus vége
  db:=0; i:=1
  Ciklus amíg N>0
    Ha n≥t(i) akkor db:=db+t(i)-1; n:=n-t(i)
           különben db:=db+n-1; n:=0
    i:=i+1
  Ciklus vége
Eljárás vége.
```

2. feladat: Javít (20 pont)

Le kell másolni a bemenő szöveget a következő szabályokkal:

- Az írásjel előtti szóközöket ki kell szedni az eredményből!
- Írásjel után kell tenni pontosan egy szóközt!
- Nyitózárójel elé kell, mögé nem kell szóköz!
- Csukózárójel elé nem kell, mögé kell szóköz!
- Csukózárójel és írásjel közé nem szabad tenni szóközt, írásjel után sem szabad, ha azt közvetlenül csukózárójel követi!
- Két nyitó- és két csukózárójel közé sem szabad szóközt tenni!

```
Írásjelek:=(',','.',';','!','?',':')
Nyitók:=('(','[','{')
Csukók:=(')',']','}')
```

```
Feldolgozás(f):
  Olvas(f,a); c:=''
  Ciklus amíg nem vége?(f)
    Ha a∈Írásjelek vagy a∈Csukók akkor
        Ír(g,a); b:=a; c:=''; Olvas(f,a)
        Ciklus amíg nem vége?(f) és a=' '
          Olvas(f,a)
        Ciklus vége
        Ha a∉Csukók és a∉Írásjelek akkor Ír(g,' ')
    különben ha a∈Nyitók akkor
        Ha b∉Nyitók akkor Ír(g,' ')
        Ír(g,a); b:=a; c:=''; Olvas(f,a)
        Ciklus amíg nem vége?(f) és a=' '
          Olvas(f,a)
        Ciklus vége
    különben ha a=' ' akkor c:=c+' '; Olvas(f,a)
    különben Ír(g,c+a); b:=a; Olvas(f,a); c:=''
  Ciklus vége
Eljárás vége.
```

3. feladat: Pakolás (15 pont)

Legyen az `A()` vektorban a konténerek elhelyezése, `A(i)=1`, ha az `i`-edik pozíción van konténer; illetve 0, ha nincs!

Az i. pozíciótól balra `Balról(i)`, jobbra `Jobbról(i)` konténer van, az i. pozícióhoz balról igazítás költsége legyen `BZár(i)`, a jobbról igazításé pedig `JZár(i)` (de a megoldásban a `JZár` tömb helyett elég egyetlen változót használni!

$$Balról(i)=\begin{cases} 0 & ha\ \ i=0 \\ Balról(i-1)+A(i) & ha\ \ i>0 \end{cases}$$

$$Jobbról(i)=\begin{cases} 0 & ha\ \ i=N+1 \\ Jobbról(i+1)+A(i) & ha\ \ i<N+1 \end{cases}$$

$$Bzár(i)=\begin{cases} Bzár(i-1) & ha\ \ A(i)=1 \\ Bzár(i-1)+Balról(i-1) & ha\ \ A(i)=0 \end{cases}$$

$$Jzár(i)=\begin{cases} Jzár(i+1) & ha\ \ A(i)=1 \\ Jzár(i+1)+Jobbról(i+1) & ha\ \ A(i)=0 \end{cases}$$

A megoldás a `BZár(i)+JZár(i)` értékek minimuma lesz, erre a pozícióra annyi elemet kell balról elmozgatni, ahányan tőle balra vannak.

```
Pakolás(N,A,Opti,Kezd):
  A(0):=0; A(N+1):=0; Balról(0):=0; Jobbról(N+1):=0
  BZár(0):=0; JZár:=0
  Ciklus i=1-től N-ig
    Balról(i):=Balról(i-1)+A(i)
    Ha A(i)=1 akkor BZár(i):=BZár(i-1)
           különben BZár(i):=BZár(i-1)+Balról(i-1)
  Ciklus vége
  Opti:=N*N; JZár:=0
  Ciklus i=N-től 1-ig -1-esével
    Jobbról(i):=Jobbról(i+1)+A(i)
    Ha BZár(i)+JZár<Opti
       akkor Opti:=BZár(i)+JZár; Kezd:=i-Balról(i)+1
    Ha A(i)=1 akkor JZár:=JZár
           különben JZár:=JZár+Jobbról(i+1)
  Ciklus vége
Eljárás vége.
```

<u>4. feladat</u>: Ajándék (15 pont)

Készítsünk egy gráfot, amelyben az `i` és a `j` pont között akkor van él, ha az `i`-edik tanuló a `j`-ediknek adna ajándékot! Ekkor a problémásak a gráf azon pontjai, ahova nem vezet be él (azaz az adott tanuló nem kapna senkitől sem ajándékot).

Nézzük meg minden ilyen tanulóra, hogy ő kinek adna, aki tőle kapja, az kinek adja, … egészen addig, amíg olyan tanulóhoz nem érünk, aki már kap valaki mástól! Aki így már másodiknak adná neki az ajándékot, azzal adassuk oda a kiindulópontnak! Ezzel a módszerrel a gráfban köröket alakítunk ki, azaz mindenki fog ajándékot kapni.

```
Ajándék(N,F,MF)
  Szín():=(fehér,…,fehér); MF():=(0,…,0)
  Módosítás:=0; Eleje:=0; Vége:=0
  Ciklus x=1-től N-ig
    Ha Szín(x)=fehér és BeFok(x)=0
      akkor Ha Eleje=0 akkor Eleje:=x
            Módosítás:=Módosítás+1; xx:=x; Szín(xx):=fekete
            Ciklus amíg Szín(F(xx))=fehér
              xx:=F(xx); Szín(xx):=fekete
            Ciklus vége

            Ha Vége=0 akkor Vége:=xx
            MF(xx):=Eleje; Eleje:=x
  Ciklus vége
  Ha Eleje≠0 akkor MF(Vége):=Eleje
Eljárás vége.
```

<u>5. feladat</u>: Vállalat (15 pont)

A feladat szerint egy bináris fában kell meghatározni a gyökértől pontosan `K` távolságra levő pontokat. Az alábbi rekurzív eljárás a globális `Fa` tömb alapján a globális `Db` változóba és `Szint` tömbbe teszi az eredményt.

```
Bejár(Fapont,K):
  Ha FaPont≠0 akkor
    Ha k=0 akkor Db:=Db+1; Szint(Db):=Fapont
        különben Bejár(Fa(Fapont).bal,K-1)
                 Bejár(Fa(Fapont).jobb,K-1)
Eljárás vége.
```

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Fák (14 pont)

A fákat feleltessük meg egy-egy intervallumnak! Minden fára az intervallum kezdőpontja legyen a fa helye (`t(i)`), a végpontja pedig a fa helyéhez hozzáadva a magassága (`t(i)+m(i)`)!

Ezután az első részfeladat annak meghatározása (`db`), hogy az így kapott intervallumok uniója hány összefüggő intervallumot eredményez. A második részfeladat pedig annak az összefüggő intervallumhalmaznak a kezdőpont meghatározása, amelyben a legtöbb intervallum van (`max`).

A megoldásban a `t(mettől)` és a `meddig` változók azt jelölik, hogy az aktuális összefüggő intervallumhalmaz mettől meddig tart, `max` a legnagyobb elemszámúnak a kezdőpontja, `maxe` pedig az elemszáma.

```
Fák(N,t,n,mettől,meddig,max):
  db:=1; maxe:=1; max:=1; hány:=1; mettől:=1; meddig:=m(1)
  Ciklus i=2-től N-ig
    Ha meddig≤t(i)       {nem ér el a következőig}
       akkor db:=db+1
             Ha hány>maxe akkor maxe:=hány; max:=mettől
             hány:=1; meddig:=t(i)+m(i); mettől:=i
    különben       {elér a következőig}
             Ha meddig<t(i)+m(i) akkor meddig:=t(i)+m(i)
             hány:=hány+1
  Ciklus vége
  Ha hány>maxe akkor max:=mettől
Eljárás vége.
```

2. feladat: Pince (16 pont)

A feladat megoldása a fúrás szimulálása a parancstömb (`p`) alapján, az elért pozíciók (`x,y`) rögzítése egy mátrixban (`t`). Itt vizsgálható a szabálytalan fúrás esete. A megoldandó probléma a mellékágak kezelése: ezekre rekurziót kell alkalmazni!

```
dx: tömb(0..3,egész)=(0,1,0,-1)
dy: tömb(0..3,egész)=(-1,0,1,0)

Pince(t,ered):
  ered:=0; ág:=0; t(,):=(0,…,0)
  t(x,y):=1; t(x,y-1):=1; Fúrás(2,x,y-1,0,'E')
Eljárás vége.

Darab(x,y,db):
  db:=0
  Ciklus i=x-1-től x+1-ig
    Ha i+max∈{0..2*max}
       akkor Ciklus j=y-1-től y+1-ig
               Ha j+max∈{0..2*max} akkor db:=db+t(i,j)
             Ciklus vége
  Ciklus vége
Eljárás vége.
```

```
Fúrás(i,x,y,irány,e):
  Ha p(i)='E'
  akkor Darab(x+dx(irány),y+dy(irány),db)
        t(x+dx(irány),y+dy(irány)):=1
        Ha ág>0 és db=ág vagy e='E' és db=1 vagy e≠'E' és db=2
          akkor ág:=0          {nincs ütközés}
                Fúrás(i+1,x+dx(irány),y+dy(irány),irány,p(i))
        különben ered:=i          {van ütközés}
  különben ha p(i)='J'          {helyben marad, irányváltás}
  akkor Fúrás(i+1,x,y,(irány+1) mod 4,p(i))
  különben ha p(i)='B'          {helyben marad, irányváltás}
  akkor Ha irány=0 akkor irány:=3 különben irány:=irány-1
        Fúrás(i+1,x,y,irány,p(i))
  különben ha p(i)='('
  akkor db:=1; j:=i+1
        Ciklus amíg db>0          {elmegy az ág végéig}
          Ha p(j)='(' akkor db:=db+1
          különben ha p(j)=')' akkor db:=db-1
          j:=j+1
        Ciklus vége
        Fúrás(j,x,y,irány,e)      {főág}
        Ha e='E' és p(j)='E' akkor ág:=3 különben ág:=2
        Fúrás(i+1,x,y,irány,e)    {mellékág}
Eljárás vége.
```

3. feladat: Osztály (15 pont)

A feladat egy irányított gráfban (a tanulók a gráf pontjai, a telefonos kapcsolatok pedig a gráf élei) megtalálni azt a pontot, ahonnan a legtöbb további pont elérhető. Ehhez minden pontból be kell járni a gráfot (pl. szélességi bejárással) és számolni, hogy az adott pontból hány másik pont érhető el.

```
Osztály(N,Cs,maxdb,max):
  maxdb:=0; max:=0
  Ciklus i=1-től N-ig
    Bejár(i,db)
    Ha db>maxdb akkor maxdb:=db; max:=i
  Ciklus vége
Eljárás vége.
```

```
Bejár(k,db):
  h:={}; Üres(S); Sorba(k); h:={k}; db:=1
  Ciklus amíg nem üres?(S)
    Sorból(i)
    Ciklus j=1-től n-ig
      Ha j∉h és Cs(i,j) akkor Sorba(j); h:=h∪{j}; db:=db+1
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

4. feladat: Vonat (15 pont)

Nézzük a városokat jobbról balra haladva! Számoljuk ki minden u városra, hogy onnan mennyi a maximálisan bejárható városok száma az utolsó városig (`Hossz(u)`) és melyikbe kell menni belőle, hogy ezeket be is járjuk (`Út(u)`)!

```
Vonat(N,M,Járat,ÚtHossz,Út):
  Hossz(N):=1; Út():=(0,…,0)
  Ciklus u=N-1-től 1-ig -1-esével
    Max:=0
    Ciklus i=1-től M-ig
      Ha Járat(i).tól=u
        akkor v:=Járat(i).ig
              Ha Hossz(v)>Max akkor Max:=Hossz(v); Út(u):=i
    Ciklus vége
    Hossz(u):=Max+1
  Ciklus vége
  ÚtHossz:=Hossz(1)
Eljárás vége.
```

<u>5. feladat</u>: Játék (15 pont)

A feladat megoldható a következő gráf-modellben. A gráf pontjai legyenek a lehetséges játékállások, azaz (p1,p2,p3) hármasok, a három korong helye a táblán. A helyet x-és y-koordinátájávan adjuk meg. Vigyázni kell, mert a korongok sorrendje nem számít, tehát a (p2,p1,p3) hármas ugyanazt a játákállást azonosítja. A gráfban a (p1,p2,p3) pontból akkor és csak akkor van irányított él a (q1,q2,q3) pontba, ha az utóbbi egyetlen lépéssel kapható az előbbiből. Tehát a megoldás a kezdő játékállásból a végállásba vezető legrövidebb út hossza, amit szélességi bejárással számíthatunk ki.

```
Szamit:
  Ciklus x1=1-től 6-ig
    Ciklus y1=1-től 6-ig
      Ciklus x2=1-től 6-ig
        Ciklus y2=1-től 6-ig
          Ciklus x3=1-től 6-ig
            Ciklus y3=1-től 6-ig
              Tav[x1,y1,x2,y2,x3,y3]:=Inf
  Ciklusok vége
  OK:=hamis; Letesit(S); Rend(P0);   Rend(Q0)
  Tav[p0[1].x,p0[1].y,p0[2].x,p0[2].y,p0[3].x,p0[3].y]:=0
  Ha (p0[1].x=q0[1].x) és (p0[1].y=q0[1].y) és
     (p0[2].x=q0[2].x) és (p0[2].y=q0[2].y) és
     (p0[3].x=q0[3].x) és (p0[3].y=q0[3].y) akkor OK:=igaz
  különben
    Sorba(S,p0)
    Ciklus amíg Elemszam(S)>0
      Sorból(S,p)
      Ciklus i=1-től 3-ig
       Ciklus L=jobbra-tól felfele-ig
        Ha Lep(p,i,L,q) akkor
           Tav[q[1].x,q[1].y,q[2].x,q[2].y,q[3].x,q[3].y]:=
           Tav[p[1].x,p[1].y,p[2].x,p[2].y,p[3].x,p[3].y]+1
           Ha (q[1].x=q0[1].x) és (q[1].y=q0[1].y) és
             (q[2].x=q0[2].x) és (q[2].y=q0[2].y) és
             (q[3].x=q0[3].x) és (q[3].y=q0[3].y) akkor
                                                  OK:=igaz; Break
        Elágazás vége
        Sorba(S,q)
      Elágazás vége
    Ciklus vége
    Ha OK akkor Break
  Ciklus vége
Eljárás vége
```

```
Lep(p,i,L,q):
    Lep:=hamis; q:=p
    Ha L=jobbra akkor {->}
      q[i].y:=p[i].y+1
      Ha (q[i].y<7) akkor
        Rend(q)
        Ha nem Ketto(q) akkor Lep:=Tav[q[1],q[2],q[3]]=Inf
        különben
          q:=p; q[i].y:=p[i].y+2
          Ha (q[i].y<7) akkor
            Rend(q)
            Ha nem Ketto(q) akkor Lep:=Tav[q[1],q[2],q[3]]=Inf
    különben ha L=lefele akkor
      q[i].x:=p[i].x+1
      Ha (q[i].x<7) akkor
        Rend(q)
        Ha nem Ketto(q) akkor Lep:=Tav[q[1],q[2],q[3]]=Inf
        különben
          q:=p; q[i].x:=p[i].x+2
          Ha (q[i].x<7) akkor Exit
            Rend(q)
            Ha nem Ketto(q) akkor Lep:=Tav[q[1],q[2],q[3]]=Inf
      Elágazás vége
    különben ha L=felfele akkor
      q[i].x:=p[i].x-1
      Ha (q[i].x>0) akkor
        Rend(q)
        Ha nem Ketto(q) akkor Lep:=Tav[q[1],q[2],q[3]]=Inf
        különben
          q:=p; q[i].x:=p[i].x-2
        Ha (q[i].x>0) akkor
          Rend(q)
          Ha nem Ketto(q) akkor Lep:=Tav[q[1],q[2],q[3]]=Inf
      Elágazás vége
    különben ha L=balra akkor
      q[i].y:=p[i].y-1
      Ha (q[i].y>0) akkor
        Rend(q)
        Ha nem Ketto(q) akkor Lep:=Tav[q[1],q[2],q[3]]=Inf
        különben
          q:=p; q[i].y:=p[i].y-2
          Ha (q[i].y>0) akkor
            Rend(q)
            Ha Ketto(q) akkor Lep:=Tav[q[1],q[2],q[3]]=Inf
      Elágazás vége
  Elágazás vége
Függvény vége.
```

```
Rend(Var P:Konfig):
  Ha (p[2].x<p[1].x) vagy
     (p[2].x=p[1].x) és (p[2].y<p[1].y) akkor
    pp:=p[1]; p[1]:=p[2]; p[2]:=pp
  Elágazás vége
  Ha (p[3].x<p[1].x) vagy
     (p[3].x=p[1].x) és (p[3].y<p[1].y) akkor
    pp:=p[3]; p[3]:=p[2]; p[2]:=p[1]; p[1]:=pp
  Egyébként ha (p[3].x<p[2].x) vagy
               (p[3].x=p[2].x) és (p[3].y<p[2].y) akkor
    pp:=p[3]; p[3]:=p[2]; p[2]:=pp
  Elágazás vége
Eljárás vége.
```

```
Ketto(const P:Konfig):
  Ketto:=(p[1].x=p[2].x) és (p[1].y=p[2].y) vagy
         (p[1].x=p[3].x) és (p[1].y=p[3].y) vagy
         (p[3].x=p[2].x) és (p[3].y=p[2].y)
Függvény vége.
```

2005. Harmadik forduló

Ötödik-nyolcadik osztályosok

1. feladat: Nyaklánc (25 pont)

Számolni kell, hogy hány egymás melletti egyforma gyöngy található! Ha újabb színű gyöngy jön, akkor a számolást elölről kell kezdeni! Ha nem mind egyforma volt, akkor a leghosszabb egyforma szakasz olyan is lehet, hogy az N. gyöngy után az 1. gyönggyel folytatódik, ezt külön kell vizsgálni!

```
Gyöngyök(N,gy,maxdb,maxgy):
  maxgy:=gy(1); maxdb:=1; db:=1
  Ciklus i=2-től N-ig
    Ha gy(i)=gy(i-1) akkor db:=db+1
      különben Ha db>maxdb akkor maxdb:=db; maxgy:=gy(i-1)
               db:=1
    Elágazás vége
  Ciklus vége
  Ha db=n akkor maxdb:=n
    különben i:=1
             Ciklus amíg i≤n és gy(i)=gy(n)
               i:=i+1; db:=db+1
             Ciklus vége
             Ha db>maxdb akkor maxdb:=db; maxgy:=gy(n)
  Elágazás vége
Eljárás vége.
```

Másik megoldás lehetne, hogy a gyöngyöket leíró vektor tartalmát lemásoljuk az eddigi elemek mögé (az N+1. helytől a 2*N. helyig), így nem a végén túlnyúló rész is egyszerűen vizsgálható.

2. feladat: Csapat (25 pont)

A kezdetben a pályán levő játékosoknak 60 percet állítunk be játékidőnek, a többieknek pedig 0 percet. Ha valakit lecserélnek, akkor a hátralevő idejét átadja annak, akire cserélik.

```
Csapat(N,kezdő,idő,M,kit,kire,mikor):
  idő():=(0,…,0)
  Ciklus i=1-től 7-ig
    idő(kezdő(i)):=60
  Ciklus vége
  Ciklus i=1-től m-ig
    idő(kit(i)):=idő(kit(i))-(60-mikor(i))
    idő(kire(i)):=idő(kire(i))+60-mikor(i)
  Ciklus vége
Eljárás vége.
```

3. feladat: Állomások (25 pont)

Érdemes az indulási és érkezési időket percre átszámítani (`idő(i)`). Ekkor a szomszédos értékek különbsége (figyelembe véve a 10 perces várakozást) maximuma az első részfeladat megoldása.

A második részfeladatban azok a vonatok állnak éjfélkor valamelyik állomáson, amelyek éjfél előtt legfeljebb 10 perccel érkeztek, s azok haladnak két állomás között, amelyek az egyik állomáson még éjfél előtt voltak, a másikra pedig éjfél után érkeztek.

```
Vonatok(N,idő,legh,dba,a,dbh,h):
  legh:=idő(2)-idő(1)
  Ciklus i=3-tól N-ig
    Ha idő(i)>idő(i-1)          {ugyanazon a napon vannak-e?}
       akkor Ha idő(i)-idő(i-1)-10>legh
                akkor legh:=idő(i)-idő(i-1)-10
    különben Ha idő(i)+24*60-idő(i-1)-10>legh
                akkor legh:=idő(i)+24*60-idő(i-1)-10
  Ciklus vége
  dba:=0; dbh:=0
  Ciklus i=1-től N-1-ig
    Ha idő(i)>24*60-10 akkor dba:=dba+1; a(dba):=i
    különben ha idő(i+1)<idő(i) akkor dbh:=dbh+1; h(dbh):=i
  Ciklus vége
Eljárás vége.
```

Kilencedik-tizedik osztályosok

1. feladat: Verseny (18 pont)

Készítsük el az eredményeket tartalmazó mátrixot, ahol `ered(j,k)` akkor legyen igaz, ha a `j`. csapat már legyőzte a `k`. csapatot! Azaz a feladat gráfját ábrázoljuk csúcsmátrix-szal!

Az első részfeladatban olyan `(i,j)` indexpárt kell keresni, ahol `ered(i,j)` és `ered(j,i)` is igaz értékű.

```
Legyőzték egymást(N,ered,van,cs1,cs2):
  i:=1; j:=1
  Ciklus amíg i≤N és nem (ered(i,j) és ered(j,i))
    Ha j=N akkor i:=i+1; j:=1 különben j:=j+1
  Ciklus vége
  van:=i≤N
  Ha van akkor cs1:=i; cs2:=j
Eljárás vége.
```

A második részfeladatban olyan `i` indexet kell keresni, ahol az í-edik csapat már játszott (azaz van olyan `j`, hogy `ered(i,j)` vagy `ered(j,i)` igaz értékű) és még nem kapott ki (azaz nincs olyan `j`, hogy `ered(j,i)` igaz értékű).

```
Győztek(N,ered,gydb,gy):
  gydb:=0
  Ciklus i=1-től N-ig
   Ha Jócsapat(i) akkor gydb:=gydb+1; gy(gydb):=i
  Ciklus vége
Eljárás vége.
```

```
Jócsapat(i):
  játszott:=hamis; vesztett:=igaz; j:=1
  Ciklus amíg j≤N és nem vesztett
    Ha ered(i,j) vagy ered(j,i) akkor játszott:=igaz
    Ha ered(j,i) akkor vesztett:=igaz
    j:=j+1
  Ciklus vége
  Jócsapat:=játszott és nem vesztett
Függvény vége.
```

A Harmadik részfeladatban olyan pontot kell keresni a gráfban, ami legalább egy körben szerepel, az ilyen csapat ugyanis közvetve legyőzte saját magát. Használhatjuk ehhez a feladathoz a gráf mélységi bejárását, színezéssel. Ha a bejárás során visszaélt találunk, akkor éppen egy körhöz jutottunk.

```
Van(N,ered,csapat):
  szín():=(fehér,…,fehér); i:=1; vankör:=hamis
  Ciklus amíg i≤N és nem vankör
    Ha szín(i)=fehér
       akkor Legyőztemagát(N,ered,i,vankör,csapat)
    i:=i+1
  Ciklus vége
Eljárás vége.
```

```
Legyőztemagát(N,ered,i,vankör,csapat):
  vankör:=hamis
  Üres(V); Verembe(V,i,1); szín(i):=szürke
  Ciklus amíg nem üres?(V) és nem vankör
    (k,i):=veremből(V); kell:=igaz
    Ciklus amíg i≤N és kell
      Ha ered(k,i) akkor
         Ha szín(i)=szürke akkor vankör:=igaz; kell:=hamis
      különben ha szín(j)=fehér akkor kell:=hamis
      i:=i+1
    Ciklus vége
    Ha vankör akkor csapat:=i
    különben Ha i≤N akkor Verembe(V,k,i+1); Verembe(V,i,1)
                          szín(i):=szürke
    különben szín(k):=fekete
  Ciklus vége
Eljárás vége.
```

<u>2. feladat</u>: Királyok (18 pont)

Az Árpád-házi királyokat egy fában tároljuk, ahol a leszármazási kapcsolatot a fa szerkezete adja (`kapcs(i).szülő` az `i`-edik kapcsolat szülő, `kapcs(i).gyerek` pedig a gyerek sorszáma. A `király(i)` az uralkodási időszakokat tartalmazza, uralkodási sorrendben. A neveket nehéz kezelni, ezért helyettesítsük őket a sorszámukkal! A királyokat sorszámozzuk a felsorolásuk sorrendjében 1-től N-ig, akik a családfában szerepelnek, de nem voltak királyok, azok pedig kapjanak N-nél nagyobb sorszámokat!

Az első részfeladat ebben az esetben azon elem megtalálása a `király` tömbben, amiből a legmesszebb lehet menni úgy, hogy a közvetlen szomszéd mindig gyerek is egyben!

```
Leghosszabb(K,kapcs,N,király,kezdet,vég):
  kezdet:=király(1).tól; vég:=király(1).ig
  ke:=kezdet; ve:=vég
  Ciklus i=2-től N-ig
    Ha apjátkövette(i) akkor ve:=király(i).ig
    különben ke:=király(i).tól; ve:=király(i).ig
    Ha ve-ke>vég-kezdet akkor kezdet:=ke; vég:=ve
  Ciklus vége
Eljárás vége.
```

```
Apjátkövette(i):
  apa:=király(i-1).sorszám; gyerek:=király(i).sorszám; j:=1
  Ciklus amíg j≤K és nem
         (kapcs(j).szülő=apa és kapcs(j).gyerek=gyerek)
    k:=k+1
  Ciklus vége
  apjátkövette:=(j≤K)
Függvény vége
```

A második részfeladat azon csomópont megtalálása, amelynek a legtöbb gyereke volt király.

```
Legtöbbgyerekekirály(K,kapcs,N,király,dbt):
  dbt:=0
  Ciklus i=1-től N-ig
    db:=királygyerekeiszáma(i)
    Ha db>dbt akkor dbt:=db
  Ciklus vége
Eljárás vége.
```

```
Királygyerekeiszáma(i)
  db:=0; apa:=király(i).sorszám
  Ciklus j=1-től K-ig
    Ha kapcs(j).szülő=apa akkor
       Ha kapcs(j).gyerek<N+1 akkor db:=db+1
  Ciklus vége
  királygyerekeiszáma:=db
Függvény vége.
```

A harmadik részfeladat azon csomópontok számának meghatározása, amelyeknek legalább 1 gyereke van, de egyik gyereke sem volt király.

```
Nemkirályok(K,kapcs,N,király,dbnk):
  dbnk:=0
  Ciklus i=1-től N-ig
    Ha gyerekeinemkirályok(i) akkor dbnk:=dbnk+1
  Ciklus vége
Eljárás vége.
```

```
Gyerekeinemkirályok(i)
  db:=0; apa:=király(i).sorszám
  vangyereke:=hamis; király:=hamis
  Ciklus j=1-től K-ig
    Ha kapcs(j).szülő=apa
       akkor vangyereke:=igaz
             Ha kapcs(j).gyerek<N+1 akkor király:=igaz
  Ciklus vége
  gyerekeinemkirályok:=vangyereke és nem király
Függvény vége.
```

3. feladat: Úthálózat (21 pont)

Az út(N,M) mátrix tartalmazza az egyes kereszteződések leírását. Két-két bit tartozik minden irányhoz, bontsuk szét először a kódokat irányokra!

```
Ellenőrzés(N,M,út)
  Ciklus i=1-től N-ig
    Ciklus j=1-től M-ig
      fel:=út(i,j) mod 4
      jobbra:=(út(i,j) div 4) mod 4
      le:=(út(i,j) div 16) mod 4
      balra:=(út(i,j) div 64) mod 4
```

Valamerre se be, se ki nem lehet menni, ha az valamelyik irány 2-bites kódja 0 (kivéve a térkép szélén):

```
      Ha fel=0 és i>1 vagy jobbra=0 és j<m vagy le=0 és i<n
         vagy balra=0 és j>1 akkor hibás(i,j)
```

Egy kereszteződésből semerre sem lehet kimenni (a 2-bites kódok második bitje mind 0):

```
      Ha fel<2 és jobbra<2 és le<2 és balra<2
         akkor hibás(i,j)
```

Egy kereszteződésbe semerről sem lehet bejönni (a 2-bites kódok első bitje mind 0):

```
      Ha fel mod 2=0 és jobbra mod 2=0 és le mod 2=0
         és balra mod 2=0 akkor hibás(i,j)
```

Jobbra levő szomszéd nem jó (azaz innen lehet menni jobbra, de a jobboldali szomszédba nem lehetne jönni innen; vagy fordítva):

```
      Ha j<M akkor
         balra:=(út(i,j+1) div 64) mod 4
         Ha nem (balra=3 és jobbra=3 vagy
                 balra=1 és jobbra=2 vagy
                 balra=2 and jobbra=1
            akkor hibás(i,j); hibás(i,j+1)
```

Lefelé levő szomszéd nem jó (azaz innen lehet menni lefelé, de a lenti szomszédba nem lehetne jönni innen; vagy fordítva):

```
      Ha i<N akkor
         fel:=út(i+1,j) mod 4
         Ha nem (fel=3 és le=3 vagy fel=1 és le=2 vagy
                 fel=2 és le=1)
            akkor hibás(i,j); hibás(i+1,j)
    Ciklus vége
  Ciklus vége
```

Van-e kilépő, illetve belépő hely a térkép bal vagy jobb szélén?

```
  be:=hamis; ki:=hamis
  Ciklus i=1-től N-ig
    Ha út(i,1) div 64>1 akkor ki:=igaz
    Ha (út(i,1) div 64) mod 2=1 akkor be:=igaz
    Ha (út(i,m) div 4) mod 4>1 akkor ki:=igaz
    Ha (út(i,m) div 4) mod 2=1 akkor be:=igaz
  Ciklus vége
```

Van-e kilépő, illetve belépő hely a térkép alsó vagy felső szélén?

```
  Ciklus j=1-től M-ig
    Ha út(1,j) mod 4>1 akkor ki:=igaz
    Ha út(1,j) mod 2=1 akkor be:=igaz
    Ha (út(n,j) div 16) mod 4>1 akkor ki:=igaz
    Ha (út(n,j) div 16) mod 2=1 akkor be:=igaz
  Ciklus vége
  Ha nem be akkor hibás(1,0)
  Ha nem ki akkor hibás(0,1)
Eljárás vége
```

4. feladat: Zenekar (18 pont)

Első lépésként rendezzük sorba a megrendeléseket kezdőidő, azon belül pedig végidő szerint! Ezzel persze az eredeti sorszámuk elveszne, tehát a két idő mellett még az eredeti sorszámot is tárolni kell!

