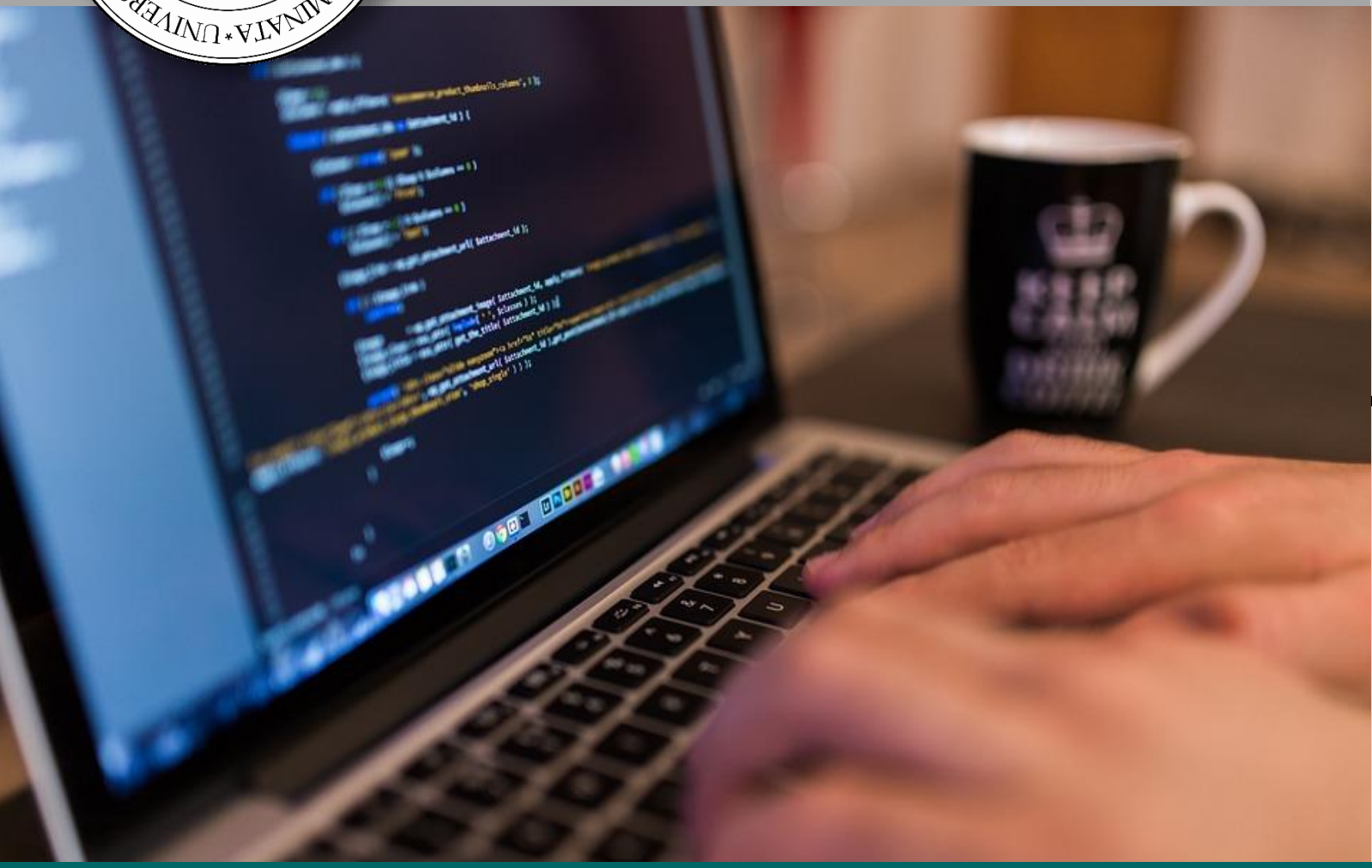
Belépő a tudás közösségébe

Informatika szakköri segédanyag



Összetett programozási tételek 1

Heizlerné Bakonyi Viktória, Horváth Győző, Menyhárt László, Szlávi Péter, Törley Gábor, Zsakó László

Szerkesztő: Abonyi-Tóth Andor, Zsakó László

A kiadvány „A felsőoktatásba bekerülést elősegítő készségfejlesztő és kommunikációs programok megvalósítása, valamint az MTMI szakok népszerűsítése a felsőoktatásban” (EFOP-3.4.4-16-2017-006) című pályázat keretében készült 2017-ben.