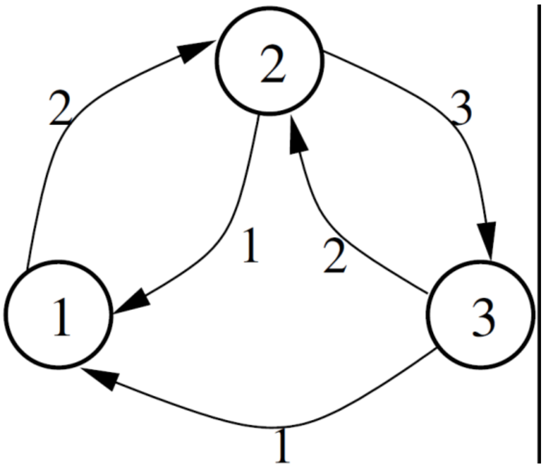
Ezután nézzük végig a megrendeléseket! Ha egy megrendelés végideje előtt van szabad nap, akkor osszuk be a lehető legkorábbi szabad napra!

```
Zenekar(N,Meg,Beoszt):
  Rendezés(N,Meg)
  szabad:=1
  Ciklus i=1-től N-ig
    Ha szabad≤Meg[i].u akkor
      Ha szabad<Meg[i].e akkor szabad:=Meg[i].e
      Beoszt[szabad]:=Meg[i].az
      szabad:=szabad+1
  Ciklus vége
Eljárás vége.
```

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Konténer (15 pont)

Számoljuk meg, hogy hány 1-es, 2-es, illetve 3-as címkéjü konténer van, ezek értéke N1,N2,N3. Ekkor az átrendezést követően az első N1 helyen 1-es, a következő N2 helyen 2-es, a következő N3 helyen 3-as konténernek kell lenni. Tekintsük azt a gráfot, amelynek csúcsai 1,2 és 3. Az I csúcsból a J csúcsba pontosan annyi él menjen, amennyi J-vel címkézett konténer van a rendezéstben I-vel címkézettek helyen



A példa bemenethez tartozó gráf.

Nyilvánvaló, hogy a gráf minden csúcsából annyi él indul ki, amennyi oda fut be. Továbbá, mivel három csúcs van, ezért a gráf összefüggő (lehet, hogy egy csúcshoz, vagy mindháromhoz nem csatlakozik él). Ezért a gráfot be lehet úgy járni, hogy minden élén pontosan egyszer megyünk át, és visszaérünk a kiindulási csúcshoz (Euler-kör). Tehát a megoldás: Tegyük ki az üres N+1-edik helyre egy olyan konténert, ami nincs a helyén, majd a gráfban innen indulva, fordított élef mentén Euler körben haladva végezzük az átpakolást, végül a szabad helyről teyük át a konténert a kiindulási helyre. Ez a megoldás optimális, mert 1+ nem a helyén lévő konténert mozgat. A Bejar eljárás az E tömbben állítja elő a kívánt sorrendet (Euller-kört).

```
Szamit:
  Ciklus i=1-től 3-ig
    Ciklus j=1-től 3-ig
      At[i,j]:=0; Poz[i,j]:=0
    Ciklus vége
  Ciklus vége
  Ciklus i=1-től n-ig
    Ha (i≤Hany[1]) akkor
      Ha (A[i]≠1) akkor
             inc(At[1,A[i]])
             Ha Poz[1,A[i]]=0 akkor Poz[1,A[i]]:=i;
      Elágazás vége
    különben ha (i≤Hany[1]+Hany[2]) akkor
      Ha (A[i]<>2) akkor
            inc(At[2,A[i]])
            Ha Poz[2,A[i]]=0 akkor Poz[2,A[i]]:=i;
      Elágazás vége
    különben ha (i>Hany[1]+Hany[2]) akkor
      Ha (A[i] ≠3) akkor
            inc(At[3,A[i]])
            Ha Poz[3,A[i]]=0 akkor Poz[3,A[i]]:=i;
      Elágazás vége
    Ciklus vége
    m:=0
    Ha (At[1,2]<>0) vagy (At[1,3]<>0)akkor kezd:=1
    különben ha (At[2,3]<>0) vagy (At[3,2]<>0) akkor kezd:=2
    különben exit
    Bejar(kezd)
Eljárás vége.

Bejar(p):
  Ciklus q=1-től 3-ig
  Ciklus amíg At[p,q]>0
      At[p,q]:=At[p,q]-1; Bejar(q)
    Ciklus vége
  Ciklus vége
  Inc(m); E[m]:=p
Eljárás vége

eolvas:
  Hany[1]:=0; Hany[2]:=0; Hany[3]:=0; Be(N)
  Ciklus i=1-től N-ig Do Begin
      Be(InFile,A[i]); Inc(Hany[A[i]])
  Ciklus vége
  A[N+1]:=1;  A[N+2]:=2; A[N+3]:=3
Eljárás vége.

Next(i,j):
  Ciklus
   inc(Poz[i,j])
  amíg A[Poz[i,j]]=j
  Ciklus vége
Eljárás vége.
```

```
KiIr:
  Ki(m)
  ures:=Poz[E[1], E[m-1]]; Next(E[1], E[m-1])
  Ki(ures,' ',N+1)
  Ciklus i=1-től m-2-ig
      j:=E[i]; jj:=E[i+1]; tol:=Poz[jj,j]
      writeln(Kif, tol,' ',ures )
      ures:=tol; Next(jj,j)
  Ciklus vége
  Ki(N+1,' ',ures)
Eljárás vége.
```

2. feladat: Bérlet (15 pont)

Rendezzük az igényeket az U(i) értékek szerint. Minden x-re ($0 \le x \le 200$) és minden i-re ($1 \le i \le N$) tekintsük azt a részproblémát, hogy mekkora a lenagyobb nyereség, ha az első x-nap adjuk ki a gépet, és csak az első i megrendelést vesszük. Jelölje ennek értékét Opt(x,i). Opt(0,i)=0, egyébként az alábbi rekurzív összefüggés áll fenn:

$$Opt(x,i)\begin{cases} Opt(x-1,i) & ha \quad x \neq U(i) \\ \max(Opt(x,i-1), Opt(E(i)-1,i-1)+1+U(i)-E(i)) & ha \quad x = U(i) \end{cases}$$

Számítsuk Opt(x,i) értékét táblázatkitöltés módszerével. Tehát olyan sorrendben kell számolni, hogy amikor Opt(x,i) értékét számítjuk, akkor Opt(x,i-1) és Opt(E(i)-1,i-1) értékét már kiszámítottuk. Egy ilyen sorrend lehet x-szerint 0-tól 200-ig, ezen belül i-szerint 1-től N-ig haladva. A kimenet első sorába az Opt(200,N) értéket kell írni. Miután kitöltöttük az Opt táblázatot, visszafejtéssel meg tudunk határozni egy megoldást.

```
Szamit:
  SzamlaloRend(Meg)
  Ciklus x=0-tól MaxNap-ig
    Ciklus i=0-től N-ig
      Opt[x,i]:=0
    Ciklus vége
  Ciklus vége
  Ciklus i=1-től N-ig
    Ciklus x=1-től MaxNap-ig
      Opt[x,i]:=Opt[x-1,i]
      Ha Opt[x,i-1]>Opt[x,i] akkor Opt[x,i]:=Opt[x,i-1]
      Ha (Meg[i].u=x) akkor
        max:=Opt[Meg[i].e-1,i-1]+Meg[i].u-Meg[i].e+1
        Ha max>Opt[x,i] akkor Opt[x,i]:=max
      Elágazás vége
    Ciklus vége
  Ciklus vége
  Opti:=Opt[MaxNap,N]
  Ciklus i=1-től N-ig
    KiAd[i]:=0
  Ciklus vége
  i:=N; Nap:=MaxNap; m:=0
  Ciklus
    Ciklus amíg (i>0) és (Opt[Nap,i-1]=Opt[Nap,i])
      i:=i-1
    Ciklus vége
    m:=m+1; Kiad[m]:=Meg[i].az; Nap:=Meg[i].e-1; i:=i-1
  amíg (Nap=0)or(i=0) vagy (Opt[Nap,i]=0)
  Ciklus vége
Eljárás vége.
```

3. feladat: Park (15 pont)

A park úthálózatát ábrázolatjuk egy két dimenziós G tömbbel, ahol G(p,i)=q (i=1,2,3). ha p részlegből van közvetlen út q részlegbe. G(p,i) legyen 0, ha nincs i-edik szomszédhoz út.

A megoldás megadható egy mélységi bejárással.

```
Bejar(p):
  m:=m+1; Seta(m):=p; NemVolt(p):=Hamis
  Ciklus i=1-től 3-ig
    q:=G[p,i]
    Ha (q>0) és (NemVolt[q]) akkor
      Bejar(q); m:=m+1; Seta[m]:=p
    Elágazás vége
  Ciklus vége
Eljárás vége.
```

4. feladat: Malom (15 pont)

A feladatnak megfelelő gráfban N csomópont van, s közöttük M malom. A feladat hasonlít a minimális költségű feszítőerdőre, ahol az erdő minden fájában pontosan egy malomnak kell lennie. A különbség az, hogy nem a feszítőerdő élei összhosszúságának kell minimálisnak lennie, hanem a malmokból kiinduló utak összhosszúságának!

```
Feszítőerdő(N,M,Cs,Ár,Honnan,Minkölt):
  db:=M; Minkölt:=0
  Ciklus amíg db<n
    minhossz:=maxint div 2
    Ciklus i=1-től N-ig
      Ha Honnan(i)=0 akkor
        Ciklus j=1-től N-ig
          Ha Honnan(j)>0 akkor
            Ha Cs(i,j)+Ár(j)<minhossz akkor
               minhossz:=Cs(i,j)+Ár(j); mini:=i; minj:=j
        Ciklus vége
    Ciklus vége
    Ha minhossz<maxint div 2
       akkor Minkölt:=Minkölt+minhossz
             Honnan(mini):=Honnan(minj); Ár(mini):=minhossz
    db:=db+1
  Ciklus vége
Eljárás vége.
```

5. feladat: Téglalap (15 pont)

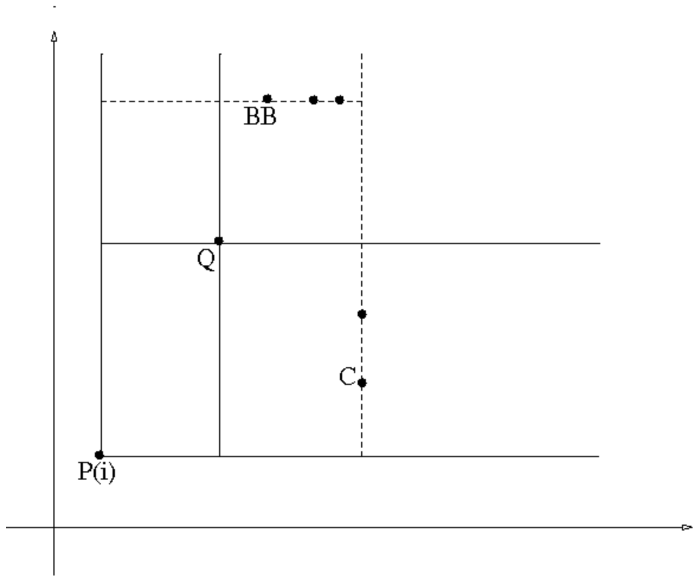
Belátható, hogy egy kivételtől eltekintve, minden esetben van olyan, a Q pontot körölvevő téglalap, hogy három ponttal teljesül, hogy mind a négy oldalra esik pontosan egy pontja a ponthalmaznak. Tehát a keresett téglalap egyik sarkában van a ponthalmaz egyik pontja. A bemenet bármely pontára a ponthalmaz elemszámával arányos futási idejű algoritmussal el tudjuk dönteni, hogy van-e olyan Q-t körbevevő táglalap, amelynek egyik csúcsa az adott pont, minden oldalára pontosan egy pontja esik a ponthalmaznak, és belsejébe nem esik egy sem. A `BalAsóSarok(P,Q,i,A,B)` függvény akkor és csak akkor ad igaz értéket, ha van olyan befoglaló téglalap, amelynek bal alsó sarka a P[i] pont, és ekkor az A és B a téglalap bal alsó, illetve jobb felső sarka lesz.

Ellenőrízzük, hogy a ponthalmaz egyik pontja sem esik a P[i] és Q pontok által meghatározott téglalapba, és határozzuk meg azt a C pontot, amely a legkisebb x-koordinátájú olyan pont, amelre teljesül: Q.x ≤ P[j].x és P[i].y≤ P[j].y < Q.y. Ha több ilyen van, azt jegyezzük meg. Majd legyen BB a legkisebb y-korrdinátájú olyan P[j] pont, amelyre teljesül, hogy:P[i].x < P[j].x < xmin. Ha több ilyen van, akkor azek közül a legnagyobb x-koordinátájut vegyük. Ha BB.x>Q.x, akkor az A=P[i] és a B=BB pontpár megoldás. Ha BB.x<Q.x, akkor A:=P[i], B.x:=C.x megoldás, feltéve, hogy egyetlen ilyen BB pont van, továbbá egyetlen olyan P[j] , Q.x ≤ P[j].x és P[i].y≤ P[j].y < Q.y. pont van amelyre P[j].x=C.x. Egyébként a függvény hamis értékkel tér vissza.

A másik három eset analóg módon kezelhető.

A kivételes eset az, amikor a befoglaló téglalap és a Q ponton átmenő, X, illetve Y-tengellyel párhuzamos egyenesek metszéspotjaiban van a négy pont.

Megjegyezzük, hogy a ponthalmaz alkalmas rendezésevel elérhető, hogy a $\log_2(N)$ időben eldönthető, hogy egy pont sarokpontja-e egy befoglaló téglalapnak. Tehát a teljes algoritmus $N.\log_2(N)$ futási idejű lehet.




```
Beolvas(P,N,Q):
  Be(Q.x,Q.y)
  T1.x:=q.x; T1.y:=MaxXY+1
  T2.x:=0; T2.y:=q.y
  T3.x:=q.x; T3.y:=0
  T4.x:=MaxXY+1; T4.y:=q.y
  Be(N)
  Ciklus i=1-től N-ig
    Be(P[i].x,P[i].y)
    Ha P[i].x=Q.x akkor
      Ha P[i].y>Q.y és P[i].y<T1.y akkor T1.y:=P[i].y
      Ha P[i].y<Q.y és P[i].y>T3.y akkor T3.y:=P[i].y
    Elágazás vége
    Ha P[i].y=Q.y akkor
      Ha P[i].x<Q.x és P[i].x>T2.x akkor T2.x:=P[i].x
      Ha P[i].x>Q.x és P[i].x<T4.x akkor T4.x:=P[i].x
    Elágazás vége
  Ciklus vége

  i:=1
  Ciklus amíg i≤N
    Ha P[i].y>T1.y vagy P[i].x<T2.x vagy P[i].y<T3.y vagy
       P[i].x>T4.x akkor P[i]:=P[n]; N:=N-1
    különben i:=i+1
  Ciklus vége
Eljárás vége.
```

```
BalAlsoSarok(P,Q,i,A,B):
  BalAlsoSarok:=Hamis; xmin:=MaxXY+1; tobbx:=Hamis
  C.x:=MaxXY+1
  Ciklus j=1-től N-ig
    Ha i≠j és p[j].x≥p[i].x és p[j].y≥p[i].y és p[j].y≤q.y
    akkor
      Ha p[j].x>q.x akkor
        Ha p[j].x<xmin akkor C:=p[j]; tobbx:=Hamis
      különben ha p[j].x=C.x akkor
        Ha P[j].y<C.y akkor C.y:=P[j].y; tobbx:=Igaz
      Elágazás vége
    Elágazás vége
  Ciklus vége
  BB.x:=P[i].x BB.y:=MaxXY+1; tobby:=Hamis
  Ciklus j=1-től N-ig
    Ha i≠j és p[j].x>p[i].x és p[j].x<xmin és p[j].y>q.y
    akkor
      Ha p[j].y<BB.y akkor BB:=p[j]; tobby:=Hamis
    különben ha p[j].y=BB.y akkor
      Ha p[j].x<BB.x akkor BB.x:=p[j].x; tobby:=Igaz
    Elágazás vége
  Ciklus vége
  Ha BB.x≠P[i].x) akkor
    A:=p[i].x.; B:=BB
    Ha BB.x>q.x akkor BalAlsoSarok:=Igaz; Kilép
  Elágazás vége
  Ha C.y≠P[i].y és nem tobbx és nem tobby akkor
    B.x:=C.x; B.y:=BB.y; BalAlsoSarok:=Igaz
Fuggvény vége.

KiIr(A, B):
  Ki(A.x,' ',A.y,' ',B.x,' ',B.y)
Eljárás vége.

Szamit(P,N,Q):
  A.x:=0; A.y:=0; B.x:=0;B.y:=0;
  Ciklus i=1-től N-ig
    Ha p[i].x<q.x akkor
       Ha p[i].y<q.y akkor
          Ha BalAlsoSarok(i) akkor KiIr(A,B)
       különben Ha p[i].y>q.y akkor
          Ha BalFelsoSarok(i) akkor KiIr(A,B)
       Elágazás vége
    különben Ha p[i].x>q.x akkor
       Ha p[i].y<q.y akkor
          Ha JobbAlsoSarok(i) akkor KiIr(A,B)
       különben Ha(p[i].y>q.y) akkor
         Ha JobbFelsoSarok(i) akkor KiIr(A,B)
       Elágazás vége
    Elágazás vége
  Ciklus vége
```

```
  Ha T1.y≤MaxXY és T3.y>0 és T2.x>0 és T4.x≤MaxXY akkor
    Ciklus i=1-től N-ig
      Ha p[i].x=T1.x és p[i].y=T1.y vagy
         p[i].x=T2.x és p[i].y=T2.y vagy
         p[i].x=T3.x és p[i].y=T3.y vagy
         p[i].x=T4.x és p[i].y=T4.y vagy
         p[i].x<T2.x vagy p[i].x>T4.x vagy
         p[i].y<T3.y vagy P[i].y>T1.y akkor {semmi}
      különben ha p[i].x≥T2.x és p[i].x≤T4.x és
                  p[i].y≤T1.y és p[i].y≥T3.y akkor KiIr(A,B)
  Ciklus vége
  A.x:=T1.x; A.y:=T3.y
  B.x:=T4.x, B.y:=T1.y
  KiIr(A, B)
Eljárás vége.
```

2006. Első forduló

Ötödik-nyolcadik osztályosok

1. feladat: Képlet (12 pont)

Megjegyzés: A képlet a legfeljebb négyjegyű X-négyzet középső két számjegyét adja.

A. X=12 → X=14 — 2 pont

X=42 → X=76 — 2 pont

X=63 → X=96 — 2 pont

B. X=0, 10, 50, 60 bármelyike jó — egyenként 2-2 pont, összesen legfeljebb 6 pont

2. feladat: Óra (20 pont)

A páratlan helyiértékeken a 8-as után már a 0 következik (azaz pl. 38 másodperc után 40 másodperc következik) 5+2 pont

A 24. óra megkezdése után is tovább számol (az előző hiba miatt 23 óra 58 perc 58 másodperc után) — 5+2 pont

24 óra helyett 60 órakor váltana 0 órára, de ekkor A(7)-et növelné (az előző hibák miatt 58 óra 58 perc 58 másodperc után) — 4+2 pont

Ha csak példát ad, az mindhárom esetre 2-2-2 pont.

3. feladat: Átrendezés (24 pont)

A1. M1=5, M2=3, M3=1 — 2 pont

A2. M1=3, M2=5, M3=1 — 2 pont

A3. M1=1, M2=3, M3=5 — 2 pont

B1. Csökkenő sorrendbe rendez (azaz M1≥M2≥M3 lesz a végén) — 6 pont

B2. M2 a legnagyobb, a következő M1, majd M3 (azaz M2≥M1≥M3 lesz a végén) — 6 pont

B3. Növekvő sorrendbe rendez (azaz M1≤M2≤M3 lesz a végén) — 6 pont

4. feladat: Vonalkód (18 pont)

A 0=0000, 1=1000, 2=0100, 3=1100, 5=1010, 6=1101, 7=1111, 9=1001 — 8*1 pont

B. Kimaradt kódok: 0110, 0111, 1110 — 3*2 pont

4=0111, vagy (visszafelé olvasva) 1110 — 2 pont

8=0110 — 2 pont

5. feladat: Szövegátalakító (26 pont)

(A szöveges megfogalmazások minták, ugyanolyan értelmű válasz is elfogadható.)

A. ’ABC****4***ABBB****5***CA’ — 6 pont
(a félkövér dőlt számok az adott karakterkódú karaktert jelölik)

Ha *4A nem szerepel, akkor 2 pont levonás

Ha *5C nem szerepel, akkor két pont levonás

Ha BBB helyett *3B szerepel, akkor 2 pont levonás

Minden más eltérés esetén a részfeladat 0 pontos

B. {*} – egymás utáni egyforma karakterekből négynél kevesebb volt — 4 pont

{**} – egymás utáni egyforma karakterekből legalább 4 volt — 4 pont

C. A háromnál hosszabb azonos karakterekből álló sorozatokat 3 karakterrel kódolja, a *-gal, az ismétlődés hosszának megfelelő kódú karakterrel és az ismétlődő karakterrel 6 pont

D. Hibás, ha az ismétlődés hosszabb, mint a legnagyobb lehetséges karakterkód (255) 6 pont

Kilencedik-tizedik osztályosok

1. feladat: Áramkör (20 pont)

Több, az alábbiakkal ekvivalens megoldás is lehet, amennyivel több kaput (műveletet) használ a mintamegoldásnál, annyival kell csökkenteni a pontszámot.

?1: A és (B vagy C) 5 pont

Más jó megoldás pl. (A és B) vagy (A és C), de mert eggyel több művelet, ezért csak 4 pont adható rá.

?2: nem(A és B) és C 5 pont

?2: Más jó megoldás pl. (nem A vagy nem B) és C, de mert eggyel több művelet, ezért csak 4 pont adható rá.

?3: (A és B) vagy nem C 5 pont

Más jó megoldás pl. (A vagy nem C) és (B vagy nem C), de mert kettővel több művelet, ezért csak 3 pont adható rá.

?4: (A és B) vagy (A vagy B) és C 5 pont

Más jó megoldás pl. (A és B) vagy (A és C) vagy (B és C), de mert eggyel több művelet, ezért csak 4 pont adható rá.

2. feladat: Kupac (20 pont)

A. Betesz(T,N,7) → T=(3,7,5,11,8,6,20,13,12,10,9) 4 pont

Betesz(T,N,8) → T=(3,7,5,11,8,6,20,13,12,10,9,8) 4 pont

Betesz(T,N,4) → T=(3,7,4,11,8,5,20,13,12,10,9,8,6) 4 pont

B. Kivesz(T,N,X) → T=(5,8,6,11,9,10,20,13,12) és X=3 3+1 pont

Kivesz(T,N,X) → T=(6,8,10,11,9,12,20,13) és X=5 3+1 pont

Megjegyzés: 2-2 pont adható, ha a második vagy a harmadik műveletet az eredeti tömbre alkalmazza.

3. feladat: Rágógumi (22 pont)

A. A pirosak száma vagy kettővel csökken, vagy marad 2 pont

A fehérek száma eggyel csökken, nő vagy marad 2 pont

A zöldek száma vagy kettővel csökken, vagy marad 2 pont

B. A pirosak párossága vagy páratlansága változatlan 2 pont

A zöldek párossága vagy páratlansága változatlan 2 pont

A rágógumik száma eggyel csökken vagy marad 1 pont

C. Végtelen ciklusba kerülünk, ha a végén 1 piros és 1 zöld rágógumi marad, azaz kezdetben mindkettőből páratlan számú volt 3 pont

D. Piros páros, zöld páros, van fehér → egy fehér marad 2 pont

Piros páros, zöld páros, nincs fehér → nem marad semmi 2 pont

Piros páros, zöld páratlan → egy zöld marad 2 pont

Piros páratlan, zöld páros → egy piros marad 2 pont

4. feladat: Fazekas (18 pont)

A. Opt=(10,10,20,35,40,50,75) 7*1 pont

B. *Opt(1)=Idő(1)* 3 pont

$$Opt(i)=\min\begin{cases} Opt(i-1)+Idő(i) & 3\ pont \\ Opt(j-1)+\max_{k=j..i} Idő(k)\ \ ahol\ \ i-j+1\le K & 5\ pont \end{cases}$$

5. feladat: Logika (20 pont)

A1. Az X nagyanyja az Y-nak 3 pont

A2. Az X az Y egy apai ágú ősnek az anyja (azaz Y anyja, vagy Y apjának az anyja, vagy Y apja apjának az anyja, ...) 5 pont

```
B1. anyaidédanya(X,Y) ha anyja(X,Z) és      2 pont
           szülője(Z,P) és                  2 pont
           anyja(P,Y).                      2 pont
```

```
B2. férfiőse(X,Y) ha apja(X,Y) vagy         2 pont
           szülője(Z,Y) és férfiőse(X,Z)    4 pont
```

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Kapuk (16 pont)

AND(A,B)=NOR(NOR(A,A),NOR(B,B)) 4 pont

OR(A,B)=NOR(NOR(A,B),NOR(A,B)) 4 pont

XOR(A,B)=NOR(NOR(A,B),NOR(NOR(A,A),NOR(B,B))) 4 pont

(azaz másképpen XOR(A,B)=NOR(NOR(A,B),AND(A,B)))

EQU(A,B)=NOR(NOR(A,NOR(B,B)),NOR(NOR(A,A),B))) 4 pont

Minden olyan megoldásra, amely több NOR kaput használ, fele pontszám adható. Ilyen pl. az EQU(A,B)=NOR(XOR(A,B),XOR(A,B)) megoldás.

2. feladat: Kupac (20 pont)

A. Módosít(T,N,8,7) → T=(3,7,5,8,9,6,20,11,12,10) 4 pont

Módosít(T,N,7,10) → T=(3,7,5,8,9,6,10,11,12,10) 4 pont

Módosít(T,N,10,2) → T=(2,3,5,8,7,6,10,11,12,9) 4 pont

B. Módosít(T,N,1,7) → T=(5,8,6,11,9,7,20,13,12,10) 4 pont

Módosít(T,N,2,11) → T=(5,9,6,11,10,7,20,13,12,11) 4 pont

Megjegyzés: 2-2 pont adható, ha a második vagy a harmadik műveletet az eredeti tömbre alkalmazza.

3. feladat: Bors (24 pont)

A. A feketék száma vagy hárommal csökken, vagy marad 2 pont

A fehérek száma kettővel csökken vagy marad 2 pont

B. A feketéknél a hárommal osztás maradéka változatlan 2 pont

A fehéreknél pedig a kettővel osztás maradéka változatlan 2 pont

C. Végtelen ciklusba kerülünk, ha a végén 2 fekete és egy fehér bors marad (mert mind visszateszszük), azaz kezdetben a feketék száma hárommal osztási maradéka 2 volt, a fehérek száma pedig páratlan 4 pont

D. Fekete 3*x típusú, fehér páros → két fehér bors marad vagy nem marad bors 2+2 pont

Fekete 3*x típusú, fehér páratlan → egy fehér bors marad 2 pont

Fekete 3*x+1 típusú, fehér páros → egy fekete bors marad 2 pont

Fekete 3*x+1 típusú, fehér páratlan → egy fehér és egy fekete bors marad 2 pont

Fekete 3*x+2 típusú, fehér páros → két fekete bors marad 2 pont

4. feladat: Tükörszó (18 pont)

A. Az átlóban és alatta 0-k vannak 1 pont

Az átló fölött 1-esek 1 pont

Az afölötti átlóban 2-esek, de a második sorban 0 1+1 pont

A következőben két 1-es, két 3-as 1+1 pont

A következőben két 2-es, egy 4-es 1+1 pont

A következőben két 3-as 1 pont

A jobb felső sarokban egy 4-es 1 pont

B. $$M(i,j)=\begin{cases}0 & \text{ha } i \geq j \quad\quad 1\ pont\\ M(i+1,j-1) & \text{ha } i<j \text{ és } S[i]=S[j] \quad\quad 3\ pont\\ 1+Min(M(i+1,j),M(i,j-1)) & \text{ha } i<j \text{ és } S[i]\neq S[j] \quad\quad 4\ pont\end{cases}$$

5. feladat: Logika (22 pont)

A1. X nagyapja Y-nak (vagy: X olyan férfi, akinek Y az unokája) 3 pont

A2. Az X az Y egy anyai ágú ősnek az apja (azaz Y apja, vagy Y anyjának az apja, vagy Y anyja anyjának az apja, ...) 5 pont

```
B1. anyaidédapa(X,Y) ha apja(X,Z) és          2 pont
            szülője(Z,P) és                    2 pont
            anyja(P,Y).                        2 pont

B2. férfiutód(X,Y) ha szülője(Y,X) és          2 pont
            apja(X,Q) vagy                     2 pont
            szülője(Y,Z) és férfiutód(X,Z).    4 pont
```

Más megoldás: `férfiutód(X,Y) ha őse(Y,X)` 6 pont
`és apja(X,Q).` 2 pont

Megjegyzés: Q helyén bármi X,Y,Z-től különböző jel lehet.

2006. Második forduló

Ötödik-nyolcadik osztályosok

1. feladat: Mozi (27 pont)

Először számoljuk ki mindhárom kedvezmény lehetséges értékét, majd válasszuk a legnagyobbat!

```
Mozi(N):
  Ha N<10 akkor csk:=0
  különben ha N<20 akkor csk:=5
  különben ha N<30 akkor csk:=8
  különben ha N<41 akkor csk:=12 különben csk:=14
  Ha N<5 akkor ik:=0
  különben ha N<12 akkor ik:=100/N
  különben ha N<20 akkor ik:=200/N
  különben ha N<29 akkor ik:=300/N
  különben ha N<41 akkor ik:=400/N különben ik:=500/N
  dk:=10
  Ha csk>ik akkor
     ha csk>dk akkor Ki: 'CSOPORTOS KEDVEZMENY'
            különben Ki: 'DIAKKEDVEZMENY'
  különben ha ik>dk akkor Ki. 'ISKOLAI KEDVEZMENY'
                 különben Ki: 'DIAKKEDVEZMENY'
Eljárás vége.
```

2. feladat: Utazás (28 pont)

Először azon állomásokon, ahol a vonat megállt (ha volt ilyen), számoljuk ki a legrövidebb várakozási időt! Ezután keressük meg a leghosszabb szakaszt, ahol a vonat nem állt meg (ha volt olyan állomás, ahol nem állt meg)!

```
Utazás(N,ind,érk):
  ind(N):=maxint
  Ha N>2 akkor
    lra:=1; lri:=maxint
    Ciklus i=2-től n-1-ig
      Ha ind(i)>érk(i) akkor
         Ha ind(i)-érk(i)<lri
            akkor lri:=ind(i)-érk(i); lra:=i
    Ciklus vége
    Ha lra>1 akkor Ki: 'Legrövidebb ideig állt: ',lra
             különben Ki: 'Nem állt meg sehol sem'
    lk:=0; lv:=0; k:=1; v:=1
    Ciklus i=2-től N-ig
      Ha érk(i)=ind(i) akkor v:=i+1
      különben Ha v=i akkor ha lv-lk<v-k akkor lk:=k; lv:=v
                k:=i
      Elágazás vége
    Ciklus vége
    Ha lk>0 akkor Ki: 'Legmesszebb levő állomások: ',lk,lv
            különben Ki: 'A vonat mindenhol megállt'
  különben Ki: 'Nincs közbülső állomás'
Eljárás vége.
```

3. feladat: Hasáb (20 pont)

A feladat lényege a hasábok hosszának meghatározása. Kihasználhatjuk, hogy legfeljebb 3 hasáb lesz.

```
Hasábok(N,K,x,h):
  hm:=(N-1) div K+1
  h(1):=0; h(2):=0; h(3):=0
  j:=1
  Ciklus i=1-től N-ig
    Ha hossz(x(i))>h(j) akkor h(j):=hossz(x(i));
    Ha i=hm akkor j:=2 különben Ha i=2*hm akkor j:=3
  Ciklus vége
Eljárás vége.
```

Kilencedik-tizedik osztályosok

1. feladat: Átló (10 pont)

Egy tetszőleges átló az adott sokszöget mindig 2 sokszögre bontja. A sokszögek csúcspontjait halmazokban tároljuk. ha egy új átló két végpontja azonos halmazban van, akkor azt a halmazt az átlónál kettéosztjuk. Ha az új átló két végpontja különböző halmazban van, akkor biztosan metsz egy korábbi átlót.

```
Átló(M,i):
  db:=1; h[1]:=[1..n]; i:=1
  Ciklus amíg i≤M és nem metsz(i)
    i:=i+1
  Ciklus vége
  Átló:=(i≤M)
Függvény vége.

Metsz(i):
  Ha x(i)>y(i) akkor z:=x(i); x(i):=y(i); y(i):=z
  volt:=hamis; ujdb:=db
  Ciklus j=1-től db-ig
    Ha x(i)∈h(j) és y(i)∈h(j)
      akkor Halmazbontás(j,x(i),y(j),ujdb,volt)
  Ciklus vége
  db:=ujdb; metsz:=nem volt
Eljárás vége

Halmazbontás(j,x,y,ujdb,volt):
  volt:=igaz; ujdb:=ujdb+1; h(ujdb):=()
  Ciklus i=1-től N-ig
    Ha i∈h(j) akkor
       Ha i<x vagy i>y akkor
          h(ujdb):=h(ujdb)+(i); h(j):=h(j)-(i)
       különben ha i=x vagy i=y akkor h(ujdb):=h(ujdb)+(i)
  Ciklus vége
Eljárás vége
```

Minden átló jó — 2 pont

2. feladat: Csőposta (15 pont)

Minden csomaghoz tároljuk az indulás helyét, az indulási időt, valamint az út megtételéhez szükséges időt.

Ütközési esetek:

1. A szemből jövő hamarabb indul, mint az előző megérkezne.
2. Az azonos helyről induló hamarabb érkezne, mint az előző.

```
Csőposta(N,cs,K,er):
  beér:=cs(1).mikor+cs(1).mennyit; idő:=cs(1).mennyit
  Ciklus i=2-től M-ig
    Ha cs(i).honnan<>cs(i-1).honnan akkor {szembe jön}
      Ha cs(i).mikor<beér akkor
        k:=k+1; er(k):=i
        Ha cs(i).mennyit≤idő akkor {ő a gyorsabb}
           beér:=cs(i).mikor+cs(i).mennyit
           idő:=cs(i).mennyit
        különben ha cs(i).honnan='A' {az előző a gyorsabb}
                    akkor cs(i).honnan:='B'
                    különben cs(i).honnan:='A'
      különben beér:=cs(i).mikor+cs(i).mennyit
               idő:=cs(i).mennyit
    különben {egy irányból jönnek}
      Ha cs(i).mikor+cs(i).mennyit<beér
         akkor k:=k+1; er(k):=i
      különben beér:=cs(i).mikor+cs(i).mennyit
               idő:=cs(i).mennyit
  Ciklus vége
Eljárás vége
```

<u>3. feladat</u>: Sorozat (10 pont)

Sorra vizsgáljuk a bemenet elemeit. Ha a következő elem egyik sorozatba sem tehető (mert mindkettő utolsójánál kisebb), akkor nincs megoldás. Ha az első sorozat utolsó eleme a kisebb, akkor először próbáljuk a másodikba tenni! Ha a másodiké a kisebb, akkor pedig az elsőbe!

```
Sorozat(N,A,N1,A1,N2,A2,van):
  N1:=0; N2:=0; A1(0):=0; A2(0):=0; van:=igaz; i:=1
  Ciklus amíg i≤N és van
    Ha A(i)<A1(N1) és A(i)<A2(N2) {A(i) egyikbe sem tehető}
       akkor N1:=0; N2:=0; van:=hamis

    különben ha A1(N1)<A2(N2) {az első végén van a kisebb}
                akkor ha A2(N2)≤A(i)
                         akkor N2:=N2+1; A2(N2):=A(i)
                      különben N1:=N1+1; A1(N1):=A(i)
             {a második végén van a kisebb}
    különben ha A1(N1)≤A(i) akkor N1:=N1+1; A1(N1):=A(i)
                         különben N2:=N2+1; A2(N2):=A(i)
  Ciklus vége
Eljárás vége.
```

<u>4. feladat</u>: Térkép (20 pont)

Kezdetben az (i,j) helyen levő csillagoknál t(i,j) értékét -1-re állítjuk. Az első oszlopban meghatározzuk minden helyre az ilyen jobb alsó sarkú üres téglalapok bal felső sarkát – ez szintén csak az első oszlopban lehet az adott hely feletti legközelebbi csillag alatt. A többi oszlopban is az adott hely feletti legközelebbi csillag alatt kell lennie, de az előző oszlopok miatt lehet, hogy ennél is lejjebb lesz, mert így lesz a téglalap területe a lehető legnagyobb.

```
Térkép(M,t,ter):
  k:=1
  Ciklus i=1-től M-ig
    Ha t(i,1).sor=-1 akkor k:=i+1
                különben t(i,1).sor:=k; t(i,1).oszlop:=1
  Ciklus vége
  Ciklus j=2-től M-ig
    k:=1
    Ciklus i=1-től M-ig
      Ha t(i,j,1)=-1 akkor k:=i+1
      különben ter:=i-k+1; t(i,j).sor:=k; t(i,j).oszlop:=j
               jj:=j-1; kke:=k
               Ciklus amíg jj≥1 és t(i,jj).sor>-1
                 kk:=i
                 Ciklus amíg kk≥kke és t(kk,jj).sor>-1
                   kk:=kk-1
                 Ciklus vége
                 kk:=kk+1
                 Ha ter<(i-kk+1)*(j-jj+1)
                    akkor ter:=(i-kk+1)*(j-jj+1)
                          t(i,j).sor:=kk; t(i,j).oszlop:=jj
                 jj:=jj-1; kke:=kk
    Ciklus vége
  Ciklus vége
  k:=1; l:=1; ter:=0
  Ciklus i=1-től M-ig
    Ciklus j=1-től M-ig
      Ha t(i,j).sor>-1 és
                (i-t(i,j).sor+1)*(j-t(i,j).oszlop+1)>ter
         akkor ter:=(i-t(i,j).sor+1)*(j-t(i,j).oszlop+1)
               k:=i; l:=j
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

<u>5. feladat</u>: Randevú (20 pont)

Tekintsük azt a gráfot,amelynek pontjai a városok, és U-ból V-be akkor és csak akkor van irányított él, ha van vonatjárat U-ból V-be! Ebben a gráfban legrövidebb utat kell keresni E-ből R-be és A-ból R-be. Ha megfordítjuk ebben a gráfban az élek irányát, akkor egyetlen, R-gyökerű szélességi bejárással meg tudjuk oldani a feladatot. Tehát már beolvasáskor a fordított (transzponált) gráfot állítjuk elő.

```
Beolvas(N,E,A,R,M,G):
  Be: N,E,A,R,M; Fok():=(0,…,0)
  Ciklus i=1-től M-ig
    Be: y,x; Fok(x):=Fok(x)+1; G(x,Fok(x)):=y;
  Ciklus vége
Eljárás vége.

Sorba(p):
  vege:=vege+1; S(vege):=p
Eljárás vége.

Sorbol:
  Sorbol:=S(eleje); eleje:=eleje+1
Függvény vége.

Nemures:
  Nemures:=eleje<=vege
Függvény vége.
```

```
Szamit(G,N,R,Apa);
  Apa():=(-1,…,-1); Apa(R):=0; eleje:=1; vege:=0; Sorba(R)
  Ciklus amíg Nemures
    p:=Sorbol
    Ciklus i=1-től Fok(p)-ig
      q:=G(p,i)
      Ha Apa(q)<0 akkor Apa(q):=p; Sorba(q)
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

```
Kiir(E,A,R,Apa):
  NE:=0; NA:=0
  Ha Apa(E)<0 vagy Apa(A)<0 akkor Ki: 0,0
  különben NE:=1 x:=E
           Ciklus
             NE:=NE+1; x:=Apa(x)
           amíg x≠R
           Ciklus vége
           NA:=1; x:=A
           Ciklus
             NA:=NA+1; x:=Apa(x)
           amíg x≠R
           Ciklus vége
           Ki: NE,NA
           x:=E; Ki(x,' ')
           Ciklus
             x:=Apa(x); Ki: x
           amíg x≠R
           Ciklus vége
           x:=A; Ki: x
           Ciklus
             x:=Apa(x); Ki: x
           amíg x≠R
           Ciklus vége
Eljárás vége.
```

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Kiállítás (15 pont)

A feladatban N intervallumnak kell előállítani az unióját. Int(x) legyen az x-ben kezdődő intervallum végpontja! Ha többen is kezdődnek x-ben, akkor a később befejeződőé! Így nincs más teendőnk, mint a lehetséges időpontokon végigmenni és tárolni a kezdődő és végződő eredmény intervallumokat.

```
Kiállítás(Int,M):
  M:=0; x:=1
  Ciklus amíg Int(x)=0
    x:=x+1
  Ciklus vége
```

```
  Kezd:=x; Vég:=Int(x)
  Ciklus amíg x≤MaxIdő
    Ha Int(x)>0
       akkor Ha x>Vég
                akkor M:=M+1; Int(Kezd):=Vég
                      Kezd:=x; Vég:=Int(x)
             különben Ha Int(x)>Vég akkor Vég:=Int(x)
       Int(x):=0
    Elágazás vége
  Ciklus vége
  M:=M+1; Int(Kezd):=Vég
Eljárás vége.
```

<u>2. feladat</u>: Futár (15 pont)

A két indulási idő sorozatot össze kell futtatni, s nézni, hogy a két futár szembe megy-e egymással. Ha szembe mennek, akkor meg kell nézni, hogy találkozhatnak-e (azaz az egyik beérkezési ideje előtt indul-e a második, vagy fordítva).

```
Futár(N,f1,M,f2,k,er):
  f1(n+1).mikor:=maxint; f2(m+1).mikor:=maxint
  i:=1; j:=1; k:=0
  Ciklus amíg i<n+1 vagy j<m+1
    Ha f1(i).mikor<f2(j).mikor
       akkor Ha f1(i).honnan≠f2(j).honnan
                akkor Ha f1(i).mikor+o>f2(j).mikor
                         akkor k:=k+1; er(k,1):=i; er(k,2):=j
             i:=i+1
    különben ha f1(i).mikor=f2(j).mikor
       akkor Ha f1(i).honnan≠f2(j).honnan
                akkor k:=k+1; er(k,1):=i; er(k,2):=j
             i:=i+1; j:=j+1
    különben {f1(i).mikor>f2(j).mikor}
             Ha f1(i).honnan≠f2(j).honnan
                akkor Ha f2(j).mikor+o>f1(i).mikor
                         akkor k:=k+1; er(k,1):=i; er(k,2):=j
       j:=j+1
  Ciklus vége
Eljárás vége.
```

<u>3. feladat</u>: Konferencia (15 pont)

A feladat visszalépéses kereséssel oldható meg. A lehetséges gyorsítás miatt először rendezzük a szekciókat előadásszám szerint csökkenő sorrendbe! Tudjuk, hogy biztosan van megoldás.

```
Konferencia(N,K,L,e,sz):
  x:=(0,…0); i:=1
  Ciklus amíg i≤N
    j:=x(i)+1
    Ciklus amíg nem befér(i,j,h)
      j:=j+1
    Ciklus vége
    Ha j≤K akkor x(i):=j
                 Ciklus c=h-e(i).db+1-től h-ig
                   sz(c,j):=e(i).nev
                 Ciklus vége
                 i:=i+1
    különben x(i):=0; i:=i-1
  Ciklus vége
Eljárás vége.
```

```
befér(i,j,h):
  h:=e(i).db
  Ciklus c=1-től i-1-ig
    Ha x(c)=j akkor h:=h+e(c).db
  Ciklus vége
  befér:=(h≤L)
Függvény vége.
```

4. feladat: Terület (15 pont)

Az oldalak száma a sarokpontok számának fele lesz. Minden második oldal vízszintes, minden második pedig függőleges lesz.

Minden függőleges szakasztól jobbra levő területet fessük be kizáró vagy művelettel! Így a belső pontok páratlan sokszor lesznek befestve, a külső pontok pedig páros sokszor, azaz a végén csak a belső pontok maradnak befestettek.

```
terület(N,M,t,ter):
  N:=N div 2; t:=(0,…,0)
  Ciklus k=1-től N-ig
    Ha v(k,1).y<v(k,2).y akkor a:=v(k,1).y; b:=v(k,2).y
                  különben b:=v(k,1).y; a:=v(k,2).y
    Ciklus i=a-tól b-1-ig
      Ciklus j=v(k,1).x+1-től M-ig
        t(i,j):=1-t(i,j)
      Ciklus vége
    Ciklus vége
  Ciklus vége
  Ciklus k=1-től N-ig
    Ha v(k,1).y<v(k,2).y akkor a:=v(k,1).y; b:=v(k,2).y
                  különben b:=v(k,1).y; a:=v(k,2).y
    Ciklus i=a-tól b-1-ig
      t(i,j):=1
    Ciklus vége
  Ciklus vége
  v(n+1):=v(1)
  Ciklus k=1-től N-ig
    a:=v(k,2).x; b:=v(k+1,1).x
    Ha a>b akkor c:=a; a:=b; b:=c
    Ciklus i=a-tól b-ig
      t(v(k,2).y,i):=1
    Ciklus vége
  Ciklus vége
  ter:=0
  Ciklus i=1-tőlM-ig
    Ciklus j=1-től M-ig
      ter:=ter+t(i,j)
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

5. feladat: Találka (15 pont)

Tekintsük azt a gráfot,amelynek pontjai a városok, és U-ból V-be akkor és csak akkor van irányított él, ha van vonatjárat U-ból V-be. Ha ebben a gráfban van út Ádám városából, A-ból Éva városába E-be, akkor ez a megoldás. Ha nincs, de van E-ből A-ba akkor egy ilyen út megoldás. Egyébként, keresni kell egy (tetszőleges) olyan T várost, amelybe van út E-ből is és A-ból is. Mivel nem kell legrövidebb utat megadni, ezért két mélységi bejárással megoldható a feladat.

```
Eljárás Beolvas(N,E,A,G):
  Be(N,E,A,M); Fok():=(0,…,0)
  Ciklus i=1-től M-ig
    Be(x,y); Fok(x):=Fok(x)+1; G(x,Fok(x)):=y
  Ciklus vége
Eljárás vége.

Kiir(E,A,EApa,AApa);
  NE:=0; NA:=0;
  Ha AApa(E)>0 akkor NA:=1; x:=E; AUt(1):=E
                     Ciklus
                       NA:=NA+1; x:=AApa(x); AUt(NA):=x
                     amíg x≠A
                     Ciklus vége
                     Ki: NE,NA
                     Ciklus i=NA-tól 1-ig
                       Ki: AUt(i)
                     Ciklus vége
  különben ha EApa(A)>0 akkor
                     NE:=1; x:=A; EUt(1):=A
                     Ciklus
                       NE:=NE+1; x:=EApa(x); EUt(NE):=x
                     amíg x≠E
                     Ciklus vége
                     Ki: NE,NA
                     Ciklus i=NE-tól 1-ig
                       Ki: EUt(i)
                     Ciklus vége
  különben x:=1
           Ciklus amíg x≤N és (AApa(x)=0 vagy EApa(x)=0)
             x:=x+1
           Ciklus vége
           Ha x>N akkor Ki: NE,NA
           különben NA:=1; R:=x; AUt(1):=x
                    Ciklus
                      NA:=NA+1; x:=AApa(x); AUt(NA):=x
                    amíg x≠A
                     Ciklus vége
                    NE:=1; x:=R; EUt(1):=R
                    Ciklus
                      NE:=NE+1; x:=EApa(x); EUt(NE):=x
                    amíg x≠E
                    Ciklus vége
                    Ki: NE,NA
                    Ciklus i=NE-től a-ig
                      Ki: EUt(i)
                    Ciklus vége
                    Ciklus i=NA-től 1-ig
                      Ki: AUt(i)
                    Ciklus vége
  Elágazás vége
Eljárás vége.

MélyBejár(G,p,Apa):
  Ciklus i=1-től Fok(p)-ig
    q:=G(p,i)
    Ha Apa(q)<0 akkor Apa(q):=p; MélyBejár(G,q,Apa)
  Ciklus vége
Eljárás vége.
```

```
Számít(G,E,A,EApa,AApa):
  AApa():=(-1,…,-1); EApa():=(-1,…,-1)
  AApa(A):=0; MélyBejár(G,A,AApa)
  EApa(A):=0; MélyBejár(E,EApa)
Eljárás vége.
```

2006. Harmadik forduló

Ötödik-nyolcadik osztályosok

1. feladat: Számrendszer (20 pont)

Egy szám számjegyei összegét A alapú számrendszerben a következőképpen számoljuk ki:

```
összeg(i,A):
  b:=0
  Ciklus amíg a>0
    b:=b+a mod i; a:=a div i
  Ciklus vége
  összeg:=b
Függvény vége.
```

Ezután a feladat megoldása:

```
Számrendszer(S):
  Ciklus i=2-től A+1-ig
    Ha összeg(i,A)=S akkor Ki: i
  Ciklus vége
Eljárás vége.
```

2. feladat: Mérés (32 pont)

Először keressük meg az első és az utolsó jó mérést! A közöttük levő üres szakaszokon lineáris közelítést kell végeznünk! A két szélén pedig az első és az utolsó 2 mérés alapján kifelé lineáris közelítést.

```
Mérés(N,t):
  i:=1
  Ciklus amíg t(i)=-1
    i:=i+1
  Ciklus vége
  ii:=i; j:=n
  Ciklus amíg t(j)=-1
   j:=j-1
  Ciklus vége
  Ciklus k=i+1-től j-ig
    Ha t(k)≥0
       akkor Ha k>ii+1
                akkor Ciklus kk=ii+1-től k-1-ig
                        t(kk):=t(kk-1)+(t(k)-t(ii))/(k-ii)
                      Ciklus vége
             ii:=k
  Ciklus vége
  Ciklus ii=i-1-től 1-ig -1-esével
    t(ii):=t(ii+1)-(t(i+1)-t(i))
  Ciklus vége
  Ciklus ii=j+1-től N-ig
    t(ii):=t(ii-1)+(t(j)-t(j-1))
  Ciklus vége
Eljárás vége.
```

3. feladat: Tükörszó (23 pont)

Sorrendben a következőket vizsgáljuk az egy karakteres hangokon kívül:

1. dzs
2. ddzs
3. cs, zs, gy, ly, ny, ty, dz, sz
4. ccs, zzs, ggy, lly, nny, tty, ddz, ssz

```
Tükörszó(s,t):
  t:=''; i:=1
  Ciklus amíg i≤hossz(s)
    Ha s(i)+s(i+1)+s(i+2)='dzs'
       akkor t:=s(i)+s(i+1)+s(i+2)+t; i:=i+3
    különben ha s(i)+s(i+1)+s(i+2)+s(i+3)='ddzs'
       akkor t:=s(i)+s(i+1)+s(i+2)+s(i+3)+t; i:=i+4
    különben ha s(i)∈{'c','z'} és s(i+1)='s' vagy
                s(i)∈{'g','l','n','t'} és s(i+1)='y' vagy
                s(i)∈{'d','s'} és s(i+1)='z'
        akkor t:=s(i)+s(i+1)+t; i:=i+2
    különben ha s(i)∈{'c','z'} és s(i+1)∈{'c','z'} és
                s(i+2)='s' vagy
                s(i)∈{'g','l','n','t'} és
                s(i+1)∈{'g','l','n','t'} és s(i+2)='y' vagy
                s(i)∈{'d','s'} és s(i+1)∈{'d','s'} és
                s(i+2)='z'
        akkor t:=s(i)+s(i+1)+s(i+2)+t; i:=i+3
    különben t:=s(i)+t; i:=i+1
  Ciklus vége
Eljárás vége.
```

Kilencedik-tizedik osztályosok

1. feladat: Ismerősök (18 pont)

Kezdetben az t(i,j) érték akkor legyen igaz, ha i közvetlenül ismeri j-t!

```
Ismerősök(N,t,K,u,L,v)
  K:=0; L:=0
  Ciklus i=1-től N-ig
    Ciklus j=i+1-től N-ig
      Ha t(i,j) {A feladat, ismerik egymást}
         akkor c:=0
               Növel(c)
               Ciklus amíg c≤N és nem (t(i,c) és t(j,c))
                 Növel(c)
               Ciklus vége
               Ha c>N akkor K:=K+1; u(K,1):=i; u(K,2):=j
```

```
      különben {B feladat, nem ismerik egymást}
               c:=0
               Növel(c)
               Ciklus amíg c≤N és nem (t(i,c) és nem t(j,c)
                        vagy nem t(i,c) and t(j,c))
                 Növel(c)
               Ciklus vége
               Ha c>N akkor L:=L+1; v(L,1):=i; v(L,2):=j
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

```
Növel(c):
  c:=c+1
  Ha c=i akkor c:=c+1
  Ha c=j akkor c:=c+1
Eljárás vége.
```

2. feladat: Csoportok (18 pont)

A tt(i) jelentse azt, hogy az i-edik tanuló hányadik nyelvet választotta, az uu(i) pedig azt, hogy hányadik sportágat!

Először határozzuk meg a sportágaknak azt a halmazát, amelyek a tt(i) nyelvhez tartoznak!

```
  Ciklus i=1-től M-ig
    t(tt(i)):=t(tt(i)) ∪ {uu(i)}
  Ciklus vége
```

Ezután határozzuk meg a nyelveknek azt a halmazát, amelyek az uu(i) sportághoz tartoznak!

```
  Ciklus i=1-től M-ig
    u(uu(i)):=u(uu(i)) ∪ {tt(i)}
  Ciklus vége
```

A feladat megoldása azon nyelvek megadása, amelyekhez csak egyetlen olyan sportág tartozik, amelyhez csak egyetlen nyelv tartozik!

```
  db:=0
  Ciklus i=1-től N-ig
    Ha t(i)≠{} és jö(i) akkor db:=db+1
  Ciklus vége.
```

Az i jó nyelv, ha t(i) egyelemű és az egyetlen x eleméhez tartozó u(x) is egyelemű, méghozzá egyetlen eleme az i.

```
jó(i):
    a:=egyelemű(t(i),x); b:=egyelemű(u(x),y)
    jó:=(a és b és y=i
Függvény vége.
```

```
egyelemű(h,x):
  i:=1
  Ciklus amíg i∉h
    i:=i+1
  Ciklus vége
  x:=i; i:=i+1
  Ciklus amíg i≤max és i∉h
    i:=i+1
  Ciklus vége
  egyelemű:=(i>max)
Függvény vége.
```

3. feladat: Szemtanúk (19 pont)

Az i-edik esemény előtt jövők közül kell kiválasztani a legkésőbb elmenőt! Ha valaki később jön az i-edik eseménynél, akkor át kell lépni azon eseményeket, amelyek alatt az előbb kiválasztott vendég még nem ment el! Akkor nincs megoldás, ha van olyan esemény, ami előtt érkező összes vendég elment, a többiek pedig utána érkeznek.

```
Szemtanú(M,jön,megy,N,E,mego,V):
  mego:=1; i:=1; j:=1; Veg:=0
  Ciklus amíg i≤N és j≤M
    Ha jön(j)≤E(i)
       akkor Ha Veg<megy(j) akkor Veg:=megy(j); V(mego):=j
             j:=j+1
    különben Ha E(i)≤Veg
                akkor Ciklus amíg i≤N és E(i)≤Veg
                        i:=i+1
                      Ciklus vége
                      Ha i≤N akkor mego:=mego+1
             különben j:=M+1
  Ciklus vége
  Ha i≤N és E(i)>Veg akkor mego:=0
Eljárás vége.
```

4. feladat: Hálózat (20 pont)

A feladat egy gráf összefüggő komponenseinek meghatározása, majd a komponensek összekötése egy-egy éllel. A komponensek meghatározásához mélységi bejárást használunk.

```
Hálózat(N,Szín,G,K,Köt):
  K:=0; Szín(x):=(Fehér,…,Fehér)
  Ciklus x=1-től N-ig
    Ha Szín(x)=Fehér akkor Bejár(x); K:=K+1; Köt(K):=x
  Ciklus vége
Eljárás vége.
```

```
Bejár(x):
  Szín(x):=Fekete
  Ciklus i=1-től KiFok(x)-ig
    q:=G(x,i); Ha Szín(q)=Fehér akkor Bejár(q)
  Ciklus vége
Eljárás vége.
```

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Barátok (15 pont)

A baráti kapcsolatokból részgráfokat építünk fel, ebben a gráfban mindenkitől oda vezet él, aki neki a legszimpatikusabb. A beolvasott szimpátia mátrix minden sorából a maximális értékű elem indexére van szükség (fel(i)), amit beolvasáskor előállítunk.

Első lépésként minden elemre a legszimpatikusabbat állítsuk át arra, aki a neki legszimpatikusabbnak a legszimpatikusabb!

```
Barátok(N,fel,k):
  Ciklus i=1-től N-ig
    Ha fel(i)≠i akkor s:=fel(i)
                      Ciklus j=1-től N-ig
                        Ha fel(j)=i akkor fel(j):=s
                      Ciklus vége
  Ciklus vége
```

Ezután számoljuk meg, hogy az így kialakult fák gyökerében melyik elem hányszor fordul elő!

```
  db:=(0,…,0)
  Ciklus i=1-től N-ig
    db(fel(i)):= db(fel(i))+1
  Ciklus vége
```

Annyi baráti csoport van, ahány db-beli elem nem 0 értékű:

```
  k:=0
  Ciklus i=1-től N-ig
    Ha db(i)>0 akkor k:=k+1
  Ciklus vége
Eljárás vége.
```

2. feladat: Részhalmazok (15 pont)

A tt(i) jelentse azt, hogy az i-edik tanuló hányadik nyelvet választotta, az uu(i) pedig azt, hogy hányadik sportágat!