SZÉCHENYI 2020

MAGYARORSZÁG KORMÁNYA

Európai Unió
Európai Szociális Alap

BEFEKTETÉS A JÖVŐBE

Eötvös Loránd Tudományegyetem
Informatikai Kar

Feladataink egy jelentős csoportjában egyetlen bemenő sorozat alapján egy vagy több sorozatot kell előállítanunk.

Másolás

Ezt a feladattípust is konkrét példákkal kezdjük, mint tettük azt az előző fejezetben.

F1. Egy osztály tanulóinak átlageredménye alapján határozzuk meg, hogy bizonyítványukba *jeles, jó, közepes* vagy *elégséges* kerül-e! Tegyük fel, hogy bukott tanuló nincs!

F2. Egy szöveg minden magánhangzóját cseréljük ki az *e* betűre!

E feladatok közös jellemzője, hogy az eredmény mindig ugyanannyi elemszámú, mint a bemenet volt, s az i. tagját a bemenet i. tagjából lehet meghatározni. Tehát a bemenő sorozatot le kell másolni, s közben egy – elemre vonatkozó – átalakítást lehet végezni rajta.[1]

```
Függvény:
  f:H_Elemtípus→G_Elemtípus

Változók:
  N:Egész          {a feldolgozandó sorozat elemei száma}
  X:Tömb(1..N:H_Elemtípus)     {a feldolgozandó sorozat}
  Y:Tömb(1..N:G_Elemtípus)      {a feldolgozott sorozat}
```

A bemenő sorozatból az eredmény elemenként kiszámítható, így nagyon egyszerű megoldást kapunk. Az első változatban minden elemből számítjuk az eredmény elemeit:

```
Másolás(N,X,Y):
  Ciklus i=1-től N-ig
    Y[I]:=f(X[i])
  Ciklus vége
Eljárás vége.
```

A második változatban egyes elemeket lemásolunk, más elemekből számítunk.

```
Másolás(N,X,Y):
  Ciklus i=1-től N-ig
    Ha T(X[i]) akkor Y[i]:=f(X[i]) különben Y[i]:=X[i]
  Ciklus vége
Eljárás vége.
```

A fejezet elején szereplő példák közül az egyik megoldása következik.

[1] Gyakran e tulajdonságot hívják *elemenkénti feldolgozhatóság*nak.

F2. Magánhangzóra cserélés

Példa

```
elemek száma: N
bemenet     : X N db elem
kimenet     : Y N db elem
f(b)        : 'e', ha Magánhangzó(b); b, különben

Másolás(N,X,Y):
  Ciklus i=1-től N-ig
    Ha Magánhangzó(X[I]) akkor Y[I]:='e' különben Y[I]:=X[I]
  Ciklus vége
Eljárás vége.
```

Minta kódok.

C++	cpp/2_Masolas/feladat.cpp
C#	cs/2_Masolas/feladat.cs
Java	java/2_Masolas/feladat.java
Pascal	pas/2_Masolas/feladat.pas
Python	py/2_Masolas/feladat.py

Kiválogatás

Nem minden esetben kell lemásolni a teljes bemenő sorozatot, hanem annak csupán egy részére kell korlátozni a vizsgálatot. Legegyszerűbb esetben a *vizsgálat* után mindössze elemmásolás következik. Ilyen feladatok szerepelnek e fejezetben.