Először határozzuk meg, hogy adott nyelvhez (tt(i)) mely sportágak tartoznak!

```
Részhalmazok(N,
  Ciklus i=1-től M-ig
    t(tt(i)):=t(tt(i))∪{uu(i)};
  Ciklus vége
```

A feladat megoldása ezek után azon nyelvek száma, amelyekhez tartozó sportágakat mások nem választottak:

```
  db:=0
  Ciklus i=1-től N-ig
    Ha t(i)≠{} és nincsmáshol(i) akkor db:=db+1
  Ciklus vége
Eljárás vége.
```

Azt kell még megnézni, hogy az i-edik nyelvet választották-e olyan sportágbeliek, akik másik nyelvet is választottak!

```
nincsmáshol(i):
  j:=1; Ha j=i akkor j:=j+1
  Ciklus amíg j≤N és t(i)∩t(j)={}
    j:=j+1; Ha i=j akkor j:=j+1
  Ciklus vége
  nincsmáshol:=(j>n)
Függvény vége.
```

3. feladat: Vásárlás (15 pont)

Először számoljuk ki, hogy hány vásárlás kell, hogy az m-edik állomásra érve pontosan j doboz maradjon a hátizsákban! Megfigyelhető, hogy ehhez elég az m-1-edik állomás adatait ismerni, azaz a db mátrix első indexe 0 és 1 lesz.

Az első állomáson biztosan vásárolunk, legfeljebb annyit, amennyi van.

A megoldás az az érték lesz, ahány vásárlás ahhoz kell, hogy az utolsó állomásra 0 doboz maradjon a hátizsákban!

```
Vásárlás(N,a,K,db):
  Ciklus i=0-tól K-ig
    ha i≤a(1).db akkor db(0,i):=1 különben db(0,i):=1000000
                                   {az első állomáson vásárolunk}
  Ciklus vége
  m:=1
  Ciklus i=2-től N+1-ig
    Ciklus j=0-tól K-ig
      Ha j+a(i-1).tav≤K
         akkor l:=0; db(m,j):=db(1-m,j+a(i-1).tav)
               Ciklus amíg l≤a(i-1).db és j-l+a(i-1).tav≥0)
                 db(m,j):=min(db(m,j),
                              1+db(1-m,j-l+a(i-1).tav))
                 l:=l+1
               Ciklus vége
      különben db(m,j):=1000000
    Ciklus vége
    m:=1-m
  Ciklus vége
Eljárás vége.
```

4. feladat: Színezés (15 pont)

A jó színezések számát Fibonacci számokkal írhatjuk le, csak a kezdőszámokat kell kicsit eltolni. Ha az i-edik emelet fehér színű, akkor a feladat visszavezethető az i+1-edik emelettől való festésre. Ha az i-edik emelet piros, akkor az i+1-edik biztosan fehér és a feladat visszavezethető az i+2. emelettől festésre.

```
Színezés(N,db,szín):
  fib(-1):=1; fib(0):=2; fib(1):=3
  Ciklus i=2-től N-ig
    fib(i):=fib(i-1)+fib(i-2)
  end
  db:=fib(N)
```

A K. színezés előállításához azt kel tudni, hogy fehérrel kezdődő (azaz 0-s értékű) színezésből pontosan Fib(n-1) van, pirossal kezdődőből pedig Fib(N-2).

```
  j:=N-1; i:=0
  Ciklus amíg i≤N
    Ha K<fib(j) akkor szín(i):=0; j:=j-1; i:=i+1
             különben szín(i):=1; szín(i+1):=0; k:=k-fib(j)
                      j:=j-2; i:=i+2
  Ciklus vége
Eljárás vége.
```

5. feladat: Két útvonal (15 pont)

Tegyük fel, hogy van közös belső pont nélküli útvonal 1-ből U-ba és 1-ből V-be. Legyenek ezek:

$1=a_1 \rightarrow a_2 \rightarrow \ldots \rightarrow a_k=U$

$1=b_1 \rightarrow b_2 \rightarrow \ldots \rightarrow b_l=V$

Képezzük a következő, számpárokból álló sorozatot.

A sorozat első elem (1,1)

Ha a sorozat utolsó eleme (a_i, b_j), akkor a következő elem (a_{i+1}, b_j), ha $a_{i+1} < b_{j+1}$, egyébként pedig legyen (a_i, b_{j+1}).

Az utolsó elem (U,V).

Ez az összefüggés azt sugallja, hogy számpárok alkotta gráfban kell utat keresni.

Tekintsük azt a gráfot, amelynek pontjai (a,b) számpárok, ahol 1≤ a,b ≤ N. (a,b) → (aa,bb) akkor és csak akkor legyen él, ha

bb=b és aa>max(a,b) és van járat a-ból aa-ba, vagy aa=a és bb>max(a,b) és van járat b-ből bb-be.

Az előbbiekből következik, hogy ha van közös belső pont nélküli útvonal 1-ből U-ba és 1-ből V-be, akkor van ebben a gráfban (1,1)-ből (U,V)-be vezető út. Fordítva, ha van ebben a gráfban (1,1)-ből (U,V)-be vezető út, akkor van közös belső pont nélküli útvonal 1-ből U-ba és 1-ből V-be, hiszen a párok első, illetve második komponenseit véve (elhagyva a többszörös előfordulásokat), közös belső pont nélküli utakat kapunk.

```
Program:
  Beolvas(n,U,V,G,KiFok); E(x,y):=(Hamis,…,Hamis)
  Keres(1,1)
  Ha nem E(U,V) akkor Ki: 0,0
  különben n1:=1; x:=U
           Ciklus amíg x≠1
             Ut1(n1):=x; x:=P1(x); n1:=n1+1
           Ciklus vége
           Ut1(n1):=1; n2:=1; x:=V
           Ciklus amíg x≠1
             Ut2(n2):=x; x:=P2(x); n2:=n2+1
           Ciklus vége
           Ut2(n2):=1; Ki: n1,n2
           Ciklus i=n1-től 1-ig
             Ki: Ut1(i)
           Ciklus vége
           Ciklus i=n2-től 1-ig
             Ki: Ut2(i)
           Ciklus vége
Program vége.
```

```
Beolvas(n,U,V,G,KiFok):
  Be: n,U,V,m; KiFok():=(0,…,0)
  Ciklus i=1-től m-ig
    Be: x,y; KiFok(x):=KiFok(x)+1; G(x,KiFok(x)):=y
  Ciklus vége
Eljárás vége.
```

```
Eljárás Keres(x,y):
  Ha x≠U vagy y≠V akkor
    E(x,y):=Igaz; xymax:=x
    Ha y>x akkor xymax:=y
    Ciklus i=1-től KiFok(x)-ig
      q:=G(x,i)
      Ha xymax<q és nem E(q,y) akkor
        P1(q):=x; Keres(q,y); Ha E(U,V) akkor kilép
    Ciklus vége
    Ha E(U,V) akkor Kilép
    Ciklus i=1-től KiFok(y)-ig
      q:=G(y,i)
      Ha xymax<q és nem E(x,q) akkor
        P2(q):=y; Keres(x,q); Ha E(U,V) akkor Kilép
  Ciklus vége
Eljárás vége.
```

2007. Első forduló

Ötödik-nyolcadik osztályosok

1. feladat: Képlet (20 pont)

A. a=8; b=-5; c=5; d=-5 2+2+2+2 pont

B. `a=max(x,y)` 3 pont

`b=min(x,y)` 3 pont

`c=min(abs(x),abs(y))` 3 pont

`d=min(x,y)` 3 pont

2. feladat: Paradicsomok (19 pont)

```
A:  Z Z Z Z P Z Z Z Z Z Z Z P P Z Z Z P Z Z Z Z Z Z
   2     0   2   1   2   0 0   1   0         1
```

A jó helyen levő számokra annyi pont, amennyi a szám, a rosszakra -1 pont 9 pont

```
B:  Z Z Z P Z Z Z Z Z P P Z Z Z P Z P Z Z Z Z Z Z Z
 3 2   0     1     0 0   1   0   0     1       2
```

A jó helyen levő számokra annyi pont, amennyi a szám, a rosszakra -1 pont 10 pont

Egyik részfeladatra sem lehet 0 pontnál kevesebbet adni.

3. feladat: Kétirányból (24 pont)

A. Alfa: ROSSZ 3 pont

Béta: 2 3 pont

B. Konkrét példa (tetszőleges, az általános szabálynak megfelelő szó) 4 pont

Általános megfogalmazás: tükörszavakra, azaz olyanokra, amelyek elölről és hátulról olvasva azonosak 5 pont

C. Konkrét példa (tetszőleges, az általános szabálynak megfelelő szó) 4 pont

Általános megfogalmazás: olyan szavakra, amelyekben elölről haladva van olyan sorszámú betű, amelyik megegyezik a hátulról vett ugyanilyen sorszámú betűvel 5 pont

4. feladat: Két index (37 pont)

A. Első: A=2, B=5 2+2 pont

Második: A=2, B=4 2+2 pont

Harmadik: A=6, B=4 2+2 pont

B. Minden olyan vektor jó, amelyben csupa egyforma érték szerepel 5 pont

C. Minden olyan vektor jó, amelyben a maximum a legelső helyen van, és nincs vele egyenlő 5 pont

D. Első: A a legnagyobb elem indexe, B a nála kisebbek közül a legnagyobb, 2+2 pont
több egyforma esetén A a legelső legnagyobb indexe 1 pont

Második: A a legnagyobb elem indexe, B a többi közül a legnagyobbé, 2+2 pont
több egyforma esetén A a legelső legnagyobb indexe, B a következőé 1 pont

Harmadik: A a legnagyobb elem indexe, B az előtte levők közül a legnagyobbé, 2+2 pont
több egyforma esetén A a legutolsó legnagyobb indexe, B az előzőé 1 pont

Kilencedik-tizedik osztályosok

1. feladat: Hamming kód (14 pont)

A. 1100011 hibás 2 pont

1010110 hibás 2 pont

1111111 helyes 2 pont

0101010 helyes 2 pont

0110011 helyes 2 pont

Ha valamelyiket nem jól adja meg, akkor arra -1 pont jár, az A részfeladat összpontszáma nem lehet negatív!

B. 1001 → 1001100 2 pont

1010 → 1010101 2 pont

2. feladat: Számjegyek vizsgálata (23 pont)

A. Alfa: k=3, v=5, h=3 2+2+2 pont

Béta: k=3, h=4 2+2 pont

B. Alfa: A k lesz a leghosszabb azonos elemekből álló szakasz (széria) kezdete, v a vége, h pedig a hossza 3+3+2 pont

Béta: A k lesz legtöbbször előforduló számjegy, h pedig az előfordulásai száma 3+2 pont

A fentiekkel ekvivalens más megfogalmazás is elfogadható.

3. feladat: Unió (17 pont)

A. Helyes táblázatra lépésenként (azaz soronként) 1 pont, összesen 7 pont

5	-1	-1	-1	5	-1	-1	-1	-1	-1
5	4	-1	4	5	-1	-1	-1	-1	-1
5	4	-1	4	5	-1	-1	9	9	-1
5	4	-1	4	5	-1	-1	5	5	-1
5	4	-1	4	5	-1	-1	5	5	-1
5	4	-1	4	5	6	6	5	5	-1
4	4	-1	4	4	6	6	4	4	-1

B. Ha T(A)=T(B) volt, azaz már ugyanabban a részhalmazban voltak 5 pont

C. Kell egy újabb elágazáság a ciklus elé, a ciklust csak akkor kell végrehajtani, ha T(A)≠T(B) 5 pont

4. feladat: Multihalmaz (25 pont)

A. Első: K=2, C=(kecske,ló) 2+2 pont

Második: K=9, C=(kecske,kecske,ló,ló,ló,nyúl,nyúl,szamár,szamár) 2+2 pont

B. K=7, C=(kecske,kecske,ló,ló,ló,nyúl,nyúl) 2+3 pont

C. Első: Csak olyanok, amelyek mindkettőben megvannak, annyian, amennyi a két vektorban levő elemszámuk minimuma 5 pont
(azaz az A és a B multihalmaz metszete)

Második: Olyanok, amelyek valamelyikben szerepelnek, a mindkettőben szereplők közül annyian, amennyi a két vektorban levő elemszámuk maximuma 5 pont
(azaz az A és a B multihalmaz uniója)

D. Ha a két vektor egyenlő (azaz A=B és N=M) 2 pont

5. feladat: Kép (21 pont)

A. KÉP1: 3 művelet 3 pont

KÉP2: 4 művelet 3 pont

B. KÉP1: `FBB` (a három betű minden sorrendje jó) 3 pont

KÉP2: `FFJJ` (a négy betű minden sorrendje jó) 3 pont
Ha jó műveletsort ad, de nem minimális lépésszámút, akkor a 1-1 pont adható.

C. C1: `JLL` (a három betű minden sorrendje jó) 3 pont

C2: `JJ` 3 pont

C3: `FB` (a két betű minden sorrendje jó) 3 pont

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Hamming kód (16 pont)

A. 1100011 hibás 1 pont

1010110 hibás 1 pont

1111111 helyes 1 pont

0101010 helyes 1 pont

0110010 hibás 1 pont

Ha valamelyiket nem jól adja meg, akkor arra -1 pont jár, az A részfeladat összpontszáma nem lehet negatív!

B. 1100011 helyesen 1000011, azaz a b_2 a hibás 1 pont

mert b_5b_7 értéke nem jó, azaz a b_6 képletében szereplő bitek helyesek 2 pont

1010110 helyesen 0010110, azaz a b_1 a hibás 1 pont

mert b_6b_7 értéke nem jó, azaz a b_5 képletében szereplő bitek helyesek 2 pont

0110010 helyesen 0110011, azaz a b_7 a hibás 1 pont

mert b_5b_6 értéke jó, azaz az első 4 bitnek jónak kell lenni 2 pont

C. 1001 → 1001100 1 pont

1010 → 0101010 1 pont

2. feladat: Számjegyek vizsgálata (20 pont)

A. Alfa: k=1, v=4, h=3 2+2+1 pont

Béta: k=2, v=7, h=5 2+2+1 pont

B. Alfa: A `k` lesz a leghosszabb szigorúan monoton növekvő szakasz kezdete (ha növekvően indul az X vektor), `v` a vége, `h` pedig a hossza (h=v-k) 2+2+1 pont

Béta: A `k` lesz az azonos számjegyek közötti leghosszabb hézag kezdete, `v` a vége, `h` pedig a hossza (h=v-k) 2+2+1 pont

A fentiekkel ekvivalens más megfogalmazás is elfogadható, azaz pl.

- h eggyel kisebb lesz a leghosszabb ... szakaszban levő értékek számánál;
- (Alfa): h lesz a leghosszabb egymás utáni növekedések száma;
- (Béta): h eggyel nagyobb lesz, mint a legtöbb, két egyforma számjegy között előforduló, tőlük különböző számok darabszáma

3. feladat: Táblázat (25 pont)

A. Helyes táblázatra lépésenként (azaz soronként) 1 pont, összesen 7 pont

100	-1	-1	-1	-1	-1	-1	-1	-1	-1
100	-1	22	-1	-1	-1	-1	-1	-1	-1
100	10	22	-1	-1	-1	-1	-1	-1	-1
100	10	22	30	-1	-1	-1	-1	-1	-1
100	10	22	30	11	-1	-1	-1	-1	-1
100	-2	22	30	11	-1	-1	-1	-1	-1
100	21	22	30	11	-1	-1	-1	-1	-1

B. Ha a táblázat megtelt, azaz 10 elem van benne 5 pont

C. Ha a táblázatban nincs benne a keresett A elem 5 pont

D. Minden olyan megoldás jó mindkét eljárásra, amely figyeli, hogy a kezdő i értékéről körbe érünk-e, s ilyenkor kilép a ciklusból 4+4 pont

4. feladat: Multihalmaz (24 pont)

A. Első: K=4, C=(ló,ló,nyúl,nyúl) 2+2 pont

Második: K=7, C=(kecske,ló,ló,nyúl,nyúl,szamár,szamár) 2+2 pont

B. K=5, C=(kecske,ló,ló,nyúl,nyúl) 1+1 pont

C. K=3, C=(kecske,szamár,szamár) 1+1 pont

D. Első: Olyanok A-ból, amelyek B-be nincsenek, vagy olyanok, amelyek megvannak B-ben is, ezekből annyian, amennyivel A-ban többen vannak 5 pont

(azaz az A és a B multihalmaz különbsége)

Második: Olyanok, amelyek vagy csak A-ban, vagy csak B-ben szerepelnek, a mindkettőben szereplők közül annyian, amennyivel az egyikben többen vannak, mint a másikban 5 pont

(azaz az A és a B multihalmaz szimmetrikus differenciája)

E. Ha B része A-nak (azaz B minden eleme benne van A-ban) 2 pont

5. feladat: Kép (15 pont)

A. KÉP1: 2 művelet 2 pont

KÉP2: 2 művelet 2 pont

B. KÉP1: `BV` vagy `JH` vagy `VJ` vagy `HB` (elég az egyiket megadni) 2 pont

KÉP2: `VB` vagy `HJ` vagy `JV` vagy `BH` (elég az egyiket megadni) 2 pont

Ha jó műveletsort ad, de nem minimális lépésszámút, akkor a pontszám fele adható.

C. C1. `JH` (vagy `BV` vagy `VJ` vagy `HB`) 2 pont

C2. `K` 3 pont

C3. `HJ` (vagy `VB` vagy `JV` vagy `BH`) 2 pont

2007. Második forduló

Ötödik-nyolcadik osztályosok

1. feladat: Japán kalendárium (20 pont)

Egyszerű képlettel számolható az évhez tartozó szín:

```
Japánnaptár(év):
  kezd=1983-4*60
  év:=(év-kezd) mod 60; i:=év mod 10
  Ha i=1 vagy i=2 akkor Ki:'Zöld év'
  különben ha i=3 vagy i=4 akkor Ki:'Piros év'
  különben ha i=5 vagy i=6 akkor Ki:'Sárga év'
  különben ha i=7 vagy i=8 akkor Ki:'Fehér év'
  különben ha i=9 vagy i=0 akkor Ki:'Fekete év'
Eljárás vége
```

2. feladat: Szilveszter (35 pont)

A napszám() tömbben az egyes hónapok napszámát tároljuk. A februári napszámot szökőévekben át kell írni! A feladat tulajdonképpen egy speciális, vegyes alapú számrendszerbeli szám kivonását jelenti. Arra kell még figyelni, hogy egyes helyiértékeken (hó, nap) 1 a legkisebb számjegy!

```
Szilveszter(év,hó,nap,óra,perc,mp):
  Ha év mod 4=0 és év mod 100≠0 vagy év mod 400=0
     akkor napszám(2):=29
  Ha mp>0 akkor atv:=1; mp:=60-mp
  Ha atv=0 akkor ha perc>0 akkor atv:=1; perc:=60-perc
        különben perc:=60-atv-perc
  Ha atv=0 akkor ha óra>0 akkor atv:=1; óra:=24-óra
        különben óra:=24-atv-óra
  Ha atv=0 akkor
     ha nap>1 akkor atv:=1; nap:=napszám(hó)-nap
           különben nap:=0
  különben nap:=napszám(hó)+1-atv-nap
  Ha atv=0 akkor ha hó>1 akkor atv:=1; hó:=13-ho
                      különben hó:=0
           különben hó:=13-atv-hó
  Ha atv=0 akkor év:=1 különben év:=0
Eljárás vége.
```

3. feladat: Mozaikszó (20 pont)

Azt kell megnézni, hogy a mozaikszó milyen részsorozata-e szövegnek! (Biztosan van megoldás.)

```
Mozaikszó(szöveg,mozaik):
  j:=1
  Ciklus i=1-től hossz(szöveg)-ig
    Ha nagybetűs(mozaik(j))=nagybetűs(szöveg(i))
      akkor Ki: i; j:=j+1
  Ciklus vége
Eljárás vége.
```

Kilencedik-tizedik osztályosok

1. feladat: Hidak (16 pont)

A h mátrix a szigetek bal és jobboldali partjának helyét tartalmazza. A t mátrix első oszlopában a szigetek közötti távolságot, a másodikban pedig a szigetek közötti hídszakaszok számát tároljuk. Mivel a leghosszabb hídszakasz hosszát kell minimalizálni, az első pillért a leghosszabb szigetek közötti szakaszra kell tenni. A következőt vagy a következő leghosszabb szakaszra, vagy pedig egy olyanra, ahol már volt pillér, így csökkentve a leghosszabb szükséges hídszakasz hosszát.

```
Hidak(K,N,t,h):
  Ciklus i=2-től N-ig
    t(i-1,1):=h(i,1)-h(i-1,2); t(i-1,2):=1
  Ciklus vége
  Ciklus i=1-től K-ig
    max:=1
    Ciklus j=2-től N-1-ig
      Ha t(j,1)*t(max,2)>t(max,1)*t(j,2) akkor max:=j
    Ciklus vége
    t(max,2):=t(max,2)+1
  Ciklus vége
Eljárás vége.
```

2. feladat: Szövegkereső (15 pont)

Először vizsgáljuk azt az esetet, amikor az s szöveg csillaggal kezdődik, majd azt, amikor csillaggal végződik!

Az `igen` és a `nem` eljárások be is fejezik a program futását.

```
Rész(s,t):
  Ha s(1)='*'
     akkor Csillaggal kezdődik(s,t)
  Ha s(length(s))='*' akkor Csillaggal végződik(s,t)
```

Ha az első szóban kérdőjelet találunk ott, ahol a másodikban kötött betű van, akkor van olyan szó, amit csak az első talál meg (pl. a csillag helyére a másik szó karakterétől különbözőt kell tenni). Ha az elsőben csillag van ott, ahol a másodikban nem csillag, akkor is van ilyen szó.

```
  i:=1
  Ciklus amíg i≤hossz(s)) és nem (s(i)='?' és t(i)≠'?' és
             t(i)≠'*') és nem (s(i)='*' és t(i)≠'*')
    i:=i+1
  Ciklus vége
  Ha i≤hossz(s) akkor nem
```

Ha azonos pozíción különböző betű van, vagy az elsőben kérdőjel van ott, ahol a másodikban nem, akkor megint van olyan szó, amit csak az első talál meg.

Ha a másodikban van csillag, akkor azonos szavakat találnak meg, ha az előtte és a mögötte levő részük is jó.

```
  i:=1
  Ciklus amíg i≤hossz(s) és t(i)≠'*' és jó(s(i),t(i))
    i:=i+1
  Ciklus vége
  Ha i>hossz(s) akkor ha i>hossz(t) akkor igen
                                   különben nem
  j:=hossz(t)
  Ciklus amíg j>1 és t(j)≠'*' és jó(s(j),t(j))
    j:=j-1
  Ciklus vége
  Ha j<1 vagy j=i akkor igen különben nem
Eljárás vége.
```

Ha az első csillaggal kezdődik, a második pedig nem, akkor lehet találni olyan szót, amit csak az első talál meg (pl. a * helyére olyan karaktert teszünk, ami különbözik a második szó első karakterétől).

Ha mindkettő csillaggal kezdődik, de a hosszuk különböző, akkor is lehet találni ilyen szót. Ha a hosszuk egyforma, akkor karakterenként meg kell vizsgálni őket. Ha azonos pozíción különböző betű van, vagy az elsőben kérdőjel van ott, ahol a másodikban nem, akkor megint van olyan szó, amit csak az első talál meg.

```
Csillaggal kezdődik(s,t):
  Ha t(1)='*'
     akkor Ha hossz(s)=hossz(t)
              akkor i:=2
                    Ciklus amíg i≤hossz(s)) és jó(s(i),t(i))
                       i:=i+1
                    Ciklus vége
                    Ha i>hossz(s) akkor igen különben nem
           különben nem
  különben nem
Eljárás vége.
```

Itt ugyanazt kell tenni, mint a csillaggal kezdődő esetben.

```
Csillaggal végződik(s,t):
  Ha t(hossz(t))='*'
     akkor Ha hossz(s)=hossz(t)
              akkor i:=hossz(s)-1
                    Ciklus amíg i≥1 és jó(s(i),t(i))
                       i:=i-1
                    Ciklus vége
                    Ha i<1 akkor igen különben nem
           különben nem
  különben nem
Eljárás vége.
```

```
jó(a,b)
  jó:=(a≠'?' és b='?' vagy a=b)
Függvény vége.
```

<u>3. feladat</u>: Hírek (16 pont)

Első lépésként számoljuk ki azt mindenkire, hogy kinek tud közvetve üzenetet küldeni:

```
Hírek(N,t,dba,dbb):
  Ciklus k=1-től N-ig
    Ciklus i=1-től N-ig
      Ciklus j=1-től N-ig
        t[i,j]:=t[i,j] vagy t[i,k] és t[k,j]
      Ciklus vége
    Ciklus vége
  Ciklus vége
```

Az első részfeladat megoldása azon párok száma, akik kölcsönösen tudnak üzenetet küldeni egymásnak:

```
  dba:=0
  Ciklus i=1-től N-1-ig
    Ciklus j=i+1-től N-ig
      Ha t[i,j] és t[j,i] akkor dba:=dba+1
    Ciklus vége
  Ciklus vége
```

A második részfeladat megoldása azon sorok száma, ahol mindenkinek lehet üzenetet küldeni:

```
  dbb:=0
  Ciklus i=1-től N-ig
    j:=1
    Ciklus amíg j≤n és t[i,j]
      j:=j+1
    Ciklus vége
    Ha j>n akkor dbb:=dbb+1
  Ciklus vége
Eljárás vége.
```

<u>4. feladat</u>: Utazó (14 pont)

A feladatban adott két rendezett intervallumsorozat, ahol az intervallumokhoz nevet is rendeltünk. Meg kell adni azokat a neveket, amelyek intervallumai átfedik egymást!

Azaz ez egy speciális összefuttatási feladat.

```
Utazó(N,a,M,b,K,c):
  k:=0; i:=1; j:=1; a(n+1].tol:=maxint; b(m+1].tol:=maxint
  Ciklus amíg i<n+1 vagy j<m+1
    Ha a(i).ig<b(j).tol akkor i:=i+1
    különben ha b(j).ig<a(i).tol akkor j:=j+1
    különben ha a(i).ig≤b(j).ig
                akkor Ha a(i).varos=b(j).varos
                         akkor k:=k+1; c(k):=a(i).varos
                               i:=i+1; j:=j+1
                      különben i:=i+1
    különben ha b(j).ig≤a(i).ig
                akkor Ha a(i).varos=b(j).varos
                         akkor k:=k+1; c(k):=a(i).varos
                               i:=i+1; j:=j+1
                      különben j:=j+1
  Ciklus vége
Eljárás vége.
```

<u>5. feladat</u>: Jelentés (14 pont)

A feladatban egy fát építünk fel (alulról felfelé), ahol minden elemtől a főnöke felé vezet él. A megoldás azon elemek megadása, amelyek a fa gyökerétől pontosan K távolságra vannak.

A 0 fokú elemek nem főnökök, náluk kezdjük a bejárást.

```
Jelent(N,Fok,Táv):
  Táv(1):=0; Mego:=0
  Ciklus x=1-től N-ig
    Ha Fok(x)=0 akkor y:=Mélység(x)
  Ciklus vége
Eljárás vége.
```

Ha valakire már kiszámoltuk a mélységet, akkor ott a rekurzív számítással leállunk. Ha még nem számoltuk ki, akkor kiszámoljuk, és ha pontosan K a mélysége, akkor számolunk az eredményben.

```
Mélység(x):
  Ha Táv(x)<0 akkor Táv(x):=1+Mélység(Főnöke(x))
                    Ha Táv(x)=K akkor Mego:=Mego+1
  Mélység:=Táv(x)
Függvény vége.
```

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Jegyek (14 pont)

A feladat megoldása: a pontszámok között meg kell találni a 4 legnagyobb rést! Tehát számoljuk meg, hogy mely pontszámból hány van! Ezután adjuk meg a 0 darabszámú pontszámokat tartalmazó intervallumokat! Harmadik lépésként ezeket rendezzük csökkenő sorrendbe, majd az első négyet a felső határuk szerint növekvőbe!

```
Jegyek(N,t,M,h):
  Ciklus i=1-től N-ig
    d(t(i)):=d(t(i))+1
  Ciklus vége
  db:=0; j:=0
  Ciklus i=1-től M-1-ig
    Ha d(i)>0 akkor j:=i
    Ha d(i)=0 és d(i+1)>0
       akkor Ha j>0 akkor db:=db+1; h(db,1):=i; h(db,2):=i-j
  Ciklus vége
  Ciklus i=1-től 4-ig
    Ciklus j=i+1-től db-ig
      Ha h(i,2)<h(j,2) akkor Csere(h(i),h(j)
    Ciklus vége
  Ciklus vége
  Ciklus i=1-től 3-ig
    Ciklus j=i+1-től 4-ig
      Ha h(i,1)>h(j,1) akkor Csere(h(i),h(j))
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

2. feladat: Szövegkereső (16 pont)

Ha a keresőszavak nem csillaggal kezdődnek, vagy nem csillaggal végződnek, akkor karakterenként hasonlíthatunk az első csillagig.

```
Keres(s,t,u):
  u:=''
  Ha s(1)≠'*' és t(1)≠'*'akkor
```

Nem *-gal kezdődnek: haladunk *-ig, vagy a szavak hosszáig. Ha a karakterek „megegyeznek”, akkor ha mindkettő ? volt, akkor .-ot teszünk az eredménybe, különben pedig az egyik szó megfelelő karakterét (figyelve arra, hogy en kérdőjelet tegyünk).

Igent válaszolunk, ha mindkét szó elfogy, vagy ha az egyik elfogy és a másik végén már csak egy * karakter maradt. Nemet válaszolunk, ha az egyik elfogy és a másik végén maradtak még karakterek.

Egyéb esetekben a * előtti részeket levágjuk és folytatjuk a feldolgozást.

```
    i:=1
    Ciklus amíg i≤hossz(s) és i≤hossz(t) és
          s(i)≠'*' és t(i)≠'*' és jó(s(i),t(i))
      Ha s(i)='?' akkor
         Ha t(i)='?' akkor u:=u+'.' különben u:=u+t(i)
      különben u:=u+s(i)
      i:=i+1
    Ciklus vége
    Ha i>hossz(s) és i>hossz(t) akkor igen
    Ha i>hossz(s) akkor Ha t(i..hossz(t))='*' akkor igen
                                              különben nem
    Ha i>hossz(t) akkor Ha s(i..hossz(s))='*' akkor igen
                                              különben nem
    s:=s(i..hossz(s)); t:=t(i..hossz(t))
  Elágazás vége
  v:=''
  Ha s(hossz(s))≠'*' és t(hossz(t))≠'*' akkor
```

Nem *-gal végződnek: hasonló a teendő, mint elölről haladva.

```
    i:=hossz(s); j:=hossz(t)
    Ciklus amíg i≥1 és j≥1 és s(i)≠'*' és t(j)≠'*'
                és jó(s(i),t(j))
      Ha s(i)='?' akkor
         Ha t(j)='?' akkor v:='.'+v különben v:=t(j)+v
      különben v:=s(i)+v
      i:=i-1; j:=j-1
    Ciklus vége
    Ha i<1 vagy j<1 akkor u:=u+v
    Ha i<1 és j<1 akkor igen
    Ha i<1 akkor Ha t(1..j)='*' akkor igen különben nem
    Ha j<1 akkor Ha s(1..i)='*' akkor igen különben nem
    s:=s(1..i); t:=t(1..j)
  Elágazás vége
  Ha s(1)='*' akkor
```

Az első *-gal kezdődik:

```
    Ciklus i=1-től hossz(t)-ig
       Ha t(i)='?' akkor u:=u+'.'
       különben ha t(i)≠'*' akkor u:=u+t(i)
    Ciklus vége
    Ciklus i=2-től hossz(s)-ig
      Ha s(i)='?' akkor u:=u+'.' különben u:=u+s(i)
    Ciklus vége
    u:=u+v; igen
  különben ha t(1)='*' akkor
```

A második *-gal kezdődik:

```
    Ciklus i=1-től hossz(s)-ig
      Ha s(i)='?' akkor u:=u+'.'
      különben ha s(i)≠'*' akkor u:=u+s(i)
    Ciklus vége
    Ciklus i=2-től hossz(t)-ig
      Ha t(i)='?' akkor u:=u+'.' különben u:=u+t(i)
    Ciklus vége
    u:=u+v; igen
  különben nem
Eljárás vége.
```

Az `igen` és a `nem` eljárások be is fejezik a program futását. A megoldásuk és a `jó` függvény a második korcsoportosok hasonló feladatában szerepel.

<u>3. feladat</u>: Rendezvény (15 pont)

Először számítsuk ki az előadások befejezési idő szerinti – növekvő – sorrendjét! Ehhez egy számlálva szétosztó rendezést használunk.

```
Rendez(N,Ig,mA,A,mB,B):
  S:=(0,…,0)
  Ciklus i=1-től N-ig
    S(Ig(i).vég):=S(Ig(i).vég)+1
  Ciklus vége
  Ciklus i=1-től Maxidő-ig
    S(i):=S(i)+S(i-1)
  Ciklus vége
  Ciklus i=1-től N-ig
    x:=S(Ig(i).vég); R(x):=Ig(i).id
    S(Ig(i).vég):=S(Ig(i).vég)-1
  Ciklus vége
```

Ezután vegyük a két termet és minden előadást abba a terembe osszunk be, amelyikben – ha befér – később ér véget az előző előadás!

```
  mA:=0;mB:=0; Avég:=0; Bvég:=0
  Ciklus i=1-től N-ig
    ii:=R(i)
    Ha Avég≥Bvég akkor
       Ha Avég<Ig(ii).kezdet akkor
          mA:=mA+1; A(mA):=Ig(ii).id; Avég:=Ig(ii).vég
       különben ha Bvég<Ig(ii).k akkor
          mB:=mB+1; B(mB):=Ig(ii).id; Bvég:=Ig(ii).vég
    különben
       Ha Bvég<Ig(ii).kezdet akkor
          mB:=mb+1; B(mB):=Ig(ii).id; Bvég:=Ig(ii).vég
       különben ha Avég<Ig(ii).kezdet akkor
         mA:=mA+1; A(mA):=Ig(ii).id; Avég:=Ig(ii).vég
  Ciklus vége
Eljárás vége.
```

<u>4. feladat</u>: Hálózat (15 pont)

Ha a K pontból bejárjuk a gráfot, akkor megkapjuk azokat a pontokat, ahova K-ból el lehet jutni. Ha ezután a transzponált gráfot járjuk be, akkor megkapjuk azokat a pontokat, ahonnan K-ba el lehet jutni. A megoldás: azon pontok, ahova az első bejárában eljutottunk, de a másodikban nem. (A gráfot szomszédsági listával ábrázoljuk.)

```
Hálózat(N,G,Gt,K,Szín):
  Szín:=(Fehér,…,Fehér)
  MélyBejár(G,K,Fehér,Fekete)
  MélyBejár(Gt,K,Fekete,Piros)
Eljárás vége.
```

```
MélyBejár(G,u,Régi,Új)
  Szín(u):=Új; Él:=G(u)
  Ciklus amíg Él≠sehova {minden u->v élre}
    v:=Él^.pont
    Ha Szín(v)=Régi akkor MélyBejár(G,v,Régi,Új)
    Él:=Él^.következő
  Ciklus vége
Eljárás vége.
```

5. feladat: Játék (15 pont)

Ez egy egyszerű dinamikus programozási feladat, amelyben minden (x,y) mezőre kiszámoljuk azt a T(x,y) értéket, amit maximum el lehet érni, amikor abból a mezőből indulunk. Kezdetben a T mátrix szélét töltsük fel -1 értékekkel, hogy a táblázat szélével ne kelljen külön foglalkoznunk! A csapdák is -1 értékűek, amire nem szabad lépni.

```
Játék(N,M,G,T):
  T(N+1,M):=0;  T(N,M+1):=0
  Ciklus x=N-től 1-ig -1-esével
    Ciklus y=M-től 1-ig -1-esével
      Ha G(x,y)=-1 akkor T(x,y):=-1
      különben Ha T(x+1,y)=-1 akkor Le:=-1
                               különben Le:=G(x,y)+T(x+1,y)
               Ha T(x,y+1)=-1 akkor Jobb:=-1
                               különben Jobb:=G(x,y)+T(x,y+1)
               Ha Le>Jobb akkor T(x,y):=Le
                           különben T(x,y):=jobb
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

2007. Harmadik forduló

Ötödik-nyolcadik osztályosok

1. feladat: Négyzetek (18 pont)

Meg kell találni az N előtti legnagyobb négyzetszámot, majd ezt levonva N-ből újra kezdeni ezt az eljárást. Ezt mindaddig végezzük, amíg N-re 0-t nem kapunk.

```
Négyzet(N):
  k:=1
  Ciklus amíg (k+1)*(k+1)≤N
    k:=k+1
  Ciklus vége
  Ciklus amíg N>0
    Ki: k; N:=N-k*k
    Ciklus amíg k*k>N
      k:=k-1
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

<u>2. feladat</u>: Bűvös négyzet (30 pont)

Ha minden sor és oszlop összege egyforma, akkor a bűvös négyzet jó. Ha az i-edik sor és a j-edik oszlop összege nem azonos a többiekével, akkor az (i,j) indexű hely a hibás.

```
Bűvös(N,b,jó,i,j,érték):
  Ciklus i=1-től N-ig
    s(i):=0; o(i):=0
    Ciklus j=1-től N-ig
      s(i):=s(i)+b(i,j); o(i):=s(i)+b(j,i)
    Ciklus vége
  Ciklus vége
  Ha s(1)=s(2) akkor k:=s(1) különben k:=s(3)
  i:=1
  Ciklus amíg i≤N és s(i)=k
    i:=i+1
  Ciklus vége
  jó:=i>N
  Ha i≤N akkor j:=1
             Ciklus amíg o(j)=k
               j:=j+1
             Ciklus vége
             érték:=k-s(i]+b(i,j]
Eljárás vége.
```

<u>3. feladat</u>: Nyelvek (27 pont)

```
Nyelvek(N,t,M,v):
  Ciklus i=1-től N-ig
    Ha nincs(t(i),v,m) akkor Ki: 'Illegális nyelv: ',i,t(i)
  Ciklus vége
  Ciklus i=1-től M-ig
    Ha nincs(v(i),t,n) akkor Ki: 'Nem választott: ',v(i)
  Ciklus vége
  s:=(0,…,0)
  Ciklus i=1-től N-ig
    Keres(t(i),van,j); Ha van akkor s(j):=s(j)+1
  Ciklus vége
  Ciklus i=1-től M-ig
    Ha s(i)>0 akkor Ki: v(i),s(i)
  Ciklus vége
Eljárás vége.

Keres(s,van,i):
  i:=1
  Ciklus amíg i≤M és s≠v(i)
    i:=i+1
  Ciklus vége
  keres:=i
Eljárás vége.

nincs(s,v,db):
  i:=1
  Ciklus amíg i≤db és s≠v(i)
    i:=i+1
  Ciklus vége
  nincs:=(i>db)
Függvény vége.
```

Kilencedik-tizedik osztályosok

1. feladat: Madarak (20 pont)

Számoljuk ki minden (i,j) helyre, hogy az mely madarak territóriumához tartozik!

```
Madarak(N,M,x,y,r,mad,van,max,x,y,üres):
  t:=({},…{}); db:=(0,…,0)
  Ciklus k=1-től M-ig
    Ciklus i=x(k)-r(k)-tól x(k)+r(k)-ig
      Ciklus j=y(k)-r(k)-tól y(k)+r(k)-ig
        Ha i≥=1 és i≤N és j≥1 és j≤N akkor t(i,j):=t(i,j)+(k)
                                           db(i,j):=db(i,j)+1
      Ciklus vége
    Ciklus vége
  Ciklus vége
```

Az első részfeladatban hagyjuk ki a madarak közül azokat, amelyek territóriumán más madár is lehet! Ezt akár nagyon egyszerűen, 3 ciklussal is elvégezhetjük:

```
  mad:={1..m}
  Ciklus i=1-től N-ig
    Ciklus j=1-től N-ig
      k:=1
      Ciklus amíg k≤M és k∉t(i,j)
        k:=k+1
      Ciklus vége
      Ha k≤M akkor Ha t(i,j)≠{k} akkor mad:=mad-{k}
    Ciklus vége
  Ciklus vége
```

A második részfeladatban a db mátrix maximumát kell megadni:

```
  max:=1
  Ciklus i=1-től N-ig
    Ciklus j=1-től N-ig
      Ha db(i,j)>max akkor max:=db; x:=i; y:=j
    Ciklus vége
  Ciklus vége
  van:=(max>1)
```

A harmadik részfeladatban az üres helyek számát kell megadni (`db(i,j)=0` helyett lehetne benne `t(i,j)={}` is):

```
  Ciklus i=1-től N-ig
    Ciklus j=1-től N-ig
      Ha db(i,j)=0 akkor üres:=üres+1
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

2. feladat: Vízirendőr (15 pont)

Nézzük végig a településeket! Az `eleje` változó legyen igaz értékű, ha a vízirendőrség helyét keressük. Ha az i-edik településről az e-ediket nem éri el, akkor az i-1-edikbe vízirendőrséget kell tenni. Ez persze A*K kilométert lefelé meg tud tenni, tehát csak azután kell keresni az újabb vízirendőrség helyét.

```
Vízi(N,A,B,K,t,M,v):
  t(1):=0; M:=0; e:=1; eleje:=igaz; t(N+1):=t(N)+K*A
  Ciklus i=2-től N+1-ig
    Ha eleje akkor
       Ha (t(i)-t(e))>b*k akkor
          m:=m+1; v(m):=i-1; eleje:=hamis
    Ha nem eleje akkor
       Ha (t(i)-t(v(m)))>a*k akkor e:=i; eleje:=igaz
  Ciklus vége
Eljárás vége.
```

<u>3. feladat</u>: Jegy (20 pont)

Előkészítő tevékenységként végezzük el: ha két ember ugyanazt a helyet kéri, akkor biztosan csak a több pénzt ajánlóval kell foglalkozni!

```
Jegy(N,hely,pénz,M):
  Igény:=(0,…,0)
  Ciklus i=1-től N-ig
    Ha Igény(hely(i)+1)<pénz(i)
       akkor Igény(hely(i)+1):=pénz(i); S(hely(i)+1):=i
  Ciklus vége
```

`Opt(x)` jelentse az x-edik ülőhelyig elérhető legnagyobb bevételt! Az x-1-edik helyről elvehetjük az előző igényt, ha az x-ediken levőt teljesítve nagyobb lesz a bevétel.

```
  Opt(0):=0;Opt(1):=0
  Ciklus x=2-től M-ig
    Ha Igény(x)>0
       akkor Opt(x):=Opt(x-1)
             Ha Opt(x-2)+Igény(x)>Opt(x)
                akkor Opt(x):=Opt(x-2)+Igény(x)
    különben Opt(x):=Opt(x-1)
  Ciklus vége
```

Eredményül az elérhető legnagyobb bevételt, valamint azon helyekhez tartozó ember sorszámokat kell kiírni, ahol `Opt(x)>Opt(x-1)`.

```
  Ki: Opt(M); x:=M
  Ciklus amíg x>1
    Ciklus amíg x>0 és Opt(x)=Opt(x-1)
      x:=x-1
    Ciklus vége
    Ha x>0 akkor Ki: S(x)
    x:=x-2
  Ciklus vége
Eljárás vége.
```

<u>4. feladat</u>: Ismeretség (20 pont)

Az első részfeladatban az ismereteket leíró gráfban meg kell adni azon pontokat, ahova vezet él A-ból, de onnan A-ba nem vezet él!

A második részfeladatban pedig azon pontokat kell megadni, ahova vezet út A-ból, de onnan A-ba nem vezet út! Az útról most feltesszük, hogy legalább 2 élt tartalmaz.

```
Ismer(N,is,volt):
  Ciklus i=1-től N-ig
    Ha is(a,i) akkor volt(i):=igaz
  Ciklus vége
```

```
  Ciklus k=1-től N-ig
    Ciklus i=1-től N-ig
      Ciklus j=1-től N-ig
        is(i,j):=is(i,j) vagy is(i,k) és is(k,j)
      Ciklus vége
    Ciklus vége
  Ciklus vége
  dbA:=0
  Ciklus i=1-től N-ig
    Ha is(a,i) és nem is(i,a) és volt(i)
       akkor dbA:=dbA+1; megA(dbA):=i
    Ha is(a,i) és nem is(i,a) és nem volt(i)
       akkor dbB:=dbB+1; megB(dbB):=i
  Ciklus vége
Eljárás vége.
```

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Madár (15 pont)

Minden madárral számoljuk meg, hogy hány másik madár territóriuma esik bele az ő territóriumába! Azaz minden (k,i) madárpárra nézzük meg, hogy a k-adik territóriumának egyik sarka beleesik-e az i-edik territóriumába, illetve az i-edik territóriumának egyik sarka beleesik-e a k-adik territóriumába. Ha a beleesés mind a 4 sarokra teljesül, akkor pedig az egyik madár territóriuma része a másik madár territóriumának.

```
Madár(N,M,mad,dbA,meg,max,dbC):
  fok:=(0,…,0); része:=(hamis,…,hamis)
  Ciklus k=1-től M-ig
    Ciklus i=1-től k-1-ig
      Ha eleme(mad(k,1)-mad(k,3),mad(k,2)-mad(k,3),i) vagy
         eleme(mad(k,1)+mad(k,3),mad(k,2)-mad(k,3),i) vagy
         eleme(mad(k,1)-mad(k,3),mad(k,2)+mad(k,3),i) vagy
         eleme(mad(k,1)+mad(k,3),mad(k,2)+mad(k,3),i)
      akkor fok(i):=fok(i)+1; fok(k):=fok(k)+1
            Ha eleme(mad(k,1)-mad(k,3),mad(k,2)-mad(k,3),i) és
               eleme(mad(k,1)+mad(k,3),mad(k,2)-mad(k,3),i) és
               eleme(mad(k,1)-mad(k,3),mad(k,2)+mad(k,3),i) és
               eleme(mad(k,1)+mad(k,3),mad(k,2)+mad(k,3),i)
            akkor része(k):=igaz
            Ha mad(k,1)=mad(i,1) és mad(k,2)=mad(i,2) és
               mad(k,3)=mad(i,3) akkor része(i):=igaz
      különben
      Ha eleme(mad(i,1)-mad(i,3),mad(i,2)-mad(i,3),k) vagy
         eleme(mad(i,1)+mad(i,3),mad(i,2)-mad(i,3),k) vagy
         eleme(mad(i,1)-mad(i,3),mad(i,2)+mad(i,3),k) vagy
         eleme(mad(i,1)+mad(i,3),mad(i,2)+mad(i,3),k)
      akkor fok(i):=fok(i)+1; fok(k):=fok(k)+1
            Ha eleme(mad(i,1)-mad(i,3),mad(i,2)-mad(i,3),k) és
               eleme(mad(i,1)+mad(i,3),mad(i,2)-mad(i,3),k) és
               eleme(mad(i,1)-mad(i,3),mad(i,2)+mad(i,3),k) és
               eleme(mad(i,1)+mad(i,3),mad(i,2)+mad(i,3),k)
            akkor része(i):=igaz
    Ciklus vége
  Ciklus vége
```

Az A részfeladat megoldása a 0 fokú madarak megadása.

```
  dbA:=0
  Ciklus i=1-től M-ig
    Ha fok(i)=0 akkor dbA:=dbA+1; meg(dbA):=i
  Ciklus vége
```

A B részfeladat megoldása a maximális fokú madár.

```
  max:=1
  Ciklus i=2-től M-ig
    Ha fok(i)>fok(max) akkor max:=i
  Ciklus vége
```

A harmadik részfeladat megoldása a része vektorban van?

```
  dbC:=0
  Ciklus i=1-től N-ig
    Ha része(i) akkor dbC:=dbC+1
  Ciklus vége
Eljárás vége.
```

<u>2. feladat</u>: Málna (15 pont)

A megoldás ötlete: minden nap a legolcsóbbat kell megvenni először, utána a következő legolcsóbbat, … Ha valamelyik áron adott málna mind már nem fér bele a K kilogrammos napi feldolgozásba, akkor csak annyit kell megvenni belőle, hogy a K kilogrammot elérjük!

Hatékonyabb megoldást is kaphatunk, ha az árusokat rendezzük ár szerint növekvő sorrendbe.

```
Málna(s,N,t,M,ár,u,v):
  Ciklus i=1-től N-ig
    s:=s+t(i).menny
  Ciklus vége

  ár:=0; M:=1; t(0).ár:=maxint; menny:=0; db:=0
  Ciklus amíg s>0
    ha menny=k akkor db:=db+1; v(db):=0; menny:=0; M:=M+1
    min:=0
    Ciklus i=1-től N-ig
      Ha t(i).menny>0 és t(i).ár<t(min).ár akkor min:=i
    Ciklus vége
    Ha k-menny≥t(min).menny
       akkor db:=db+1; v(db):=min; u(db):=t(min).menny
             menny:=menny+t(min).menny; s:=s-t(min).menny
             ár:=ár+t(min).menny*(t(min).ár-(M-1)*f)
             t(min).menny:=0
    különben ár:=ár+(k-menny)*(t(min).ár-(M-1)*f)
             db:=db+1; v(db):=min; u(db):=k-menny
             t(min).menny:=t(min).menny-(k-menny)
             s:=s-(k-menny), menny:=k
  Ciklus vége
Eljárás vége.
```

<u>3. feladat</u>: DNS (15 pont)

Bontsuk részproblémákra a kiindulási problémát. Legyen n=Hossz(S1), m=Hossz(S2). Minden i-re és j-re, $0 \leq i \leq n$, és $0 \leq j \leq m$, tekintsük azt a részproblémát, hogy mi a hasonlósági értéke az S1 sorozat első i, az S2 sorozat első j betűjét tartalmazó sorozatoknak. Jelölje ezt az értéket Opt(i,j). Opt(0,j)=2*j, illetve Opt(i,0)=2*i, hiszen a 0 hosszú, azaz üres sorozat csak az üres sorozatból állítható elő. Az alábbi rekurzív összefüggések teljesülnek, ha i>0 és j>0:

Opt(i,j)=Opt(i-1,j-1), ha S1[i]=S2[j]

Opt(i,j)=min{ Opt(i-1,j-1)+1, Opt(i,j-1)+2, Opt(i-1,j)+2 } ha S1[i] $\neq$ S2[j]

S1 és S2 hasonlósági értéke Opt(n,m). Az Opt(i,j) értékeket táblázatkitöltéssel számíthatjuk. Olyan sorrendben kell számítani, hogy amikor az (i,j) részprobléma értékét számítjuk, akkor az Opt(i-1,j-1), Opt(i-1,j) és Opt(i,j-1) értékeket már ki kellett számítani.

A feladatban szereplő második, harmadik és negyedik sor előállításához célszerű segéd információt tárolni minden (i,j) részproblémához az S[i,j] tömbelemben. A segéd információ azt tartalmazza, hogy Opt(i,j) minimuma melyik esetnek felel meg.

```
Program:
  Be: S1,S2; n:=Hossz(S1); m:=Hossz(S2)
  Ciklus j=0-tól m-ig
    Opt(0,j):=2*j; S(0,j):='B'
  Ciklus vége
  Ciklus i=0-től n-ig
    Opt(i,0):=2*i; S(i,0):='A'
  Ciklus vége
  Opt(0,0):=0
  Ciklus i=1-től n-ig
    Ciklus j=1-től m-ig
      Opt(i,j):=2*(n+m)
      Ha S1(i)=S2(j)
         akkor Opt(i,j):=Opt(i-1,j-1); S(i,j):='='
      különben
        Ha Opt(i,j)>Opt(i-1,j-1)+1 akkor
          Opt(i,j):=Opt(i-1,j-1)+1; S(i,j):='X'
        Elágazás vége
        Ha Opt(i-1,j)<Opt(i,j-1) akkor
          Ha Opt(i-1,j)+2<Opt(i,j) akkor
            Opt(i,j):=Opt(i-1,j)+2; S(i,j):='A'
          Elágazás vége
        különben
          Ha Opt(i,j-1)+2<Opt(i,j) akkor
            Opt(i,j):=Opt(i,j-1)+2; S(i,j):='B'
          Elágazás vége
        Elágazás vége
      Elágazás vége
    Ciklus vége
  Ciklus vége
  AA:='' BB:='' C:=''; i:=n; j:=m
  Ciklus amíg i>0 és j>0
    Ha S(i,j)='=' akkor
       AA:=AA+S1(i); BB:=BB+S2(j); C:=C+S1(i); i:=i-1;j:=j-1
    különben ha S(i,j)='X' akkor
       AA:=AA+'X'; BB:=BB+S2(j);  C:=C+S2(j); i:=i-1; j:=j-1
    különben ha S(i,j)='A' akkor AA:=AA+'_'; i:=i-1
    különben ha S(i,j)='B' akkor BB:=BB+'_'; j:=j-1
  Ciklus vége
  Ciklus amíg i>0
     AA:=AA+'_'; i:=i-1
  Ciklus vége
  Ciklus amíg j>0
     BB:=BB+'_'; j:=j-1
  Ciklus vége
  Ki: Opt(n,m),C,AA,BB
Program vége.
```

<u>4. feladat</u>: Vidámpark (15 pont)

Jelölje `H(i,j)=p>0`, ha `j` elérhető `i`-hosszú sétával a kezdőpontból, és ekkor a `p→j` élen!

```
Vidámpark(N,G,Kifok,OK,K,S):
  H(1):=(0,…,0); H(1,1):=1; i:=2; OK:=igaz
  Ciklus amíg i≤K és OK
    OK:=hamis; H(i):=(0,…0)
    Ciklus j=1-től N-ig
      Ha H(i-1,j)>0 akkor Ciklus u=1-től Kifok(j)-ig
                            q:=G(j,u); H(i,q):=j; OK:=igaz
                          Ciklus vége
    Ciklus vége
  Ciklus vége
  Ha OK akkor OK:=H(K,N)>0
              Ha OK akkor i:=K; p:=N
                          Ciklus amíg i>0
                            S(i):=p; p:=H(i,p); i:=i-1
                          Ciklus vége
Eljárás vége.
```

<u>5. feladat</u>: Labirintus (15 pont)

Tekintsük azt a gráfot, amelynek csúcsai a labirintus mezői, azaz (x,y) számpárok. Minden ponthoz rendeljük hozzá, hogy az a mező fal, ajtó, vagy kapcsol-e. Ebben a gráfban kell legrövidebb utat keresni az (1,1) pontból az (M,N) pontba, figyelembe véve a kapcsolókat és az ajtókat. Mivel kapcsolóra lépve minden ajtó ellentétes állapotra vált, ezért csak azt kell tudni, hogy mi volt a kiindulási állapota, és hogy az adott mezőre páros, vagy páratlan számú lépéssel értünk-e. A megoldás megadható szélességi bejárással.

A `Lép` tömb a lehetséges előre, a `VLép` a visszalépéseket tartalmazza, az `IR` pedig a lépések irányát.

```
Lép: Tömb(1..4: Rekord x,y:-1..1)=((x:-1;y:0), (x:0;y:1),
(x:1;y:0), (x:0;y:-1) )
```

```
VLép: Tömb(1..4) Rekord x,y:-1..1) =((x:1;y:0), (x:0;y:-1), (x:-
1;y:0), (x:0;y:1) )
```

```
IR: Tömb(1..4: Karakter)=('F','J','L','B')
```

```
Labirintus(M,N,G,K,L):
  SorÜres; p.x:=1; p.y:=1; p.kb:=igaz
  D(1,1,igaz):=0; Apa(1,1,igaz):=1; Sorba(p)
  Ciklus amíg nem ÜresSor? és
            (Apa(M,N,igaz)=0 vagy Apa(M,N,hamis)=0)
     Sorból(p)
     Ciklus i=1-től 4-ig
       q.x:=p.x+Lép(i).x; q.y:=p.y+Lép(i).y
       Ha G(q.x,q.y)=0 vagy G(q.x,q.y)=2 vagy
          G(q.x,q.y)=3 és p.kb vagy G(q.x,q.y)=4 és nem p.kb
          akkor Ha G(q.x,q.y)=2 akkor q.kb:=nem p.kb
                         különben q.kb:=p.kb
                Ha Apa(q.x,q.y,q.kb)=0
                   akkor D(q.x,q.y,q.kb):=D(p.x,p.y,p.kb)+1
                         Apa(q.x,q.y, q.kb):=i
                         S(Sv):=q; inc(Sv)
     Ciklus vége
  Ciklus vége
```

```
  Ha Apa(M,N,igaz)=0 és Apa(M,N,hamis)=0 akkor Mego:=0
  Ha Apa(M,N,igaz)>0 akkor Mego:=D(M,N,igaz); kb:=igaz
  Ha Apa(M,N,hamis)>0 és D(M,N,hamis)<Mego)
     akkor Mego:= D(M,N,hamis); kb:=hamis
  K:=Mego; x:=M; y:=N
  Ciklus amíg K>0
    lk:=Apa(x,y,kb)
    Ha G(x,y)=2 akkor kb:=nem kb
    x:=x+VLép(lk).x; y:=y+VLép(lk).y; L(k):=IR(lk); K:=K-1
  Ciklus vége
Eljárás vége.
```

2008. Első forduló

Ötödik-nyolcadik osztályosok

Számítógép nélküli feladatok

1. feladat: Képlet (20 pont)

A. 1 3 5 7 9 11 13 15 17 19	5 pont
B. 1 4 9 16 25 36 49 64 81 100	5 pont
C. 1 2 3 5 8 13 21 34 55 89	5 pont
D. 1 3 6 10 15 21 28 36 45 55	5 pont

2. feladat: Park (23 pont)

A. A=20, B=35, C=47, D=36, E=72, F=22, Ki=80	1+2+2+1+2+1+2 pont
B. Be – A – C – E – Ki	5 pont
C. Be – D – E – B – A – C – Ki	7 pont

(ahány szobrot tartalmazó helyes utat ad, annyi pont adható)

3. feladat: Kétirányból (17 pont)

Az alábbi algoritmusban aláhúzással jelöljük a hibák javítását. A versenyzőtől a javítást nem várjuk, elég, ha bejelöli, hogy hol hibás a fenti algoritmus.

Részpontszámok:

- A három karakter-elírás (≤, ≠, -) felismerése egyenként 2-2 pontot ér;
- az értékadás két oldalának felcserélése (C:=E) 3 pont;
- a két kimaradt feltétel észrevétele egyenként 4-4 pont.

```
...
E:=1; U:=N; A:=0; B:=0; C:=0
Ciklus amíg E≤U
  Ciklus amíg E≤U és X(E)≠0
    E:=E+1
  Ciklus vége
  Ha E≤U és A=0 akkor A:=E
  Ha E≤U akkor C:=E
  Ciklus amíg E≤U és X(U)≠0
    U:=U-1
  Ciklus vége
  Ha E≤U és B=0 akkor B:=U
  Ciklus vége
...
```

Számítógépes feladatok

1. feladat: Távolság (40 pont)

Kiírja a beolvasott nevet	5 pont
Nincs benne E-betű (YXCVB ⇒ -1)	5 pont
Egyetlen E-betű van benne (QWERT ⇒ 0)	5 pont
Pontosan két E-betű van benne, az elején és a végén (ERRE ⇒ 3)	5 pont
Pontosan két E-betű van benne, nem az elején és a végén (WEREW⇒ 2)	5 pont

Több E-betű is van benne (WEWEWEW⇒ 4)	5 pont
Csupa E-betűből áll (EEEEEE ⇒ 5)	5 pont
Szomszédos E-betűk (QWEERT ⇒ 1)	5 pont

Ha a távolságok eggyel térnek el – azaz a két index különbségét rosszul számolja – akkor tesztenként 3-3 pont adható.