F3. Egy személyzeti nyilvántartásban emberek neve és személyi száma szerepel, adjuk meg a 20 évnél fiatalabb lányokat!

F4. Adjuk meg egy természetes szám összes osztóját!

F5. Adjuk meg egy osztály azon tanulóit, akik jeles átlagúak!

A feladat részben a keresésre hasonlít, részben pedig a megszámlálásra. Adott tulajdonságú elemet kell megadni, de nem egyet, hanem az összeset; illetve nem megszámolni kell az adott tulajdonságú elemeket, hanem megadni.

Az eredmény tárolásához egy új tömböt használunk, amelynek elemszámát előre sajnos nem tudjuk megállapítani, csupán egy felső korlátot ismerünk: a lehetséges elemek számát.

Többféle megoldást készíthetnénk a feladatra. Az első változatban a keresett elemek sorszámait gyűjtjük ki egy vektorba. A megoldás hasonlít a megszámolásra, csupán a számolás mellett még az elemek sorszámait is feljegyezzük. Éppen ezért e változat neve: *kiválogatás kigyűjtéssel.*

```
Függvény:
  T:Elemtípus→Logikai
```

```
Változók:
  N:Egész           {a feldolgozandó sorozat elemei száma}
  X:Tömb(1..N:Elemtípus)  {a feldolgozandó sorozat elemei}
  Db:Egész                      {a megfelelő elemek száma}
  Y:Tömb(1..N:Egész)        {a megfelelő elemek sorszámai}

Kiválogatás(N,X,Db,Y):
  Db:=0
  Ciklus i=1-től N-ig
    Ha T(X[i]) akkor Db:=Db+1; Y[Db]:=i
  Ciklus vége
Eljárás vége.
```

Minimális változtatásra lenne szükség, ha nem a sorszámokat, hanem magukat az elemeket kellene kigyűjteni.

```
Kiválogatás(N,X,Db,Y):
  Db:=0
  Ciklus i=1-től N-ig
    Ha T(X[i]) akkor Db:=Db+1; Y[Db]:=X[i]
  Ciklus vége
Eljárás vége.
```

A második változat, a *kiválogatás kiírással* még ennél is egyszerűbb, ebben nincs szükség számolásra, a megtalált elemet azonnal kiírhatjuk. Ez természetesen csak akkor alkalmazható, ha a kiválogatott elemekre további feldolgozás miatt nincs szükség.

```
Kiválogatás(N,X):
  Ciklus i=1-től N-ig
    Ha T(X[i]) akkor Ki: X[i]
  Ciklus vége
Eljárás vége.
```

A harmadik változatban az elemeket gyűjtjük ki, de – feltételezve, hogy az eredeti sorozatra már nincs szükség – ezeket az eredeti vektorban hagyjuk – ez lesz a *kiválogatás helyben*.

```
Kiválogatás(N,X,Db,Y):
  Db:=0
  Ciklus i=1-től N-ig
    Ha T(X[i]) akkor Db:=Db+1; X[Db]:=X[i]
  Ciklus vége
Eljárás vége.
```

Az utolsó változatban sem lesz szükség a kihagyott elemekre, de a megmaradtakat sem akarjuk mozgatni. Emiatt a felesleges elemek helyére egy speciális értéket teszünk, ezzel jelölve a kihagyásukat. Természetesen e megoldás akkor célszerű, ha a későbbi feldolgozások során e speciális érték vizsgálata jóval egyszerűbb, mint maga a kiválogatás volt. E módszer neve: *kiválogatás kihúzással*. Ennél a változatnál nincs szükség a `Db` változóra.

```
Kiválogatás(N,X,Y):
   Ciklus i=1-től N-ig
    Ha nem T(X[i]) akkor X[i]:=speciális érték
   Ciklus vége
Eljárás vége.
```

Nézzük meg egy feladat megoldását!

F5. Jeles átlagú tanulók neve

Példa

```
elemek száma: N
tanulók     : X (N db elem, Rekord(név,átlag))
jelesszám   : Db
jelesek     : Név (maximum N db elem)

Kiválogatás(N,X,Db,Név):
  Db:=0
  Ciklus i=1-től N-ig
    Ha X[i].átlag≥4 akkor DB:=DB+1; Név[Db]:=X[i].név
  Ciklus vége
Eljárás vége.
```

Minta kódok

C++	cpp/5_Kivalogatas/feladat.cpp
C#	cs/5_Kivalogatas/feladat.cs
Java	java/5_Kivalogatas/feladat.java
Pascal	pas/5_Kivalogatas/feladat.pas
Python	py/5_Kivalogatas/feladat.py

Szétválogatás

Feltehető a kiválogatás feladattípussal kapcsolatban az a kérdés, hogy mi lesz a ki nem válogatott elemekkel. E fejezetben erre a kérdésre adunk választ.

F6. Adott N db természetes szám, válogassuk szét a párosakat és a páratlanokat!

F7. Az osztály tanulóit névsoruk alapján válogassuk szét lányokra, illetve fiúkra!

F8. Az osztály tanulói félévi átlageredményük alapján válogassuk szét jelesekre, jókra, közepesekre, elégségesekre, valamint elégtelenekre!

Ez egy új feladattípus osztály első tagja: itt **egy sorozathoz több sorozatot kell rendelni**.

E feladatok mindegyike megfogalmazható úgy, hogy végezzünk el egymás után valahány kiválogatást. Az első két feladatban pontosan kettőről van szó, az utolsóban többről, de ez utóbbi megoldható úgy, hogy először válogassuk szét jelesekre és a többiekre, majd a többieket jókra és ... Ezek alapján megállapítható, hogy a szétválogatást elég megfogalmazni két sorozatra, s ha többről van szó, akkor egyszerűen csak többször kell alkalmazni.

A kétfelé szétválogatást azonban felesleges két kiválogatással megoldani, ugyanis egyértelmű, hogy amit az első menetben nem válogattunk ki, pontosan azok fognak szerepelni a másodikban.

Az első változatban a kétféle elemet két vektorba válogatjuk szét, ezek elemszáma sajnos mindkét esetben az eredeti vektor elemszáma kell legyen, ugyanis elképzelhető, hogy az összes elemet az egyik vektorba kell majd elhelyezni.

Nézzük tehát a szétválogatást két tömbbe!

```
Függvény:
  T:Elemtípus→Logikai

Változók:

  N:Egész            {a feldolgozandó sorozat elemei száma}
  X:Tömb(1..N:Elemtípus)   {a feldolgozandó sorozat elemei}
  DbY,DbZ:Egész                  {a megfelelő elemek száma}
  Y,Z:Tömb(1..N:Egész)       {a megfelelő elemek sorszámai}

Szétválogatás(N,X,DbY,Y,Z,DbZ):
  DbY:=0; DbZ:=0
  Ciklus i=1-től N-ig
    Ha T(X[i]) akkor DbY:=DbY+1; Y[DbY]:=X[i]
            különben DbZ:=DbZ+1; Z[DbZ]:=X[i]
  Ciklus vége
Eljárás vége.
```

Világos, hogy a két vektor alkalmazásával sok felesleges helyet használunk, hiszen `2*N` helyet foglalunk, pedig összesen csak `N`-re van szükségünk.

Ha az elemeket egyetlen tömbbe helyezzük el, mégpedig úgy, hogy a `T` tulajdonságúakat a tömb elejétől, a nem `T` tulajdonságúakat pedig a végétől kezdve helyezzük el, akkor `N` helyet megtakaríthatunk. Ez lesz a *szétválogatás egy tömbbe*.

```
Szétválogatás(N,X,Db,Y):
  Db:=0
  Ciklus i=1-től N-ig
    Ha T(X[i]) akkor Db:=Db+1; Y[Db]:=X[i]
            különben Y[N+1-i+Db]:=X[i]
  Ciklus vége
Eljárás vége.
```

Vegyük észre, hogy ebben a megoldásban a nem `T` tulajdonságú elemek sorrendje az eredetihez képest éppen megfordul.