Kilencedik-tizedik osztályosok

1. feladat: Ütemezés (17 pont)

A. A=2, B=3, C=3, D=5, V=9	1+1+1+3+3 pont
B. B, 1 időegységgel, C, 4 időegységgel	1+3+1+3 pont

2. feladat: Gyorsabbra (21 pont)

Maximális pontszám (21) adható az alábbi (vagy ezzel ekvivalens) megoldásra:

```
Ciklus J=1-től N-ig
  S:=(X(J)+19) DIV 20
  DB(S):=DB(S)+1
Ciklus vége
I:=DB(0)+DB(1)
II:=DB(2)
...
```

Részpontszámok:

Független elágazások helyett minden értelmes helyen különben-ágat használ	3 pont
Ha különben-ágat használ, akkor a feltételek első részét (≤) elhagyja	3 pont
Észreveszi, hogy az `0≤X(J)` felesleges	3 pont
Észreveszi, hogy az `X(J)≤100` felesleges	3 pont
Észreveszi, hogy `I` és `BUK` ugyanaz	3 pont
Észreveszi, hogy `II+III+IV+V=SIK`	3 pont
Észreveszi, hogy `I+II+III+IV+V=N`, azaz az egyiket nem kell számítani	3 pont

3. feladat: Mik a hibák? (21 pont)

Az alábbi algoritmusban aláhúzottan szerepel a 7 javított hiba. A versenyzőnek a hibát nem kell kijavítania, csupán bejelölnie. Mindegyik megtalálása 3 pontot ér. Az alábbival ekvivalens hibafelismerések is elfogadhatók.

```
JÓösszeg(N,X,M,Y,K,Z):
  K:=N; Z:=X
  Ciklus I=1-től M-ig
    J:=1
    Ciklus amíg J≤N és Y(I).név≠X(J).név
      J:=J+1
    Ciklus vége
    Ha J≤N akkor Z(J).db:=X(J).DB+Y(I).db
        különben K:=K+1; Z(K):=Y(I)
    Elágazás vége
  Ciklus vége
Eljárás vége.
```

4. feladat: Pakolás (17 pont)

Amennyivel nagyobb a válasz, annyi pontot kell levonni az alábbiakból úgy, hogy 0-nál kisebb pontszám egyik részfeladatra sem lehet.

A. 2 4 pont

B. 4 4 pont

C. 5 5 pont

D. 5 4 pont

5. feladat: Kifejezések (24 pont)

A. A B * C D * + E F / + 4 pont

B. X Y – X Z + * 4 pont

C. X Y + Z A B – C – * + 4 pont

D. (X+Y)*(A+B) 4 pont

E. (A–B)*C+(C+D)*B 4 pont

F. ((A+B)*C+(A+B))/(A+B) 4 pont

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Ütemezés (20 pont)

A. A=20, B=70, C=47, D=36, E=122, F=22, V=130 1+2+1+1+2+1+3 pont

B. C, 25 időegységgel, D, 6 időegységgel, F, 18 időegységgel 1+2+1+2+1+2 pont

2. feladat: Gyorsabbra (18 pont)

Maximális pontszám (18) adható az alábbi (vagy ezzel ekvivalens) megoldásra:

```
Ciklus J=1-től N-ig
  DB(X(J)):=DB(X(J))+1
Ciklus vége
I:=DB(5)+DB(6)+DB(7)+DB(8)
II:=DB(9)+DB(10)
...
```

Részpontszámok:

Független elágazások helyett mindenhol különben-ágat használ 2 pont

Észreveszi, hogy az `5≤X(J)` felesleges 2 pont

Észreveszi, hogy az `X(J)≤12` felesleges 2 pont

Észreveszi, hogy `OK` és `IO` ugyanaz 3 pont

Észreveszi, hogy `CE=II`, valamint a 11. osztályosok száma 3 pont

Észreveszi, hogy `IO és OK=ER`, valamint a 11. osztályosok száma vagy `CE=II+OK-ER` 3 pont

Észreveszi, hogy `I+II+OK=N`, azaz az egyiket nem kell számítani 3 pont

3. feladat: Mik a hibák? (21 pont)

Az alábbi algoritmusban aláhúzottan szerepel a 7 javított hiba. A versenyzőnek a hibát nem kell kijavítania, csupán bejelölnie. Mindegyik megtalálása 3 pontot ér. Az alábbival ekvivalens hibafelismerések is elfogadhatók

```
JÓunió(N,X,M,Y,K,Z):
  K:=N; Z:=X
  Ciklus I=1-től M-ig
    J:=1
    Ciklus amíg J≤N és Y(I).név≠X(J).név
      J:=J+1
    Ciklus vége
    Ha J≤N akkor Ha Y(I).db>Z(J).db akkor Z(J).db:=Y(I).db
                 Elágazás vége
        különben K:=K+1; Z(K):=Y(I)
    Elágazás vége
  Ciklus vége
Eljárás vége.
```

4. feladat: Pakolás (17 pont)

Amennyivel nagyobb a válasz, annyi pontot kell levonni az alábbiakból úgy, hogy 0-nál kisebb pontszám egyik részfeladatra sem lehet.

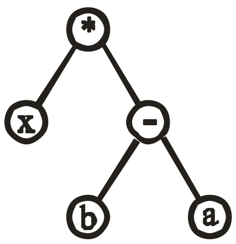
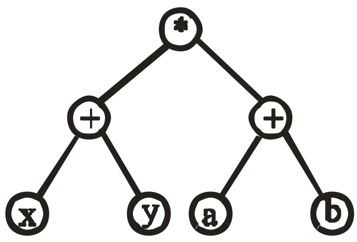
A. 6	5 pont
B. 4	6 pont
C. 4	6 pont

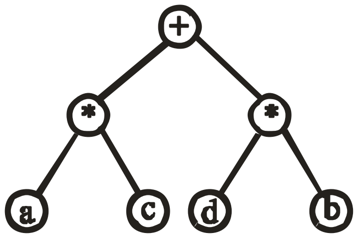
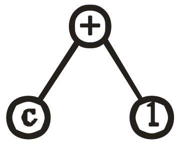
5. feladat: Kifejezésfa (24 pont)

Az A, B, C, D részfeladatokra maximálisan 6-6 pont adható, mindegyik 2+1+3 pont bontásban.

Az alábbi minták szerint az ábráknak megfelelő képlet megadása esetenként 2 pont, az átalakított képlet megadása esetenként 1 pont, az átalakított fa helyes felrajzolása esetenként 3 pont.

Figyelem: más ekvivalens megoldások is lehetnek!

A: –x*(a–b) ⇒ x*(b–a)	B: (x+y)*a+b*(x+y) ⇒ (x+y)*(a+b)
	

C: (a–b)*c+(c+d)*b ⇒ a*c+d*b	D: ((a+b)*c+(a+b))/(a+b) ⇒ c+1
	

2008. Második forduló

Ötödik-nyolcadik osztályosok

<u>1. feladat</u>: Titkos kód (20 pont)

Meg kell keresni az első számot, ami jó maradékot ad, majd meg kell számolni a többit!

```
Titkos(N,A,B,C,L,DB):
  h:=1
  Ciklus i=1-től N-ig
    h:=h*10
  Ciklus vége
  i:=0
  Ciklus amíg i<h és nem
                     (i mod 5=A és i mod 7=B és i mod 11=C)
    i:=i+1
  Ciklus vége
  Ha i<h akkor L:=i; i:=i+1; db:=1
               Ciklus amíg i<h
                 Ha i mod 5=A és i mod 7=B és i mod 11=C
                    akkor db:=db+1
                  i:=i+1
               Ciklus vége
  különben db:=0
Eljárás vége.
```

<u>2. feladat</u>: Szótagolás (35 pont)

Az első magánhangzó biztosan az első szótaghoz tartozik. Ezután meg kell keresni az elválasztás helyét. Az egyszerűsítés miatt ezt könnyű megtalálni.

```
Szótagol(s,u):
  i:=1; u:=''
  Ciklus amíg i≤hossz(s)) és nem magánhangzó(s(i))
    u:=u+s(i); i:=i+1
  Ciklus vége
```

```
  u:=u+s(i); i:=i+1
  Ciklus amíg i≤hossz(s)
    Ha magánhangzó(s(i)) akkor u:=u+'-'+s(i)); i:=i+1
    különben ha i<hossz(s) és magánhangzó(s(i+1))
             akkor u:=u+'-'+s(i)+s(i+1); i:=i+2
    különben ha i≤hossz(s)-3 és s(i)+s(i+1)+s(i+2)='dzs'
                és magánhangzó(s(i+3))
             akkor u:=u+'-dzs'+s(i+3); i:=i+4
    különben ha i≤hossz(s)-2 és kettős(s(i)+s(i+1))
                és magánhangzó(s(i+2))
             akkor u:=u+'-'+s(i)+s(i+1)+s(i+2)); i:=i+3
    különben ha i≤hossz(s)-1 és kettős(s(i)+s(i+1))
             akkor u:=u+s(i)+s(i+1)); i:=i+2
    különben u:=u+s(i); i:=i+1
  Ciklus vége
Eljárás vége.

magánhangzó(c):
  magánhangzó:=c∈{'a','á','e','é','i','í','o','ó',
                  'ö','ő','u','ú','ü','ű'}
Függvény vége.

kettős(s):
  kettős:=(s='cs' vagy s='dz' vagy s='gy' vagy s='ly'
           vagy s='ny' vagy s='sz' vagy s='ty' vagy s='zs')
Függvény vége.
```

<u>3. feladat</u>: Sziget (20 pont)

Első lépésként határozzuk meg a szigetek közötti távolságokat, majd elölről és hátulról indulva is addig menjünk, amíg a távolságok nem nagyobbak T-nél!

```
Sziget(N,érték,T,dbb,dbj)
  j:=1; föld:=igaz; s(j):=0
  Ciklus i=1-től N-ig
    Ha érték(i)=0 akkor s(j):=s(j)+1; föld:=hamis
    különben ha érték(i)=1 akkor
                Ha nem föld akkor j:=j+1; s(j):=0
                                  föld:=igaz
  Ciklus vége
  j:=j-1; dbb:=1
  Ciklus amíg dbb≤j és s(dbb)≤T
    dbb:=dbb+1
  Ciklus vége
  dbj:=j
  Ciklus amíg dbj≥1 és s(dbj)≤T
    dbj:=dbj-1
  Ciklus vége
  dbj:=j-dbj+1
Eljárás vége.
```

Kilencedik-tizedik osztályosok

<u>1. feladat</u>: Tagok (20 pont)

A kapcsolatok alapján építsük fel a titkos társaság bináris fáját! Ha a társaság egyetlen tagból, azaz a főnökből áll, akkor a megoldás (2,0,0). Ha a főnöknek nincs közvetlen A típusú beosztottja, akkor egy közvetlen beosztott jöhet a társaságba, valamint a B típusú beosztottjához annyi, ahány új elemet el lehet helyezni a jobboldali részfába. Hasonlót kell számolni akkor, ha a főnöknek nincs közvetlen B-típusú beosztottja. Ha mindkét típusú közvetlen beosztottja van már, akkor már újabb közvetlen beosztott nem jöhet.

```
Tag(N,x,y,c,dbA,dbB,dbC):
  t:=(0,…0)
  Ciklus i=1-től N-1-ig
    Ha c(i)='A' akkor t(y(i),1):=x(i)
              különben t(y(i),2):=x(i)
  Ha N=1 akkor dbA:=2; dbB:=0; dbC:=0
  különben
    ha t(1,1)=0 akkor dbA:=1 dbB:=0; dbC:=számol(t(1,2))
    különben ha t(1,2)=0
                akkor dbA:=1; dbB:=számol(t(1,1)); dbC:=0
    különben dbA:=0; dbB:=számol(t(1,1))
              dbC:=számol(t(1,2))
Eljárás vége.
```

```
számol(i):
  Ha t(i,1)+t(i,2)=0 akkor számol:=2
  különben ha t(i,1)=0 akkor számol:=1+számol(t(i,2))
  különben ha t(i,2)=0 akkor számol:=1+számol(t(i,1))
  különben számol:=számol(t(i,1))+számol(t(i,2))
Függvény vége.
```

2. feladat: Ismerősök (20 pont)

Számoljuk ki mindenkire az ismerősei halmazát! Az A részfeladat megoldásai azon (i,j) párok, amelyek ismerősei közösek (figyelve arra, hogy i-nek ismerőse j, j-nek pedig i). A B részfeladat megoldásai pedig azon (i,j) párok, amelyek ismerősei halmazának metszete nem üres.

```
Ismer(N,x,y,M,K,mind,L,van):
  ism:=({},…,{})
  Ciklus i=1-től M-ig
    ism(x(i)):=ism(x(i))∪{y(i)}
    ism(y(i)):=ism(y(i))∪{x(i)}
  Ciklus vége
  K:=0; L:=0
  Ciklus i=1-től N-1-ig
    Ciklus j=i+1-től N-ig
      Ha ism(i)-{j}=ism(j)-{i} és ism(i)≠{j} és ism(i)≠{}
         akkor k:=k+1; mind(k,1):=i; mind(k,2):=j
      Ha ism(i)∩ism(j)≠{}
         akkor L:=L+1; van(L,1):=i; van(L,2):=j
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

3. feladat: Pakolás (20 pont)

Amíg a ládaméret növekszik, a ládák pakolhatók jobbra. Ettől kezdve, amíg a ládaméret csökken, pakolhatók balra. Ekkor pl. a lokális maximum megtalálásánál számolhatjuk az eredményt.

```
Pakol(N,x,M):
  nő:=igaz; e:=0; M:=0
  Ciklus i=1-től N-ig
    Ha nő akkor Ha x(i)<e akkor M:=M+1; nő:=hamis
    különben Ha x>e akkor nő:=igaz
    e:=x
  Ciklus vége
  Ha nő akkor M:=M+1
Eljárás vége.
```

4. feladat: Vitorlás (15 pont)

Először mindenkire számoljuk meg, hogy hány versenyen indult, valamint hogy milyen helyezést hányszor ért el! Ezután mindenkinek számoljuk ki a pontszámát úgy, hogy csak a legjobb L helyezést vesszük figyelembe!

```
Vitorlás(N,z,M,K,dbx,x,db,t):
  Ciklus i=1-től M-ig
    t(i).sor:=i; t(i).pont:=0; t(i).db:=0
    Ciklus j=1-től K-ig
      t(i).hely(j):=0
    Ciklus vége
  Ciklus vége
  Ciklus i=1-től N-ig
    Ciklus j=1-től K-ig
      t(z(i,j)).db:=t(z(i,j)).db+1
      t(x).hely(j):=t(x).hely(j)+1
    Ciklus vége
  Ciklus vége
  Ciklus i=1-től M-ig
    s:=0
    Ciklus j=1-től K-ig
      Ha s+t(i).hely(j)>L akkor t(i).hely(j):=L-s
      s:=s+t(i).hely(j)
      t(i).pont:=t(i).pont+(K-j+1)*t(i).hely(j)
    Ciklus vége
  Ciklus vége
```

Az első részfeladat azon versenyzők megadása, akik minden futamban az első K között végeztek, azaz pontosan N eredményüket ismerjük:

```
  dbx:=0
  Ciklus i=1-től M-ig
    Ha t(i).db=N akkor dbx:=dbx+1; x(dbx):=i
  Ciklus vége
```

A második részfeladatban sorba kell rendezni a versenyzőket, figyelve arra, hogy ugyanannyi pont esetén a jobb helyezések nagyobb száma dönt közöttük. Itt csupán annyi a teendőnk, hogy a 0 pontszámúakat – azaz azokat, akik nem kerültek be egyszer sem az első K versenyző közé – ne írjuk ki!

```
  db:=0
  Ciklus i=1-től M-1-ig
    max:=i
    Ciklus j=i+1-től M-ig
      Ha kisebb(max,j) akkor max:=j
    Ciklus vége
    Ha max≠i akkor y:=t(i); t(i):=t(max); t(max):=y
    Ha t(i).pont>0 akkor db:=i
  Ciklus vége
Eljárás vége.
```

```
kisebb(i,j):
  Ha t(i).pont<t(j).pont akkor kisebb:=igaz
  különben ha t(i).pont>t(j).pont akkor kisebb:=hamis
  különben ii:=1
           Ciklus amíg ii≤k és t(i).hely(ii)=t(j).hely(ii)
             ii:=ii+1
           Ciklus vége
           Ha ii≤k akkor kisebb:=t(i).hely(ii)<t(j).hely(ii)
                különben kisebb:=hamis
Függvény vége.
```

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Tagok (15 pont)

A kapcsolatok alapján építsük fel a titkos társaság nem-bináris fáját! Ha a társaság egyetlen tagból, azaz a főnökből áll, akkor a megoldás (K,0). Ha a főnöknek van L közvetlen beosztottja, akkor K-L újabb közvetlen beosztottja lehet, valamint az L közvetlen beosztottjára meg kel számolni. hogy nekik még hány újabb beosztottjuk lehet!

```
Tagok(N,x,y,s,K,t,a,b):
  t:=(0,…,0)
  Ciklus i=1-től N-1-ig
    t(y(i),s(i)):=x(i)
  Ciklus vége
  Ha n=1 akkor a:=K; b:=0
      különben a:=0
               Ciklus i=1-től K-ig
                 Ha t(1,i)=0 akkor a:=a+1
               Ciklus vége
               Ciklus i=1-től K-ig
                 Ha t(1,i)=0 akkor b(i):=0
                          különben b(i):=számol(t(1,i))
               Ciklus vége
Eljárás vége.
```

```
számol(i):
  db:=0
  Ciklus j=1-től K-ig
    Ha t(i,j)=0 akkor db:=db+1
             különben db:=db+számol(t(i,j))
  Ciklus vége
  számol:=db
Függvény vége.
```

2. feladat: Város (15 pont)

A részfeladat: egy város külső pont, ha van olyan szomszédja, ami csak egyszer szomszéd.

B részfeladat: a háromszögek száma, ha belső pont, illetve ennél eggyel több, ha külső (azaz a konvex burkon levő) pont.

```
Város(N,h,dbK,M,db):
  Ciklus i=1-től N-ig
    Ha külső(i) akkor dbk:=dbk+1; db(i):=1
             különben db(i):=0
  Ciklus vége
  Ciklus i=1-től M-ig
    db(h(i,1)):=db(h(i,1))+1; db(h(i,2)):=db(h(i,2))+1
    db(h(i,3)):=db(h(i,3))+1
  Ciklus vége
Eljárás vége.
```

```
külső(i):
  szom:=(0,…,0)
  Ciklus j=1-től M-ig
    Ha i=h(j,1) vagy i=h(j,2) vagy i=h(j,3)
       akkkor Ciklus k=1-től 3-ig
                szom(h(j,k)):=szom(h(j,k))+1
              Ciklus vége
  Ciklus vége
  j:=1
  Ciklus amíg j≤n és szom(j)≠1
    j:=j+1
  Ciklus vége
  külső:=(j≤n)
Függvény vége.
```

3. feladat: Konténer (15 pont)

Minden lehetséges konténerméréthez adjuk meg az éppen akkora konténerek listáját! Ezután a méreteket csökkenő sorrendben véve (azt is, amibe belerakjuk és azt is, amit belerakunk) nézzük, hogy az egyes konténerek melyik másikba tehetők!

```
Konténer(N,A,M):
  maxi:=0; Első:=(0,…,0); Köv:=(0,…,0)
  Ciklus i=N-től 1-ig -1-esével
    x:=A(i); Köv(i):=Első(x); Első(x):=i
    Ha x>maxi akkor maxi:=x
  Ciklus vége
  M:=0
  Ciklus amíg maxi>0
    i:=Első(maxi)
    Ciklus y=maxi-1-től 1-ig -1-esével
      Ha Első(y)>0 és Első(y)<i
         akkor M:=M+1; ii:=Első(y); Első(y):=Köv(ii)
    Ciklus vége
    Első(maxi):=Köv(Első(maxi))
    Ciklus amíg maxi>0 és Első(maxi)=0
      maxi:=maxi-1
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

4. feladat: Pakolás (15 pont)

Ez egy dinamikus programozási feladat, legyen `P(x,i)` igaz, ha az első `i` tárgyból `x` mennyiségű felrakható a kamionra!

Az első részfeladat megoldása a legnagyobb `S` érték, amire `P(S,N)` igaz. A második részfeladat megoldása pedig `P(S,N)`-ből indulva a jó pakolás visszakövetése.

```
Pakol(N,K,s,S,db,B):
  P:=(hamis,…,hamis)
  Ciklus i=0-tól N-ig
    P(0,i):=igaz
  Ciklus vége
  Ciklus i=1-től N-ig
    Ciklus x=1-től K-ig
      P(x,i):=P(x,i-1) vagy x≥s(i) és P(x-s(i),i-1)
  Ciklus vége
  S:=K; db:=0
  Ciklus amíg nem P(S,N)
    S:=S-1
  Ciklus vége
  x:=S; i:=N
  Ciklus amíg x>0
    Ciklus amíg i>0 és P(x,i)
      i:=i-1
    Ciklus vége
    db:=db+1; B(db):=i+1; x:=x-s(i+1)
  Ciklus vége
Eljárás vége.
```

5. feladat: Kimérés (15 pont)

A feladatra egy gráfmodell építhető, ahol a gráf pontjai a lehetséges tejeskanna kombinációk, a gráf élei pedig az ezekből kiinduló szabályos öntések.

A megoldás ezen gráfra egy szélességi bejárás alkalmazása. A lépések előállításához pedig a végső állapotból vissza kell követni az optimális öntögetéseket.

```
Kimér(T1,T2,T3,A,B,M,mit,mibe,önt):
  L(1).dx:=-A; L(1).dy:=A; L(2).dx:=-A; L(2).dy:=0
  L(3).dx:=A; L(3).dy:=-A; L(4).dx:=0; L(4).dy:=-A
  L(5).dx:=A; L(5).dy:=0; L(6).dx:=0; L(6).dy:=A
  L(7).dx:=-B; L(7).dy:=B; L(8).dx:=-B; L(8).dy:=0
  L(9).dx:=B; L(9).dy:=-B; L(10).dx:=0; L(10).dy:=-B
  L(11).dx:=B; L(11).dy:=0; L(12).dx:=0; L(12).dy:=B
  L1(1).dx:=A; L1(1).dy:=-A; L1(2).dx:=A; L1(2).dy:=0
  L1(3).dx:=-A; L1(3).dy:=A; L1(4).dx:=0; L1(4).dy:=A
  L1(5).dx:=-A; L1(5).dy:=0; L1(6).dx:=0; L1(6).dy:=-A
  L1(7).dx:=B; L1(7).dy:=-B; L1(8).dx:=B; L1(8).dy:=0
  L1(9).dx:=-B; L1(9).dy:=B; L1(10).dx:=0 ; L1(10).dy:=B
  L1(11).dx:=-B; L1(11).dy:=0; L1(12).dx:=0; L1(12).dy:=-B
```

```
  K12:=(0,…,0); K12(T1,T2):=1; T:=T1+T2+T3
  SorÜres; OK:=hamis; Sorba(T1,T2)
  Ciklus amíg nem ÜresSor? és nem OK
    Sorból(x,y)
    Ha x=y és 3*x=T akkor OK:=igaz
    különben Ciklus i=1-től 12-ig
               xx:=x+L(i).dx; yy:=y+L(i).dy
               Ha xx≥0 és yy≥0 és xx+yy≤T és K12(xx,yy)=0
                  akkor K12(xx,yy):=i; Sorba(xx,yy)
             Ciklus vége
  Ciklus vége
  Ha nem OK akkor M:=-1
  különben M:=0
           Ciklus amíg x≠T1 vagy y≠T2
             M:=M+1; i:=K12(x,y); Lep(m):=i
             x:=x+L1(i).dx; y:=y+L1(i).dy
           Ciklus vége
           Ciklus i=M-től 1-ig -1-esével
             Ha Lep(i)≤6 akkor AB:=A különben AB:=B
             Ha L(Lep(i)).dx<0
                akkor x:=1
                      Ha L(Lep(i)).dy>0 akkor y:=2
                                     különben y:=3
             különben ha L(Lep(i)).dx=0
                         akkor Ha L(Lep(i)).dy<0
                                  akkor x:=2; y:=3
                               különben y:=3; y:=2
             különben Ha L(Lep(i)).dy<0
                         akkor x:=2; y:=1
                      különben x:=3; y:=1
             mit(i):=x; mibe(i):=y; önt(i):=AB
Eljárás vége.
```

2008. Harmadik forduló

Ötödik-nyolcadik osztályosok

1. feladat: Szél (24 pont)

Az első részfeladat megoldása a 0 sebességek száma, a második részfeladaté a 10-nél kisebbek, a harmadik részfeladaté a 100-nál nagyobbak száma.

Ha közülük bármelyiket találjuk, akkor ott a verseny szempontjából rossz időszak következik. Ha a rossz időszak kezdetétől K napon át jó idő volt, akkor megtaláltuk a negyedik részfeladat megoldását.

```
Szél(N,seb,K,szdb,kdb,vdb,kezdet):
  szdb:=0; kdb:=0; vdb:=0; rossz:=0; kezdet:=0
  Ciklus i=1-től N-ig
    Ha seb(i)=0 akkor rossz:=i; szdb:=szdb+1
    különben ha seb(i)<10 akkor rossz:=i; kdb:=kdb+1
    különben ha seb(i)>100 akkor rossz:=i; vdb:=vdb+1
    különben ha i-rossz=k akkor ha kezdet=0
                                 akkor kezdet:=rossz+1
  Ciklus vége
Eljárás vége.
```

2. feladat: Vasút (25 pont)

Az első állomáson kel felszállni! Ezután, az utolsó állomástól, ahol megállt meg kell keresni a legelső, K kilométernél távolabb levő állomást – itt kell leszállnia legközelebb!

```
Vasút(N,t,K,max,hol):
  max:=1; hol(max):=1
  Ciklus i=2-től N-ig
    Ha t(i)-t(hol(max))≥K akkor max:=max+1; hol(max):=i
  Ciklus vége
  Ha hol(max)≠N akkor hol(max):=N
Eljárás vége.
```

3. feladat: Olimpia (26 pont)

Számoljuk ki az egyes országok éremszámait, majd az éremszám szerint rendezzük csökkenő sorrendbe őket!

```
olimpia(N,s,M,érem):
  érem(i,j):=(0,…,0)
  Ciklus i=1-től M-ig
    érem(i,0):=i
  Ciklus vége
  Ciklus i=1-től N-ig
    érem(s(i,1),1):=érem(s(i,1),1)+1
    érem(s(i,2),2):=érem(s(i,2),2)+1
    érem(s(i,3),3):=érem(s(i,3),3)+1
    Ha s(i,4)>0 akkor érem(s(i,4),3):=érem(s(i,4),3)+1
  Ciklus vége
  Ciklus i=1-től M-1-ig
    max:=i
    Ciklus j=i+1-től M-ig
      Ha nagyobb(j,max) akkor max:=j
    Ciklus vége
    Csere(érem(max),érem(i))
  Ciklus vége
Eljárás vége.

nagyobb(i,j):
  nagy:=hamis
  Ha érem(i,1)>érem(j,1) akkor nagy:=igaz
  különben ha érem(i,1)=érem(j,1) akkor
              Ha érem(i,2)>érem(j,2) akkor nagy:=igaz
              különben ha érem(i,2)=érem(j,2) akkor
                          Ha érem(i,3)>érem(j,3)
                             akkor nagy:=igaz
  nagyobb:=nagy
Függvény vége.
```

Kilencedik-tizedik osztályosok

1. feladat: Címlet (15 pont)

Minden olyan összegre, amely kifizethető az adott címletekkel igaz, hogy őt megnövelve valamelyik címlettel, az szintén kifizethető lesz.

```
Címlet(N,M,címlet,db,fizethet):
  fizethet:=(hamis,…,hamis); fizethet(0):=igaz; db:=0
  Ciklus i=0-tól M-ig
    Ha fizethet(i) akkor Ciklus j=1-től N-ig
                           fizethet(i+címlet(j)):=igaz
                         Ciklus vége
    különben db:=db+1
  Ciklus vége
Eljárás vége.
```

2. feladat: Vitorlás (20 pont)

Először keressük meg a lehetséges versenynapokat! Ha van K lehetséges versenynap, akkor van megoldása a feladatnak.