A kiválogatáshoz hasonlóan itt is megoldhatjuk a *helyben szétválogatás*t is, ha nincs szükség az elemek eredeti sorrendjére.

Egyszerű lenne a következő algoritmus: Elindulunk a tömbben elölről és hátulról, s keresünk olyan elemeket, amelyeket fel kell cserélni. Ha találunk, akkor cserélünk, majd folytatjuk a keresést. Mi ennél egy kicsit hatékonyabb változatot fogunk vizsgálni.

A megoldást úgy végezzük el, hogy a tömb első elemét kivesszük a helyéről. Az utolsó elemtől visszafelé keresünk egy olyat, amely `T` tulajdonságú, s ezt előre hozzuk a kivett elem helyére. Ezután a hátul felszabadult helyre elölről keresünk egy nem `T` tulajdonságú elemet, s ha találtunk azt hátra tesszük. Mindezt addig végezzük, amíg a tömbben két irányban haladva össze nem találkozunk.

```
Szétválogatás(N,X,Db):
  e:=1; u:=N; segéd:=X[e]
  Ciklus amíg e<u
    Ciklus amíg e<u és nem T(X[u])
      u:=u-1                    {T tul. elem keresése}
    Ciklus vége
    Ha e<u akkor X[e]:=X[u]; e:=e+1
                 Ciklus amíg e<u és T(X[e])
                   e:=e+1       {nem T tul. elem keresése}
                 Ciklus vége
                 Ha e<u akkor X[u]:=X[e]; u:=u-1
    elágazás vége
  Ciklus vége
  X[e]:=segéd
  Ha T(X[e]) akkor Db:=e különben Db:=e-1
eljárás vége.
```

Nézzük meg az egyik feladat megoldását, a szétválogatott elemeket egy tömbben elhelyezve elölről, illetve hátulról!

F6. Párosak-páratlanok szétválogatása

Példa

```
elemek száma: N
számok      : X (N db elem)
páros-szám  : Db
eredmény    : Y (N db elem)
T(i)        : Páros(i)

Szétválogatás(N,X,Db,Y):
  Db:=0
  Ciklus i=1-től N-ig
    Ha Páros(X[i]) akkor Db:=Db+1; Y[Db]:=X[i]
                   különben Y[N+1-i+Db]:=X[i]
  Ciklus vége
Eljárás vége.
```

Minta kódok

C++	cpp/6_Szetvalogatas/feladat.cpp
C#	cs/6_Szetvalogatas/feladat.cs
Java	java/6_Szetvalogatas/feladat.java
Pascal	pas/6_Szetvalogatas/feladat.pas
Python	py/6_Szetvalogatas/feladat.py

Feladatok programozási tételekre
a Nemes Tihamér OITV-ről és az Informatika OKTV-ről

1. feladat

Egy kieséses versenyben ismerjük a csapatok mérkőzéseit: ki kit győzött le. Írj programot, amely megadja:

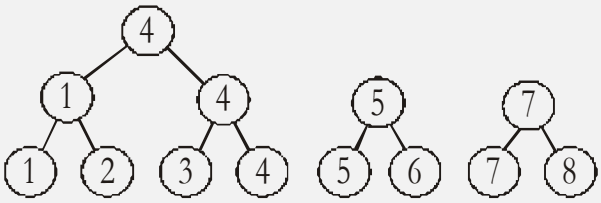
A. a még versenyben levőket;

B. azokat a csapatokat, amelyek legalább egyszer győztek, de már kiestek!

Példa

```
Legyen 8 csapat, 5 mérkőzés!
győztes vesztes
   8       5
   1       2
   4       3
   4       1
   7       8
   5       6

Afeladat: 3 csapat - 4 5 7
Bfeladat: 1 csapat - 1
```



Első lépésként minden csapatra leszámoljuk, hogy hány győzelme és hány veresége van. Bár vereségből egy kieséses versenyen maximum egy lehet, de mégis egyszerűbb megszámolásként kezelni.

Ezután mindkét részfeladat egy kiválogatás. Az A részfeladat megoldása azok kiválogatása, akikre a vereségek száma 0, a B részfeladaté pedig azok kiválogatása, akik győzelmei és veresége száma sem nulla.

Legyen ismert `N` csapat, `M` mérkőzés eredménye!

```
Számolás(N,M,Gy,V):
  Gy:=(0,…,0); V:=(0,…,0)
  Ciklus i=1-től M-ig
    Be: A,B            {A legyőzte B-t}
    Gy[A]:=Gy[A]+1
    V[B]:=V[B]+1  {lehetne V[B]:=1 is}
  Ciklus vége
Eljárás vége.

Afeladat(N,V,Adb,A):
```

```
  Adb:=0
  Ciklus i=1-től N-ig
    Ha V[i]=0 akkor Adb:=Adb+1; A[Adb]:=i
  Ciklus vége
Eljárás vége.

Bfeladat(N,Gy,V,Bdb,B):
  Bdb:=0
  Ciklus i=1-től N-ig
    Ha Gy[i]*V[i]>0 akkor Bdb:=Bdb+1; B[Bdb]:=i
  Ciklus vége
Eljárás vége.
```

A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	https://mester.inf.elte.hu/
Szint	Haladó
Téma	Rekurzív adatszerkezetek
Feladat	29. Kieséses csapatverseny

2. feladat

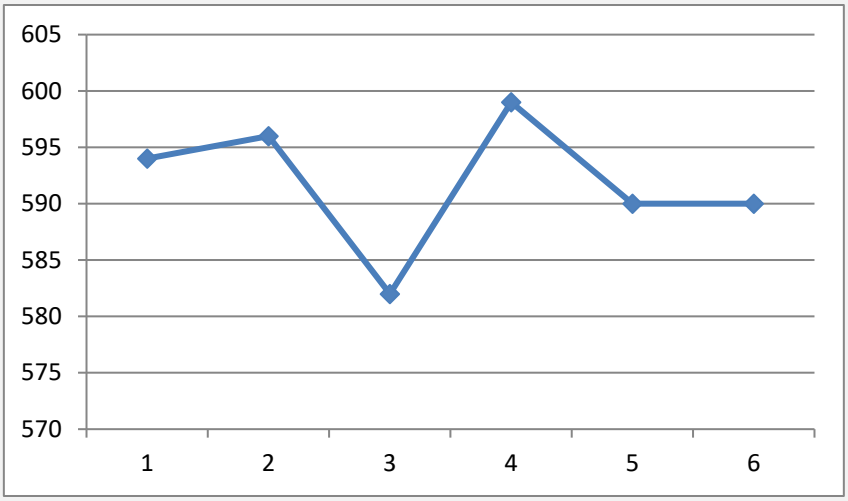
Egy lövészversenyen a versenyzők egymás után lőnek, ismerjük az eredményeiket a szereplésük sorrendjében. Készíts programot, amely beolvassa a versenyzők számát (1≤N≤100), majd az N db eredményt, majd megadja az alábbiakat:

A. Azokat a versenyzőket, akik a verseny valamelyik időszakában álltak az első helyen!

B. Azokat a versenyzőket, akik a verseny valamelyik időszakában álltak az utolsó helyen!

Példa:

```
Bemenet: N=6,
         eredmények= 594, 596, 582, 599, 590, 590
Kimenet: Elsők= 1 2 4
         Utolsók= 1 3
```