Ekkor a K. és az első versenynap különbségéből kiszámolható, hogy hány szünnapot kellett tartani. Ezután nézzük a további lehetséges versenynapokat, s ha az azokhoz tartozó szünnapok száma kisebb lenne, akkor azt őrizzük meg!

```
Vitorlás(N,K,min,max,seb,mindb,minek):
  j:=0; i:=1
  Ciklus amíg j<K és i≤N
    Ha seb(i)≥min és seb(i)≤max akkor j:=j+1; sor(j):=i
    i:=i+1
  Ciklus vége
  Ha j=K akkor
     mindb:=1; minert:=sor(K)-sor(1)+1-K
     minek(mindb):=sor(1); j:=sor(K)+1
     Ciklus i=j-től N-ig
       Ha seb(i)≥min és seb(i)≤max) akkor
          Ciklus j=1-től K-1-ig
            sor(j):=sor(j+1)
          Ciklus vége
          sor(K):=i; uj:=sor(k)-sor(1)+1-K
          ha uj<minert akkor mindb:=1; minert:=uj
                             minek(mindb):=sor(1)
          különben ha uj=minert
                      akkor mindb:=mindb+1
                            minek(mindb):=sor(1)
     Ciklus vége
  különben mindb:=0
Eljárás vége.
```

3. feladat: Délkert (20 pont)

Először rendezzük a bemenetet úgy, hogy az i-edik megelőzi a j-ediket, ha `max(a[i],b[i])>max(a[j],b[j])`, vagy `max(a[i],b[i])=max(a[j],b[j])` és `min(a[i],b[i])>min(a[j],b[j])`!

Az indexek rendezés szerinti sorrendjét az S tömb tartalmazza, azaz a sorrend S[1], …, S[N]. Ha N1=0 vagy N2=0, akkor a megoldás nyilvánvaló.

Tegyük fel, hogy N1>0 és N2>0, továbbá a[S[1]]<b[S[1]]. Ekkor van olyan optimális megoldás, amely esetén az S[1] termelő az A hűtőházba szállít.

Tekintsünk egy optimális megoldást, amelyben m1 termelő szállít az A, m2 termelő a B hűtőházba, és a határidő K. K nyilván ≥ a[S[1]]. Ha m1=0 lenne, akkor S[1]-et az A-ba szállítva is optimális megoldás lenne ezzel a K határidővel. Ha m1>0, de az optimális megoldásban S[1] nem A-ba szállít. Tehát a[S[1]]≤b[S[1]]≤K, és minden i-re b[i]≤b[S[1]], így S[1]-től A-ba, i-től B-be szállítva is optimális megoldást kapunk, bármely i-re, akitől A-ba szállított az optimális megoldás.

Tehát a mohó választás: a rendezés szerint soron következő A-ba szállít, ha még tud fogadni az A hűtőház és az A-ba szállítás ideje kisebb, mint B-be szállításé, egyébként a B-be szállít.

```
Délkert(N,N1,N2,a,b,K,i1,M1,i2,M2):
  Beolvas(N,N1,N2,a,b)
  Ciklus i=1-től N-ig
    S(i):=i
  Ciklus vége
  Rendez(1,n)
  i1:=0; i2:=0; K:=0
  Ciklus i=1-től N-ig
    ii:=S(i)
    Ha i1<n1 és a(ii)<b(ii) vagy i2=n2 akkor
       i1:=i1+1; M1(i1):=ii; Ha a(ii)>K akkor K:=a(ii)
    különben i2:=i2+1; M2(i2):=ii
             Ha b(ii)>K akkor K:=b(ii)
  Ciklus vége
Eljárás vége.
```

4. feladat: Falak (20 pont)

A térképen -1 jelöli a falakat. A falból biztos nem lehet kijutni a térkép szélére. ha nem falban vagyunk, akkor a kezdőpontból indítsunk egy bejárást, s ha a bejárás során elértünk a térlép szélére, akkor IGEN a válasz, ha pedig nem értünk el, akkor NEM a válasz.

```
Falak(H,L,K,t,kp,lehet):
  Ciklus i=1-től K-ig
    Ha t(kp(i).y,kp(i).x)>-1 akkor
       ki:=hamis; bejár(i,kp(i).y,kp(i).x,ki)
       Ha ki akkor lehet(i):='IGEN'
          különben lehet(i):='NEM'
    különben lehet(i):='NEM'
  Ciklus vége
Eljárás vége.
```

```
bejár(s,i,j,ki):
  Ha nem ki akkor
     t(i,j):=s
     Ha i∈{1,h} vagy j∈{1,h} akkor ki:=igaz
     különben
        Ha t(i-1,j)>-1 és t(i-1,j)<s akkor bejár(s,i-1,j,ki)
        Ha t(i,j-1)>-1 és t(i,j-1)<s akkor bejár(s,i,j-1,ki)
        Ha t(i+1,j)>-1 és t(i+1,j)<s akkor bejár(s,i+1,j,ki)
        Ha t(i,j+1)>-1 és t(i,j+1)<s akkor bejár(s,i,j+1,ki)
Eljárás vége.
```

Minden tesztben legalább 3 kezdőpont van, amelyekre nincs csupa egyforma jó válasz: (1000, 1000), illetve egy falban levő pont.

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Robot (15 pont)

Ez egy dinamikus programozási feladat, ahol minden tárgyra kiszámoljuk, hogy egyedül jobb-e vinni, vagy az előzővel együtt, vagy az előző kettővel együtt.

```
Robot(N,X,Y,K,Utem):
  K(-2):=Inf; K(-1):=Inf; K(0):=0;
  K(1):=Út(1,1); Egybe(1):=1
  Ciklus i=2-től N-ig
    K(i):=K(i-1)+Út(i,1); Egybe(i):=1
    Ciklus d=2-től 3-ig
      Ha i-d+1≥1 akkor
         rész:=Út(i-d+1,d)
         Ha rész+K(i-d)<K(i) akkor K(i):=rész+K(i-d)
                                   Egybe(i):=d
    Ciklus vége
  Ciklus vége
  i:=N; u:=0;
  Ciklus amíg i>0
    u:=u+1; Utem(u):=Egybe(i); i:=i-Egybe(i)
  Ciklus vége
Eljárás vége.
```

Az út kiszámolása: el kell vinni a tárgyat a nagyobb X-koordinátáig, illetve a nagyobb Y-koordinátáig, majd vissza kell térni a kiinduló pozícióra!

```
Út(i,k):
  mX:=X(i); mY:=Y(i)
  Ciklus ii=i+1-től i+k-1-ig
    Ha mX<X(ii) akkor mX:=X(ii)
    Ha mY<Y(ii) akkor mY:=Y(ii)
  Ciklus vége
  Út:=(mX+mY)*2
Függvény vége.
```

2. feladat: Szolga (15 pont)

Ez egy szélességi bejárás, ahol az összes kiszolgálóból párhuzamosan indulunk. Így minden ponthoz a legrövidebb utat a legközelebbi kiszolgálóból kapjuk meg. Az út utolsó lépése pedig éppen azt adja meg, hogy az adott gépnek merre kel elküldenie az igényét.

```
Szolga(N,Él,K,Kis,Ki,Fok,maxD,D,Apa):
  eleje:=1; vége:=0; maxD:=0
  D:=(0,…,0); Apa:=(0,…,0); SorÜres
  Ciklus i=1-től K-ig
    Apa(Kis(i)):=Kis(i); D(Kis(i)):=0; Sorba(Kis(i))
  Ciklus vége

  Ciklus amíg nem ÜresSor?
    p:=Sorból
    Ciklus i=Ki(p)-től Ki(p)+Fok(p)-1-ig
      q:=Él(i)
      Ha Apa(q)=0 akkor Apa(q):=p; D(q):=D(p)+1; Sorba(q)
                        Ha D(q)>maxD akkor maxD:=D(q)
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

3. feladat: Intervallumok (15 pont)

Minden lehetséges X-re (1≤X≤3600) jelölje Kezd(X) azon intervallumok számát, amelyek kezdőpontja ≥X. Hasonlóan, Vég(X) azon intervallumok száma, amelyek végpontja ≤X. Tehát azon intervallumok száma, amelyeknek van közös pontja adott [A,B] intervallummal: N-Vég(A-1)-Kezd(B+1). Tehát ha A-t rögzítjük, akkor meg tudjuk határozni azt a legkisebb B-t, amelyre N-Vég(A-1)-Kezd(B+1)≥K, ha ez lehetséges. Továbbá, ha [A,B]-re teljesül a feltétel, és növeljük 1-el A-t, akkor meg tudjuk mondani, hogy mennyivel kell (ha kell) növelni B-t, hogy olyan legkisebb [A+1,B] intervallumot kapjunk, amelynek legalább K bementi intervallummal van közös pontja.

```
Inter(N,x,y,K,Mina,Minb):
  Kezd:=(0,…,0); Vég:=(0,…,0)
  Ciklus i=1-től N-ig
    Kezd(x(i)):=Kezd(x(i))+1; Vég(y(i)):=Vég(y(i))+1
  Ciklus vége
  Vég(bal-1):=0; Kezd(jobb+1):=0
  Ciklus x=bal+1-től jobb-ig
    Vég(x):=Vég(x)+Vég(x-1)
  Ciklus vége
  Ciklus x=jobb-1-től bal-ig -1-esével
    Kezd(x):=Kezd(x)+Kezd(x+1)
  Ciklus vége
  a:=bal
  Ciklus amíg Vég(a)=0
    a:=a+1
  Ciklus vége
  b:=a+1
  Ciklus amíg N-(Vég(a-1)+Kezd(b+1))<K
    b:=b+1
  Ciklus vége
  d:=b-a; Mina:=a; Minb:=b
  Ciklus amíg a<jobb és b≤jobb
    a:=a+1
    Ha a=b akkor b:=a+1
    Ciklus amíg b≤jobb és N-(Vég(a-1)+Kezd(b+1))<K
      b:=b+1
    Ciklus vége
    Ha b≤jobb akkor Ha b-a<d akkor d:=b-a; Mina:=a; Minb:=b
   Ciklus vége
Eljárás vége.
```

4. feladat: Lapok (15 pont)

Mivel 1 000 000 hely van, a megoldás egy egyszerű szélességi bejárás lesz.

```
Lapok(L,H,x,y,dx,dy,szín,db):
  Ciklus k=1-től L-ig
    Ciklus j=x(k)-tól x(k)+dx(k)-1-ig
      Ciklus i=y(k)-tól y(k)+dy(k)-1-ig
        t(i,j):=szín(k)
      Ciklus vége
    Ciklus vége
  Ciklus vége
  db:=0
  Ciklus i=1-től H-ig
    Ciklus j=1-től H-ig
      Ha t(i,j)>-1 akkor letöröl(i,j,t(i,j)); db:=db+1
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

```
letöröl(i,j,szín):
  t(i,j):=-1
  Ha i>1 akkor Ha t(i-1,j)=szín akkor letöröl(i-1,j,szín)
  Ha j>1 akkor Ha t(i,j-1)=szín akkor letöröl(i,j-1,szín)
  Ha i<h akkor Ha t(i+1,j)=szín akkor letöröl(i+1,j,szín)
  Ha j<h akkor Ha t(i,j+1)=szín akkor letöröl(i,j+1,szín)
Eljárás vége.
```

5. feladat: Hírek (15 pont)

Tekintsük azt a gráfot, amelynek pontjai a tanulók azonosítói, és p → q akkor és csak akkor irányított él a gráfban, ha a p. tanuló továbbadja a hírt a q. tanulónak. Ha K=1, akkor kiszámítandó, hogy melyik az a pontja a gráfnak, amelyből a legtöbb pont elérhető.

Ekkor minden pontra hajtsunk végre egy bejárást (ez lehet akár mélységi, akár szélességi), és számoljuk az elérhető pontokat. Ha K=2, akkor minden (p,q), p<q pontpárra hajtsunk végre egy olyan módosított szélességi bejárást, a sorba először berakjuk a p és a q pontot, továbbá a (p-1)*N+q-1 színnel színezzük azokat a pontokat, amelyek elérhetők p-ből, vagy q-ból.

```
Hírek(N,K,Ki,Fok,Él,MaxE,MaxP,MaxQ):
  Szín:=(Fehér,…,Fehér); első:=igaz; fn:=N
  Ciklus p=1-től N-ig
    Ha Szín(p)=Fehér akkor MélyBejár(p)
  Ciklus vége
  első:=hamis; Dn:=0; Szín:=(Fehér,…,Fehér)
  Ciklus i=1-től N-ig
    p:=F(i)
    Ha Szin(p)=Fehér akkor MélyBejár(p); Dn:=Dn+1; F(Dn):=p
  Ciklus vége
  Ha K=1 akkor Elér(MaxE,MaxP)
  különben Ha Dn=1 akkor Elér(MaxE,MaxP); MaxQ:=MaxP
  különben Elér2(MaxE,MaxP,MaxQ)
Eljárás vége.

MélyBejár(p):
  Szín(p):=Szürke
  Ciklus i=Ki(p)-től Ki(p)+Fok(p)-1-ig
    q:=Él(i)
    Ha Szín(q)=Fehér akkor MélyBejár(q)
  Ciklus vége
  Ha első akkor F(fn):=p; fn:=fn-1
Eljárás vége.

Elér(MaxE,MaxP):
  D:=(0,…,0); MaxE:=0
  Ciklus i=1-től Dn-ig
    p:=F(i); E:=1; pqszin:=i; D(p):=pqszin
    SorÜres; Sorba(p)
    Ciklus amíg nem ÜresSor?
      r:=Sorból
      Ciklus ii=Ki(r)-től Ki(r)+Fok(r)-1-ig
        q:=Él(ii)
        Ha D(q)≠pqszin akkor D(q):=pqszin; E:=E+1; Sorba(q)
      Ciklus vége
    Ciklus vége
    Ha E>MaxE akkor MaxE:=E; Maxp:=p
  Ciklus vége
Eljárás vége.
```

```
Elér2(MaxE,MaxP,MaxQ):
  D:=(0,…,0); MaxE:=0
  Ciklus i=1-től Dn-ig
    p:=F(i)
    Ciklus j=i+1-től Dn-ig
      q:=F(j); E:=2; pqszin:=(i-1)*n+j-1
      D(p):=pqszin; D(q):=pqszin
      SorÜres; Sorba(p); Sorba(q)
      Ciklus amíg nem ÜresSor?
        r:=Sorból
        Ciklus ii=Ki(r)-től Ki(r)+Fok(r)-1-ig
          q:=Él(ii)
          Ha D(q)≠pqszin akkor D(q):=pqszin; E:=E+1; Sorba(q)
        Ciklus vége
      Ciklus vége
      Ha E>MaxE akkor MaxE:=E; Maxp:=p; Maxq:=q
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

2009. Első forduló

Ötödik-nyolcadik osztályosok

Számítógép nélküli feladatok

1. feladat: Vércsoport (18 pont)

A táblázatban levő értékek felülről lefelé: 2, 3, 2, 3, 1, 1, 2, 1, 0 — 9*2 pont

2. feladat: Hegy (30 pont)

A. DB=17, a 800-nál nagyobb értékek száma	2+4 pont
B. DB=3, a 800 fölé jutások száma	2+4 pont
C. DB=2, a csúcsok (szomszédjaiknál nagyobbak) száma	2+4 pont
D. DB=2, a völgyek (szomszédjaiknál kisebbek) száma	2+4 pont
D. M=110, a legnagyobb emelkedés	2+4 pont

3. feladat: Római szám (20 pont)

N=1 ⇒ 10. ág	2 pont
N=3 ⇒ 10., 11., 12. ág	3 pont
N=7 ⇒ 9., 10., 11. ág	3 pont
N=24 ⇒ 4., 5., 8., 9. ág	3 pont
N=45 ⇒ 3., 9. ág	3 pont
N=62 ⇒ 2., 4., 10., 11. ág	3 pont
N=89 ⇒ 2., 4., 5., 6., 7. ág	3 pont

Számítógépes feladat – VÁLASZTHATÓ

4. feladat: Hőmérséklet (32 pont)

(A) hányszor fagyott – megszámolás: a negatív számok darabszáma

(B) hány fagyos időszak volt – megszámolás: a negatív számokkal kezdődő sorozatok száma

(C) mennyi volt a leghidegebb – minimumkiválasztás

```
Hőmérsékletek(N,H,A,B,C):
  Ha H(1)<0 akkor A:=1 különben A:=0
  B:=0; C:=H(1)
  Ciklus i=2-től N-ig
    Ha H(i)<0 akkor A:=A+1
    Ha H(i)<0 és H(i-1)≥0 akkor B:=B+1
    Ha H(i)<C akkor C:=H(i)
  Ciklus vége
Eljárás vége.
```

Számítógép nélküli feladat – VÁLASZTHATÓ

4. feladat: Szigetek (32 pont)

A ciklus felső határa nem jó, helyesen U-1 vagy U	3+5 pont
Az első feltétel nem jó, **vagy** helyett **és** kell	3+5 pont
A második feltételben az indexelés nem jó, helyesen X(i)>0 és X(i+1)=0	3+5 pont
A második elágazás ágban a DB:=DB+1 hibás, nem kell	3+5 pont

Megjegyzés: a hibák másféle javítása is elfogadható!

Az alábbi algoritmust rontottuk el (aláhúzott a beszúrt, áthúzott a kitörölt):

```
E:=az első 0 helye; U:=az utolsó 0 helye; DB:=0
Ciklus i=E+1-től N~~U~~-1-ig
  Ha X(i)>0 ~~és~~vagy X(i-1)=0 akkor DB:=DB+1; K(DB):=i
  Ha X(i-1)>0 és X(i~~+1~~)=0 akkor DB:=DB+1; V(DB):=i
Ciklus vége
```

Kilencedik-tizedik osztályosok

1. feladat: Kakastaréj (16 pont)

A táblázatban levő értékek felülről lefelé: Dió, Dió, Dió, Rózsa, Dió, Dió, Dió, Rózsa, Dió, Dió, Dió, Rózsa, Borsó, Borsó, Borsó, Egyszerű 16*1 pont

2. feladat: Szigetek (34 pont)

A. E=4, az első 0 (azaz tenger) indexe	1+2 pont
B. U=41, az utolsó 0 (azaz tenger) indexe	1+2 pont
C. DB=3, a szigetek száma	2+2 pont
K()= 6,14,31, a szigetek kezdete	2+2 pont
V()=8,16,36, a szigetek vége	2+2 pont
D. M=25, a legmagasabb sziget magassága	2+2 pont
S=2, a legmagasabb sziget sorszáma, több egyforma esetén az elsőé	1+2+1 pont
E. M=6, a legszélesebb sziget szélessége	2+2 pont
S=3, a legszélesebb sziget sorszáma, több egyforma esetén az elsőé	1+2+1 pont

Megjegyzés: a szövegessel egyenértékű más megfogalmazások is elfogadhatók!

3. feladat: Mik a hibák? (16 pont)

Hibánként 2-2 pont, összesen 16 pont

Az elrontott helyeket piros színnel jelöltük a jó programban. A csillaggal jelölt sorban a relációs jel és a változónév hibája külön hibának számít.

```
kulcsszavak:
  DB:=0
  Ciklus i=1-től KulcsDB-ig
    j:=1
    Ciklus amíg j≤KönyvDB és kulcs(i).kód≠könyv(j).kód
      j:=j+1
    Ciklus vége
```

```
    Ha j≤KönyvDB akkor {az i-edik kulcsszó illik a j-edik könyvre}
       k:=j+1
{*}    Ciklus amíg k≤KönyvDB és (kulcs(i).kód≠könyv(k).kód
                   vagy könyv(k).szerző=könyv(j).szerző)
         k:=k+1
       Ciklus vége
       Ha k>KönyvDB akkor {más szerző könyvére nem illik}
          DB:=DB+1; SZ(DB):=kulcs(i).szó
       Elágazás vége
    Elágazás vége
  Ciklus vége
Eljárás vége.
```

4. feladat: Benzinkút (16 pont)

A. Tankolások száma: 6, lehetséges helye: 1,2,4,6,9,12 — 3+5 pont
(Ha Piripócsot – 1. pont – nem számolja bele, akkor 1+3 pont adható.)

B. Tankolások száma: 5, lehetséges helye: 1,2,4,6,10 — 3+5 pont
(Ha Piripócsot – 1. pont – nem számolja bele, akkor 1+3 pont adható.)

Megjegyzés: a tankolások helyére más helyes megoldás is elfogadható.

5. feladat: Rekurzió (18 pont)

`egyik((1 2 3 4 5),3): (1 2 3)` — 2 pont
`egyik`: megadja az x sorozat első y darab elemét — 4 pont

`másik((1 2 3 4 5),3): (3 4 5)` — 2 pont
`másik`: megadja az x sorozat utolsó y darab elemét — 4 pont

`három((1 2 3 4 5),3,2): (3 4)` — 2 pont
`három`: megadja az x sorozat y. elemétől kezdődő z darab elemét — 4 pont

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Kutyaszín (16 pont)

A táblázatban levő értékek felülről lefelé: világos, világos, világos, vörös, sötét, sötét, sötét, vörös, sötét, sötét, sötét, vörös, fekete, fekete, fekete, vörös — 16*1 pont

2. feladat: Szigetek (27 pont)

A. E=4, az első 0 (azaz tenger) indexe — 1+2 pont

B. U=41, az utolsó 0 (azaz tenger) indexe — 1+2 pont

C. DB=3, a szigetek száma — 1+2 pont
K()= 6,14,31, a szigetek kezdete — 1+2 pont
V()=8,16,36, a szigetek vége — 1+2 pont

D. M=25, a legmagasabb sziget magassága — 1+2 pont
S=2, a legmagasabb sziget sorszáma, több egyforma esetén az elsőé — 1+1+1 pont

E. M=3, a legkeskenyebb sziget szélessége — 1+2 pont
S=1, a legkeskenyebb sziget sorszáma, több egyforma esetén az elsőé — 1+1+1 pont

Megjegyzés: a szövegessel egyenértékű más megfogalmazások is elfogadhatók!

3. feladat: Mik a hibák? (16 pont)

Hibánként 2-2 pont, összesen — 16 pont

Az elrontott helyeket piros színnel jelöltük a jó programban. A csillaggal jelölt sorban a relációs jel és a változónév hibája külön hibának számít.

```
kulcsszavak:
  DB:=0
  Ciklus i=1-től KulcsDB-ig
    j:=1
    Ciklus amíg j≤KönyvDB és kulcs(i).kód≠könyv(j).kód
      j:=j+1
    Ciklus vége
    Ha j≤KönyvDB akkor {az i-edik kulcsszó illik a j-edik könyvre}
       k:=j+1
       Ciklus amíg k≤KönyvDB és (kulcs(i).kód≠könyv(k).kód
                   vagy könyv(k).cím=könyv(j).cím)
         k:=k+1          EZ EGYETLEN HIBÁNAK SZÁMÍT!
       Ciklus vége
{*}    Ha k>KönyvDB akkor {más szerző könyvére nem illik}
          DB:=DB+1; SZ(DB):=kulcs(i).szó
       Elágazás vége
    Elágazás vége
  Ciklus vége
Eljárás vége.
```

<u>4. feladat</u>: Pakolás (23 pont)

A1. égetési idő: 65, egy lehetséges sorrend: 1-2,3-4,5 — 2+3 pont
(jó még: 1-2,3,4-5; 1,2-3,4-5)

A2. égetési idő: 65, a lehetséges sorrend: 1-2,3,4-5 — 2+3 pont

A3. égetési idő: 65, egy lehetséges sorrend: 1-2,3-4,5-6-7 — 2+3 pont
(jó még: 1,2-3-4,5-6-7; 1-2-3,4,5-6-7)

$$Opt(i) = \min\left\{Opt(j-1) + \max_{t=j,\dots i} Idő(t) \;\; ahol \;\; i - j + 1 \le K\right\}$$ — 7 pont

(5 pont adható, ha a képlet formailag jó, csak az indexhatárok hibásak)

Opt(0)=0 vagy Opt(1)=Idő(1) — 1 pont

(Más, a fentivel ekvivalens képlet is elfogadható.)

<u>5. feladat</u>: Rekurzió (18 pont)

`egyik((1 2 3 4 5),3): (1 2 3)` — 2 pont

`egyik`: megadja az `x` sorozat első `y` darab elemét — 4 pont

`másik((1 2 3 4 5),3): (3 4 5)` — 2 pont

`másik`: megadja az `x` sorozat utolsó `y` darab elemét — 4 pont

`három((1 2 3 4 5),3,2): (3 4)` — 2 pont

`három`: megadja az `x` sorozat `y.` elemétől `z.` eleméig levő részét — 4 pont

2009. Második forduló

Ötödik-nyolcadik osztályosok

1. feladat: Törtek (32 pont)

Törtek összeadásához közös nevezőre hozásra van szükség. Ezután a törtekkel bármelyik alapművelet elvégezhető, majd egyszerűsítést kell végrehajtani (ha lehetséges). A legvégén még a 0 kiírására kell figyelni!

```
Törtek(a,b,c,d,e,f):
  Ha a.n≠b.n akkor i:=a.s*b.n; j:=b.s*a.n; k:=a.n*b.n
          különben i:=a.s; j:=b.s; k:=a.n
  c.n:=k; c.s:=i+j; Egyszerűsít(c)
  d.n:=k; d.s:=i-j; Egyszerűsít(d)
  e.n:=a.n*b.n; e.s:=a.s*b.s; Egyszerűsít(e)
  f.n:=a.n*b.s; f.s:=a.s*b.n; Egyszerűsít(f)
Eljárás vége.
```

```
Egyszerűsít(x):
  Ha x.s=0 akkor x.n:=1
  különben k:=2
           Ciklus amíg k≤c.n és k≤c.s
             Ha x.n mod k =0 és x.s mod k =0
                akkor x.n:=x.n div l; x.s:=x.s div l
             különben k:=k+1
           Ciklus vége
Eljárás vége.
```

2. feladat: Hangrend (24 pont)

A kapott szó betűit kell vizsgálni! El kell dönteni a szóról, hogy van-e benne magas hangrendű magánhangzó, illetve van-e benne mély hangrendű magánhangzó! Ezután a kérdés egyszerűen megválaszolható.

```
Hangrend(szó,h):
  magas:=hamis; mély:=hamis
  Ciklus i=1-tól hossz(szó)-ig
    Ha szó(i)∈{'a','á','o','ó','u','ú'} akkor mély:=igaz
    Ha szó(i)∈{'e','é','i','í','ö','ő','ü','ű'}
       akkor magas:=igaz
  Ciklus vége
  Ha magas és mély akkor h:="Vegyes"
  különben ha magas akkor h:="Magas"
  különben ha mély akkor h:="Mély"
Eljárás vége.
```

3. feladat: Lövészverseny (19 pont)

A. A verseny valamely időszakában álltak az első helyen – maximumkiválasztás közben az adott elem éppen maximális lett.

B. A verseny valamely időszakában álltak az utolsó helyen – minimumkiválasztás közben az adott elem éppen maximális lett.

```
Lövész(n,t,adb,a,bdb,b):
  ma:=1; adb:=1; a(adb):=1
  Ciklus i=2-től n-ig
    Ha t(i)≥t(ma) akkor ma:=i; adb:=adb+1; a(adb):=i
  Ciklus vége
  mi:=1; bdb:=1; b(bdb):=1
  Ciklus i=2-től n-ig
    Ha t(i)≤t(mi) akkor mi:=i; bdb:=bdb+1; b(bdb):=i
  Ciklus vége
Eljárás vége.
```

Kilencedik-tizedik osztályosok

1. feladat: Rendszergazda (20 pont)

A feladat absztrakt modellje: ismerünk két intervallumhalmazt, s meg kell adnunk ezen halmazok metszetét, illetve a vizsgált időszaknak a két halmaz unióján kívüli részét.

Mivel N napról van szó, vegyünk egy N elemű tömböt (t), ami kezdetben 0-val van feltöltve! Amely napokon valamelyik rendszergazda szabadságon van, ott ezen tömbelemek értékét növeljük meg eggyel! Ezután a 0 értékű elemek lesznek a biztonságos napok (itt van mind a 2 rendszergazda), a 2 értékűek pedig a veszélyesek (mindkettő szabadságon van).

```
Rendszergazda(n,t,b,tb,v,tv):
  b:=0; v:=0; t(0):=-1; t(n+1):=-1
  Ciklus i=1-től n+1-ig
    Ha t(i)=0 és t(i)≠t(i-1) akkor b:=b+1; tb(b).e:=i
    különben ha t(i)≠0 és t(i-1)=0 akkor tb(b).u:=i-1
    Ha t(i)=2 és t(i)≠t(i-1) akkor v:=v+1; tv(v).e:=i
    különben ha t(i)≠2 és t(i-1)=2 akkor tv(v).u:=i-1
  Ciklus vége
Eljárás vége.
```

2. feladat: Tömörítés (20 pont)

A képet soronként azonos jelekből álló sorozatokra kell bontani, meghatározva, hogy egymás mellett milyen jelből hány egyforma volt!

```
Tömörítés(n,m,t,d,jel,db):
  Ciklus a=1-től n-ig
    b:=1; d(i):=0
    Ciklus amíg b≤m
      c:=b
      Ciklus amíg c≤m és t(a,b)=t(a,c))
        c:=c+1
      Ciklus vége
      d(i):=d(i)+1; db(i,d(i)):=c-b; jel(i,d(i)):=t(a,b)
      b:=c
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

3. feladat: Huszár (15 pont)

Ez egy gráf mélységi bejárás adott lépésszám korláttal, ahol a sakktáblán *-gal jelöljük azokat a mezőket, ahol már jártunk.

```
Tesz(n,sor,oszlop,max,t):
  Ha sor≥1 és sor≤8 és oszlop≥1 és oszlop≤8 és max≤n
     akkor Ha t(sor,oszlop)=' '
              akkor t(sor,oszlop):='*'
                    Tesz(sor-1,oszlop-2,max+1)
                    Tesz(sor-2,oszlop-1,max+1)
                    Tesz(sor+1,oszlop-2,max+1)
                    Tesz(sor-2,oszlop+1,max+1)
                    Tesz(sor-1,oszlop+2,max+1)
                    Tesz(sor+2,oszlop-1,max+1)
                    Tesz(sor+1,oszlop+2,max+1)
                    Tesz(sor+2,oszlop+1,max+1)
          max:=max+1
Eljárás vége.
```

4. feladat: Olimpiai láng (20 pont)

Ez egy mohó stratégiával megoldható feladat. Az első városból nyilván kell indítani egy futót. Ha ő a lehető legkésőbb adja át az olimpiai lángot a következő futónak, akkor ezzel a legközelebb jutunk a célig. Tehát innen ugyanezzel a módszerrel folytatva, megkapjuk a lángvivők minimális számát.