Ez két kiválogatási feladat. Az A részfeladatban azokat kell kiválogatni, akik megegyeznek az odáig szereplő elemek maximumával, a B részfeladatban pedig azokat, akik megegyeznek az odáig szereplő elemek minimumával. Azaz a kiválogatás feltételében egy-egy maximumkiválasztás szerepel.

```
Afeladat(N,T,Adb,A):
  ma:=1; A[1]:=1
  Ciklus i=2-től N-ig
    Ha T[i]≥T[ma] akkor Adb:=Adb+1; A[Adb]:=i
  Ciklus vége
Eljárás vége.

Bfeladat(N,T,Bdb,B):
  mi:=1; B[1]:=1
  Ciklus i=2-től N-ig
    Ha T[i]≤T[mi] akkor Bdb:=Bdb+1; B[Bdb]:=i
  Ciklus vége
Eljárás vége.
```

A feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	https://mester.inf.elte.hu/
Szint	
Téma	
Feladat	

3. feladat

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 ($1 \le N \le 100$) és az egyes napokon mért szennyezettséget ($0 \le$ szennyezettség ≤ 500), majd megadja

A. a legszennyezettebb napot;

B. a mérés közbeni javítások számát és a javítási napokat;

C. azon napokat, amikor szmogriadót kellett elrendelni!

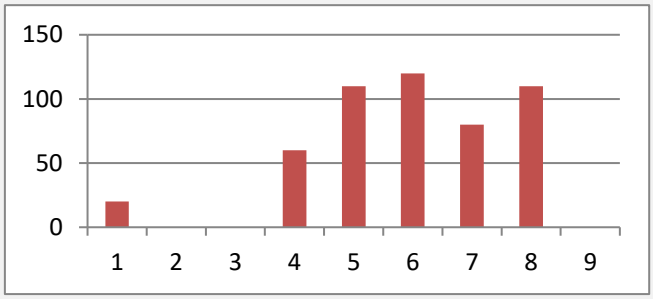
Példa:

```
Bemenet

Napok száma?9
1. nap?20
2. nap?0
3. nap?0
4. nap?60
5. nap?110
6. nap?120
7. nap?80
8. nap?110
9. nap?0
```

```
Kimenet

Legszennyezettebb nap: 6
Javítások száma: 1
Javítások napjai: 4
Szmogriadók száma: 2
Szmogriadók napjai: 5 8
```



Az első részfeladat nem összetett programozási tétel, a második és a harmadik pedig egy-egy kiválogatás. A B részfeladatban azon napokat kell kiválogatni, ahol az aktuális érték nagyobb nullánál, az előző pedig 0 volt; a C részfeladatban pedig azon napokat, amikor az aktuális érték nagyobb 100-nál, az előző pedig nem nagyobb.

```
Bfeladat(N,T,Bdb,B):
  Bdb:=0
  Ciklus i=2-től N-ig
    Ha T[i]>0 és T[i-1]=0 akkor Bdb:=Bdb+1; B[Bdb]:=i
  Ciklus vége
Eljárás vége.
```

```
Cfeladat(N,T,Cdb,C):
  Cdb:=0
  Ciklus i=2-től N-ig
    Ha T[i]>100 és T[i-1]≤100 akkor Cdb:=Cdb+1; C[Cdb]:=i
  Ciklus vége
Eljárás vége.
```

Az A. feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	https://mester.inf.elte.hu/
Szint	Kezdő
Téma	Programozási tételek: minimum, maximum számítás
Feladat	24. Legszennyezettebb nap

A B. feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	https://mester.inf.elte.hu/
Szint	Kezdő
Téma	Programozási tételek: megszámolás
Feladat	23. Légszennyezettségmérő javításainak száma

A C. feladat megoldása tesztelhető az elkészült forráskód feltöltésével itt:

Weboldal	https://mester.inf.elte.hu/
Szint	Kezdő
Téma	Programozási tételek – Kiválogatás
Feladat	8. Elrendelt szmogriadók *