Ehhez sorba kellene rendezni a futókat indulási helyüknek a kezdő várostól vett távolsága alapján. Ehelyett azonban megtehetjük az, hogy minden kilométerre megadjuk, hogy onnan milyen sorszámú futó indulhat.

```
Láng(N,x,M,y):
  Ind:=(0,…,0)
  Ciklus i=1-től N-ig
    Ind(x(i)):=i
  Ciklus vége
  M:=1; y(1):=Ind(0); U1:=H
  Ciklus j=1-től K-ig
    Ha Ind(j)>0
       akkor Ha j≤U1 akkor U2:=j
             különben M:=M+1; y(M):=Ind(U2); U1:=U2+H; U2:=j
  Ciklus vége
  Ha U1<K akkor M:=M+1; y(M):=Ind(U2)
Eljárás vége.
```

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Rendszergazda (14 pont)

Ez a feladat a dátum kezelésben nehezebb az előző korcsoport hasonló feladatánál. Emiatt a beolvasásnál a (hónap,nap) párokat napsorszámmá alakítjuk, kiírásnál pedig a napsorszámból újra (hónap,nap) párokat készítünk. A beolvasás előkészítéseként ki kell számolni az év napjai számát, valamint szükség esetén a február napjai számát az előre tárolt tömbben 29-re kell átírni.

```
Gazda(év,n,t,b,tb,v,tv):
  Ha év mod 4 >0 akkor szökő:=hamis
  különben ha év mod 100=0 és év mod 400>0 akkor szökő:=hamis
  különben szökő:=igaz
  Ha szökő akkor n:=366; hó(2):=29 különben n:=365
  Beolvasás
  b:=0; v:=0; t(0):=-1; t(n+1):=-1
  Ciklus i=1-től n+1-ig
    Ha t(i)=0 és t(i)≠t(i-1) akkor b:=b+1; tb(b).e:=i
    különben ha t(i)≠0 és t(i-1)=0 akkor tb(b).u:=i-1
    Ha t(i)=2 és t(i)≠t(i-1) akkor v:=v+1; tv(v).e:=i
    különben ha t(i)≠2 és t(i-1)=2 akkor tv(v).u:=i-1
  Ciklus vége
Eljárás vége.
```

2. feladat: Zászló (15 pont)

Az első sor kódolása azonos a kisebb korcsoport hasonló feladatának megoldásával. A többi sortnak pedig csak azt a részét változtatjuk, ami az előző sorhoz képest változik.

```
Zászló(n,m,t,db,sor,tól,ig,jel):
  db:=0;
  Ciklus a=1-től N-ig
    b:=1
    Ciklus amíg b≤m
      Ciklus amíg b≤m és t(a,b)=t(a-1,b)
        b:=b+1
      Ciklus vége
      Ha b≤m akkor c:=b
                   Ciklus amíg c≤m és t(a,b)=t(a,c) és
                               t(a,c)≠t(a-1,c)
                     c:=c+1
                   Ciklus vége
                   db:=db+1; sor(db):=a; tól(db):=b
                   ig(db):=c-1; jel(db):=t(a,b)
                   b:=c
    Ciklus vége
  Ciklus vége
Eljárás vége.
```

3. feladat: Sakk (15 pont)

Ez a feladat is hasonló az előző korcsoport feladatához. A különbség: nem azt kell nézni, hogy egy mezőre eljuthat-e a huszár, hanem azt, hogy minimum hány lépésben (a még nem elért mezőknél ez legyen -1).

Emiatt más megoldási elv kell: szélességi bejárást, vagy ahhoz hasonló (kevésbé hatékony) algoritmust használhatunk. Mi most ezt választjuk (8*8*12 lépés nem számottevő idejű), tudva azt, hogy legfeljebb 12 lépésben bármely mező elérhető.

```
Elhelyez(n,min,t):
  Ciklus z=1-től 12-ig
    Ciklus sor=1-től 8-ig
      Ciklus oszlop=1-től 8-ig
        Ha t(sor,oszlop)=min-1
           akkor Tesz(sor-1,oszlop-2,min)
                 Tesz(sor-2,oszlop-1,min)
                 Tesz(sor+1,oszlop-2,min)
                 Tesz(sor-2,oszlop+1,min)
                 Tesz(sor-1,oszlop+2,min)
                 Tesz(sor+2,oszlop-1,min)
                 Tesz(sor+1,oszlop+2,min)
                 Tesz(sor+2,oszlop+1,min)
      Ciklus vége
    Ciklus vége
    min:=min+1
  Ciklus vége
Eljárás vége.
```

```
Tesz(sor,oszlop,max,k):
  Ha sor≥1 és sor≤8 és oszlop≥1 és oszlop≤8
     akkor Ha t(sor,oszlop)=-1 akkor t(i,j):=k
Eljárás vége.
```

4. feladat: Staféta (15 pont)

Ez is hasonló az előző korcsoport olimpiai lángos feladatához, itt azonban a futók különböző távolságokat vállalnak. Mindenkiről azt ismerjük, hogy mettől meddig vállalná a futást. A megoldást úgy alakítjuk át, hogy az egyes kilométerekhez nemcsak azt tároljuk, hogy melyik futó indulna innen, hanem azt is, hogy meddig vinné a lángot. Ha valamelyik helyről többen is indulnának, azt választjuk, aki a legmesszebb menne.

```
Láng(N,x,M,y):
  Ind().az:=(0,…,0)
  Ciklus i=1-től N-ig
    Ha Ind(x(i)).az=0 vagy Ind(x(i)).vég<y(i)
       akkor Ind(x(i)).az:=i; Ind(x(i)).vég:=y(i)
  Ciklus vége
  M:=1; y(1):=Ind(0); U1:=Ind(0).vég
  Ciklus j=1-től K-ig
    Ha Ind(j)>0
       akkor Ha j≤U1 akkor
                Ha Ind(U2).vég<Ind(j).vég akkor U2:=j
             különben M:=M+1; y(M):=Ind(U2).az
                      U1:=Ind(U2).vég; U2:=j
  Ciklus vége
  Ha U1<K akkor M:=M+1; y(M):=Ind(U2).az
Eljárás vége.
```

5. feladat: Munka (16 pont)

Jelölje Fél a gyártási idők összegének a felét. Ha az optimális megoldásban az egyik gép T1, a másik T2 ideig dolgozik, akkor min(T1,T2) a legkisebb olyan érték, amely Fél-nél nem nagyobb, és az alkatrészeket el lehet úgy osztani a két gépre, hogy az egyik T1, a másik T2 ideig dolgozik.

Tehát az optimális megoldási érték kiszámításához keressük azt a legnagyobb X≤Fél értéket, amelyre teljesül, hogy X előállítható gyártási idők összegeként. Bontsunk részproblémákra, minden X≤Fél, és 1≤i≤N párra tekintsük azt a problémát, hogy X előállítható-e az első i darab gyártási idő összegeként? Jelölje ezt E(X,i). Rekurzív összefüggések E(X,i)-re (az i-edik alkatrész gyártási ideje m(i)):

E(m(1),1)=Igaz, E(x,1)=Hamis, ha x≠m(1)

E(x,i)=E(x,i-1) vagy x≥m(i) és E(x-m(i),i-1) ha i>1

E(X,i) értékeket táblázatkitöltés módszerével számítjuk. Majd meghatározzuk a legnagyobb X-et, amelyre E(X,N) Igaz, és visszafejtéssel előállítunk egy megoldást.

```
Munka(N,m,idő,edb,első,mdb,második):
  t:=0
  Ciklus i=1-től N-ig
    t:=t+m(i)
  Ciklus vége
  Fél:=t div 2
  E(1..Fél,1):=(hamis,…,hamis); E(m(1),1):=igaz
  E(0,1..N):=(igaz,…,igaz)
  Ciklus i=2-től N-ig
    Ciklus x=1-től Fél-ig
       E(x,i):=E(x,i-1) vagy x≥m(i) és E(x-m(i),i-1)
    Ciklus vége
  Ciklus vége
  t1:=Fél
  Ciklus amíg nem E(t1,N)
    t1:=t1-1
  Ciklus vége
  idő:=t-t1
  x:=t1; i:=N; edb:=0; mdb:=0
  Ciklus amíg x>0
    Ciklus amíg i>0 és E(x,i)
      i:=i-1
    Ciklus vége
    edb:=edb+1; első(edb):=i+1; x:=x-m(i+1); m(i+1):=0
  Ciklus vége
  Ciklus i=1-től N-ig
    Ha m(i)>0 akkor mdb:=mdb+1; második(mdb):=i
  Ciklus vége
Eljárás vége.
```

2009. Harmadik forduló

Ötödik-nyolcadik osztályosok

1. feladat: Szmogriadó (24 pont)

A. a legszennyezettebb nap – maximumkiválasztás

B. a mérés közbeni javítások száma – jó működés után következő 0 értékű mérések száma

C. azon napok, amikor szmogriadót kellett elrendelni – a 100% alattiból 100% felettibe váltások helye

```
Szmogriadó(n,t,a,b,c,ct):
  a:=1
  Ciklus i=2-től n-ig
    Ha t(i)>t(a) akkor a:=i
  Ciklus vége
  b:=0
  Ciklus i=2-től n-ig
    Ha t(i-1)=0 és t(i)>0 akkor b:=b+1
  Ciklus vége
  c:=0
  Ciklus i=2-től n-ig
    Ha t(i-1)≤100 és t(i)>100 akkor c:=c+1; ct(c):=i
  Ciklus vége
Eljárás vége.
```

<u>2. feladat</u>: Nevek (24 pont)

Ki kell számolni minden hasáb szélességét, s közben a megfelelő helyre kiírni a neveket!

```
Nevek(N,K,név):
  poz:=1; sor:=1; h:=0
  Ciklus i=1-től N-ig
    Pozícionálás(sor,poz); Ki: név(i)
    Ha hossz(név(i))>h akkor h:=hossz(név(i))
    Ha i mod K=0 akkor sor:=1; poz:=poz+h+1; h:=0
              különben sor:=sor+1
  Ciklus vége
Eljárás vége.
```

<u>3. feladat</u>: Kémek (27 pont)

A. a főnök – megkeressük azt, aki senkinek sem beosztottja

B. egy tag, akinek a legtöbb közvetlen beosztottja van – megszámoljuk mindenkire a beosztottak számát, majd egy maximumkiválasztás

C. egy tag, aki a „legmesszebb" van a főnöktől – kiszámoljuk mindenkire a főnök távolságát, majd maximumkiválasztás

```
Kémek(n,x,):
  Ciklus i=1-től n-1-ig
    be(x(i,2)):=be(x(i,2))+1; fő(x(i,1)):=x(i,2)
  Ciklus vége
  a:=0; főnök:=hamis
  Ciklus amíg nem főnök
    j:=1; a:=a+1
    Ciklus amíg j≤n és x(j,1)≠a
      j:=j+1
    Ciklus vége
    főnök:=(j>n)
  Ciklus vége
  b:=1
  Ciklus i=2-től n-ig
    Ha be(i)>be(b) akkor b:=i
  Ciklus vége
```

```
  táv(a):=0; c:=a; fő(a):=0
  Ciklus i=1-től n-ig
    Ha fő(i)>0 akkor j:=i; táv(i):=0
                     Ciklus amíg j>0
                       táv(i):=táv(i)+1; j:=fő(j)
                     Ciklus vége
                     Ha táv(i)>táv(c) akkor c:=i
  Ciklus vége
Eljárás vége
```

Csak főnök van (N=1 ⇒ A=1, B=1, C=1)	1+1+1 pont
Egyetlen beosztott (N=2, (1,2) ⇒ A=2, B=2, C=1)	2+2+2 pont
Több közvetlen beosztott (N=3, (2,1), (3,1) ⇒ A=1, B=1, C=2 vagy 3)	2+2+2 pont
Több beosztott (N=5, (1,2),(3,1),(4,3),(5,3) ⇒ A=2, B=3, C=4 vagy 5)	2+2+2 pont
Nagy teszt (N=9, (1,2),(3,1),(4,3),(5,3),(2,6),(7,6),(8,3),(9,8) ⇒ A=6, B=3, C=9)	2+2+2 pont

Kilencedik-tizedik osztályosok

1. feladat: Játék (20 pont)

Legyem Maxi(i,j) az (i.j) pozíciótól jobbra lefelé található gyöngyök számának maximuma! Az első részfeladat megoldása T(i,j)+MAXI(i,j) értékének maximuma. Ennek helye egyben az első pont, amit az optimális úton el kell érni. A második pont ettől jobbra lefelé található, az a hely, ahol a Maxi(i2,j2) megegyezett a tőle közvetlenül jobbra vagy közvetlenül alatta levő gyöngyök számával.. Az út megadása ezután nagyon egyszerű, az első pont oszlopáig jobbra megyünk, utána a második pont soráig lefelé megyünk (közben áthaladunk az első ponton), s végül elmegyünk a jobb szélre, majd le a jobb alsó sarokba.

```
Játék(M,N,T,mego,i1,i2,j1,j2,lép):
  Maxi(M,N):=0; mego:=0;
  Maxi(1..M,N+1):=(0,…,0); Maxi(M+1,1..N):=(0,…,0)
  Ciklus i=M-től 1-ig -1-esével
    Ciklus j=N-től 1-ig -1-esével

Maxi(i,j):=Max(T(i,j+1),T(i+1,j),Maxi(i,j+1),Maxi(i+1,j))
      Ha T(i,j)+Maxi(i,j)>mego
         akkor mego:=T(i,j)+Maxi(i,j); i1:=i; j1:=j
    Ciklus vége
  Ciklus vége

  max2:=Maxi(i1,j1); i2:=i1; j2:=j1
  Ciklus amíg i2=i1 és j2=j1 vagy max2≠T(i2,j2)
    Ha max2=T(i2,j2+1) akkor j2:=j2+1
    különben ha max2=T(i2+1,j2) akkor i2:=i2+1
    különben ha Maxi(i2,j2+1)>Maxi(i2+1,j2) akkor j2:=j2+1
    különben i2:=i2+1
  Ciklus vége
  i:=1; j:=1; lép:=""
  Ciklus k=1-től M+N-2-ig
    Ha j<j1 vagy i=i2 és j<N akkor lép:=lép+"J"; j:=j+1
                          különben lép:=lép+"L"; i:=i+1
  Ciklus vége
Eljárás vége.
```

2. feladat: Kísérlet (15 pont)

Először minden időpontra számoljuk ki, hogy attól kezdődően hány sejt keletkezett, illetve addig hány sejt pusztult el! Ezután minden a időponthoz keressük meg azt a legközelebbi b időpontot, amikor legalább K sejt volt életben (nincs életben, aki legfeljebb `a-1`-ig élt, vagy `b+1`-től kezdődően született). A feladat megoldása az ilyen (a,b) intervallumok közül a legrövidebb megadása.

```
Kísérlet(N,u,v,ma,mb):
  Kezd:=(0,…,0); Vég:= (0,…,0); maxb:=0
  Ciklus i=1-től N-ig
    Kezd(u(i)):=Kezd(u(i))+1; Vég(v(i)):=Vég(v(i))+1
    Ha v(i)>maxb akkor maxb:=v(i)
  Ciklus vége
  Kezd(maxb+1):=0; Vég(0):=0
  Ciklus a=maxb-től 1-ig -1-esével
    Kezd(a):=Kezd(a)+Kezd(a+1)
  Ciklus vége
  Ciklus a=1-től maxb-ig
    Vég(a):=Vég(a)+Vég(a-1)
  Ciklus vége
  ma:=0; mb:=M; a:=1; b:=1
  Ciklus amíg b≤maxb
    Ciklus amíg b≤maxb és N-(Vég(a-1)+Kezd(b+1))<K
      b:=b+1
    Ciklus vége
    Ha b≤maxb akkor
       Ha (b-a)<(mb-ma) akkor ma:=a; mb:=b
       a:=a+1
       Ha b<a akkor b:=a
  Ciklus vége
Eljárás vége.
```

<u>3. feladat</u>: Rendezvény (20 pont)

A feladat modellje: intervallumokat kell minimális számú csoportra osztani úgy, hogy a csoportokon belüli intervallumok diszjunktak legyenek!

A megoldási elv: Mivel az intervallumok a kezdetük szerint vannak sorbarendezve, minden intervallumot próbáljunk betenni abba a terembe, ahova betehető. Ha egyikbe sem tehető be, akkor új termet nyitunk neki.

```
Termek(N,M,a,b,TVég,TV):
  TVég:=(0,…,0); TV(1..M,0).b:=(0,…,0); t:=0
  Ciklus i=1-től N-ig
    x:=1; tb:=0
    Ciklus amíg x<=t és TV(x,TVég(x)).b>a
      x:=x+1
    Ciklus vége
    TVég(x):=TVég(x)+1
    TV(x,TVég(x)).b:=b(i); TV(x,TVég(x)).az:=i
    Ha x>t akkor t:=t+1
  Ciklus vége
Eljárás vége.
```

4. feladat: Kémek (20 pont)

A kémszervezete egy fával lehet leírni. Ebben a fában meg kell keresni a gyökérhez (főnökhöz) legközelebbi olyan elemet, akinek legfeljebb egy közvetlen beosztottja van. Az összes ilyen közül azt kell választani, akinek a részfája (beosztottjai) elemszáma a lehető legnagyobb!

```
Kémek(n,f,b,s):
  Ciklus i=1-től n-1-ig
    x(f(i)).fel:=b(i); x(b(i)).kbe:=x(b(i)).kbe+1
  Ciklus vége
  Ciklus i=1-től n-ig
    Ha x(i).fel≠0
       akkor j:=x(i).fel
             Ciklus amíg j≠0
               x(j).be:=x(j).be+1; j:=x(j).fel
               x(i).táv:=x(i).táv+1
             Ciklus vége
  Ciklus vége
  x(0).táv:=maxint; x(0).be:=-1; s:=0
  Ciklus i=1-től n-ig
    Ha x(i).kbe<2
       akkor Ha x(i).táv<x(s).táv vagy
                x(i).táv=x(s).táv és x(i).be>x(s).be
             akkor s:=i
  Ciklus vége
Eljárás vége.
```

Tizenegyedik-tizenharmadik osztályosok

1. feladat: Sejtek (15 pont)

A feladat modellje: egy intervallumhalmaz alapján meg kell adni azt a legszűkebb intervallumot, amely legalább K teljes intervallumot tartalmaz!

Első lépésként minden u időponthoz adjuk meg az u-ban kezdődő intervallumok végpontjainak listáját, illetve minden v időponthoz a v-ben végződő intervallumok kezdeteinek listáját! Ezután minden a kezdőponthoz keressük meg azt a legkisebb intervallumhatárt, amely legalább K intervallumot tartalmaz, s ezek közül válasszuk a minimálisat!

```
Sejtek(N,u,v,ma,mb):
  Ciklus i=1-től N-ig
    Lefoglal(Guv); Guv^.x:=v(i); Guv^.csat:=Vég(u(i))
    Vég(u(i)):=Guv
    Lefoglal(Gvu); Gvu^.x:=u(i); Gvu^.csat:=Kezd(v(i))
    Kezd(v(i)):=Gvu
  Ciklus vége
  jó:=0; ma:=0; mb:=M; a:=1; b:=0
```

```
  Ciklus amíg a≤M és b≤M
    Ciklus amíg b<m és jó<K
      b:=b+1; p:=Kezd(b)
      Ciklus amíg p≠sehova
        Ha a≤p^.x akkor jó:=jó+1; p:=p^.csat
      Ciklus vége
    Ciklus vége
    Ha jó≥K és b-a<mb-ma akkor ma:=a; mb:=b
    Ha b≤M akkor p:=Vég(a)
                 Ciklus amíg p≠sehova
                   Ha p^.x≤b akkor jó:=jó-1; p:=p^.csat
                 Ciklus vége
                 a:=a+1
  Ciklus vége
Eljárás vége.
```

2. feladat: Mérőkannák (15 pont)

Ez egy klasszikus gráf szélességi bejárás, ahol a gráf csúcsai a lehetséges kanna-állapotok. Az egyik nehézség a sok állapotátmenet kezelése. A megoldás másik nehézsége az öntögetési sorrend megadása. Emiatt a végállapotból vissza kell jutni a kezdőállapotba, majd onnan felépíteni az öntögetési sorrendet.

```
Kannák(A,B,L,lehet,m,Lépés):
  EA(0..A,0..B).x:=(-1,…,-1)
  S:=ÜresSor; EA(0,0).x:=0; U.x:=0; U.y:=0; Sorba(S,U)
  Ciklus amíg Nem üresSor?(S)
    Sorból(S,U)
    Ha U.x≠L akkor
       Ha EA(A,U.y).x<0 akkor {1. lépés}
          V.x:=A; V.y:=U.y; EA(V.x,V.y):=U; Sorba(S,V)
       Ha EA(U.x,B).x<0 akkor {2. lépés}
          V.x:=U.x; V.y:=B; EA(V.x,V.y):=U; Sorba(S,V)
       ha EA(0,U.y).x<0 akkor {3. lépés}
          V.x:=0; V.y:=y; EA(V.x,V.y):=U; Sorba(S,V)
       Ha EA(U.x,0).x<0 akkor {4. lépés}
          V.x:=U.x; V.y:=0; EA(V.x,V.y):=U; Sorba(S,V)
       Ha U.x+U.y≤B akkor V.x:=0; V.y:=U.x+U.y
                különben V.x:=U.x-(B-U.y); V.y:=B
       Ha EA(V.x,V.y).x<0 akkor {5. lépés}
          EA(V.x,V.y):=U; Sorba(S,V)
       Ha U.x+U.y≤A akkor V.y:=0; V.x:=U.x+U.y
                különben V.y:=U.y-(A-U.x); V.x:=A
       Ha EA(V.x,V.y).x<0 akkor {6. lépés}
          EA(V.x,V.y):=U; Sorba(S,V)
  Ciklus vége
```

```
  lehet:=hamis; y:=0
  Ciklus amíg y≤B és EA(L,y).x≤
    y:=y+1
  Ciklus vége
  lehet:=y≤B
  Ha lehet
     akkor U.y:=y; U.x:=L; m:=0
           Ciklus amíg U.x≠0 vagy U.y≠
             m:=m+1; LepSor(m):=U; U:=EA(U.x,U.y)
           Ciklus vége
           U.x:=0; U.y:=0
           Ciklus i=m-től 1-ig -1-esével
             V:=LepSor(i)
             Ha V.x=A és U.y=V.y akkor Lép:=1
             különben ha U.x=V.x és V.y=B akkor Lép:=2
             különben ha V.x=0 és U.y=V.y akkor Lép:=3
             különben ha U.x=V.x és V.y=0 akkor Lép:=4
             különben ha V.x=0 és V.y=U.x+U.y vagy
                    V.x=U.x-(B-U.y))and(V.y=B) akkor Lép:=5
             különben Lép:=6
             Lépés(m-i+1):=Lép; U:=V
           Ciklus vége
Eljárás vége.
```

3. feladat: Koncert (15 pont)

Tekintsük azt az irányítatlan gráfot, amelynek pontjai az ülőhelyek (sorszámai). A p és q pont között akkor és csak akkor legyen i-vel címkézett él, ha az i-edik igény a p és q ülőhelyet igényelte. Tehát két pont között több él is lehet.

Tekintsük a gráf összefüggő komponenseit. Egy összefüggő komponens a pontoknak olyan maximális halmaza, amelyre teljesül, hogy bármely két pont között van út a gráfban. Ha egy ilyen komponens nem tartalmaz kört, azaz fa, akkor a pontok száma eggyel nagyobb, mint az élek száma. Tehát egy kivételével minden ponthoz tudunk különböző igényt rendelni. Ezt egy mélységi bejárással megtehetjük. Ha a komponensben van kör, akkor pedig minden ponthoz tudunk különböző igényt rendelni a következő módon. Készítsünk egy olyan p0 gyökerű mélységi feszítőfát, ahol p0 körben lévő pont. Ekkor a bejárás során biztosan találunk egy q--p0 visszaélt. Ezen él címkéjéhez rendeljük a p0 pontot, a többi, nem gyökér ponthoz pedig a bejárásban a fa-élet (amellyel az új ponthoz jutunk a bejárásban).

Két mélységi bejárást hajtunk végre. Az elsővel minden komponenshez meghatározunk egy körben lévő pontot, ha van. Ha van körben lévő pont, akkor a második bejárást, amely a hozzárendelést végzi, ebből a pontból indítjuk.

```
Koncert:
  Hany:=N
  Ciklus p=1-től M-ig
    Ha Szin(p)=Fehér akkor
      p0:=MelyBejar1(p)
      Ha p0=0 akkor Hany:=Hany-1; MelyBejar2(0,p)
      különben MelyBejar2(p0,p0)
  Ciklus vége
Eljárás vége.
```

```
MelyBejar1(p):
  MelyBejar1:=0; Szin(p):=Szürke; pq:=G(p)
  Ciklus amíg pq≠0
    qi:=Elek(pq).kipont
    Ha p=Ker(qi).a akkor q:=Ker(qi).b
                különben q:=Ker(qi).a
    Ha Szin(q)=Fehér akkor
      Apa(q):=qi; p0:=MelyBejar1(q)
      Ha p0≠0 akkor MelyBejar1:=p0
      különben ha qi≠Apa(p) akkor MelyBejar1:=p; Apa(p):=qi
    Elágazás vége
    pq:=Elek(pq).csat
  Ciklus vége
Függvény vége.
```

```
MelyBejar2(p0,p):
  Szin(P):=Fekete; pq:=G(p)
  Ciklus amíg pq≠0
    qi:=Elek(pq).kipont
    Ha p=Ker(qi).a akkor q:=Ker(qi).b
                különben q:=Ker(qi).a
    Ha Szin(q)≠Fekete akkor KiUl(q):=qi; MelyBejar2(p0,q)
    különben ha q=p0 akkor KiUl(p0):=qi
    pq:=Elek(pq).csat
  Ciklus vége
Eljárás vége
```

<u>4. feladat</u>: Ügynökök (15 pont)

A feladat modellje egy fa. A fában ki kell jelölni minimális számú csomópontot, amelyeknek legfeljebb 2 közvetlen leszármazottjuk van és a belőlük kiinduló részfák elemszáma a fa elemszámának legalább a fele! Az biztosan teljesül, hogy ha egy elemet kiválasztottunk, akkor a belőle kiinduló részfából nem kell másik elemet választani!

A megoldásban számoljuk mindenkire a közvetlen főnöke sorszámát, a közvetlen beosztottjai számát, az összes beosztottja számát, valamint azt, hogy ki a legmagasabban levő olyan főnöke, akinek legfeljebb 2 beosztottja van!

Ha mindezt kiszámoltuk, rendezzük sorba az elemeket a beosztottjaik száma szerint, de az eredmény kiszámításában csak azokat vegyük figyelembe, amelyeknek nincs legfeljebb 2 közvetlen beosztottal rendelkező főnöke!

```
Ügynök(n,a,b,k):
  Ciklus i=1-től n-1-ig
    x(a(i)).főnök:=b(i); x(b(i)).kbe:=x(b(i)).kbe+1
  Ciklus vége
  Ciklus i=1-től n-ig
    Ha x(i).főnök≠0 akkor j:=x(i).főnök
                          Ciklus amíg j≠0
                            Ha x(j).kbe≤2 akkor x(i).lm:=j
                            x(j).be:=x(j).be); j:=x(j).főnök
                          Ciklus vége
  Ciklus vége
```

```
  x(0).be:=-1; db:=0
  Ciklus i=1-től n-ig
    Ha x(i).kbe≤2 akkor inc(db); csere(db):=i
  Ciklus vége
  i:=1; s:=0; k:=0
  Ciklus amíg i≤db és 2*s<n
    max:=i
    Ciklus j=i+1-től db-ig
      ha x(csere(j)).be>x(csere(max)).be akkor max:=j
    Ciklus vége
    y:=csere(i); csere(i):=csere(max); csere(max):=y
    Ha x(csere(i)).lm=0 akkor s:=s+x(csere(i)).be+1; k:=k+1
    i:=i+1
  Ciklus vége
Eljárás vége.
```

<u>5. feladat</u>: Házak (15 pont)

Ez egy geometriai algoritmus. balról jobbra nézve a házakat azt kell vizsgálni, hogy az aktuális ház jobb felső sarka árnyékban van-e az utolsó nem árnyékban levő ház miatt, azaz az annak jobb felső sarkán átmenő fénysugár irányú egyenestől az aktuális ház jobb felső sarka balra vagy jobbra van-e!

Mivel a bemenetben a bal felső sarkokat és a házak szélességét kapjuk, beolvasáskor kell kiszámolni a jobb felső sarkot

```
Házak(n,ház,dx,dy,db,árnyék):
  i:=1; db:=1; árnyék(1):=1
  Ciklus j=2-től n-ig
    Ha irány(ház(i).x-dx,ház(i).y+dy,ház(i).x,ház(i).y,
             ház(j).x,ház(j).y)>0
       akkor i:=j; db:=db+1; árnyék(db):=j
  Ciklus vége
Eljárás vége.
```

```
irány(a,b,c,d,p,q):
  cr:=(c-a)*(q-b)-(p-a)*(d-b)
  Ha cr<0 akkor irány:=-1
  különben ha cr>0 akkor irány:=1
  különben irány:=0
Függvény vége.
```

Mivel a=c-dx és b=d+dy, ezért cr másképp is számolható:

```
  cr:=dx*(q-b)+(p-a)*dy
```