Kedves Versenyzők!

Javasoljuk minden versenyzőnek, hogy gyakoroljon az online értékelővel a Mester-en, címe:

https://mester.inf.elte.hu. Google azonosítóval kell regisztrálni.

Újonnan indul az idei tanévben egy online programozási verseny is:

<http://tehetseg.inf.elte.hu/nemes/nemes_aktualis.html>.

A versenyre nevezni a verseny honlapján lehet, a szükséges adatok (név, település, iskola, évfolyam, e-mail) megadásával a https://biro.inf.elte.hu/faces/versenyre.xhtml oldalon. A megadott email címre küldjük az azonosítót és a jelszót.

1. feladat: Favágás (20 pont)

Egy fasorba N fát ültettek balról jobbra, egy vonalba. Mindegyik fának ismerjük a bal szélső fáról vett távolságát és a magasságát. Ha egy fát kivágunk, akkor az a jobboldali szomszédja felé dől, s amelyik szomszédjára rádől, az is kidől. Az 1 távolságra levő fára az 1 magasságú fa nem dől még rá, a 2 magasságú viszont igen.

Az alábbi fák esetén add meg, hogy mely fákat kell kivágni, hogy az összes fa kidőljön! A számpárok első tagja a fa távolsága a bal szélső fától, a második pedig a magassága.

A. (0,6), (3,1), (5,2), (8,1), (15,10)

B. (0,3), (2,3), (4,1), (6,3), (8,2), (10,1)

C. (0,4), (3,4), (4,3), (6,1), (7,5), (9,2), (11,4), (12,2), (14,1), (15,1), (16,5)

2. feladat: Keresőfa (35 pont)

Egy keresőfára igaz, hogy bármely elemét megnézve, tőle balra csak nála kisebb, jobbra pedig csak nála nagyobb elemek vannak. A baloldali ábrán látható keresőfát a jobboldali, 4 mezőt tartalmazó T tömbben ábrázoljuk.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | |  |  |  |  | | --- | --- | --- | --- | | érték | bal | jobb | szülő | | 30 | 2 | 3 | 0 | | 20 | 4 | 5 | 1 | | 40 | 6 | 7 | 1 | | 10 | 0 | 0 | 2 | | 25 | 8 | 9 | 2 | | 35 | 0 | 0 | 3 | | 45 | 10 | 0 | 3 | | 22 | 0 | 0 | 5 | | 28 | 0 | 0 | 5 | | 42 | 0 | 0 | 7 | |

Az alábbi algoritmusoknak i és j a bemenő, p és v a kimenő paramétere:

Egyik(i,p):  
 p:=i  
 Ciklus amíg T(p).bal>0  
 p:=T(p).bal  
 Ciklus vége  
Eljárás vége.

Másik(j,p,v):  
 p:=1; v:=0  
 Ciklus amíg v=0 és T(p).érték≠j  
 Ha j<T(p).érték akkor  
 Ha T(p).bal>0 akkor p:=T(p).bal különben v:=1  
 különben  
 Ha T(p).jobb>0 akkor p:=T(p).jobb különben v:=2  
 Ciklus vége  
Eljárás vége.

Harmadik(i,p):  
 p:=i  
 Ha T(p).jobb>0 akkor Egyik(T(p).jobb,p)  
 különben q:=T(p).szülő  
 Ciklus amíg q≠0 és p=T(q).jobb  
 p:=q; q:=T(p).szülő  
 Ciklus vége  
 p:=q  
Eljárás vége.

Mi az eredménye a fenti keresőfára az alábbi eljáráshívásoknak?

A. Egyik(1,p)

B. Egyik(3,p)

C. Másik(35,p,v)

D. Másik(34,p,v)

E. Másik(46,p,v)

F. Harmadik(2,p)

G. Harmadik(9,p)

3. feladat: Városok (20 pont)

Ismerjük egy megye települései közötti utakat (a két település sorszámával, amelyeket összekötnek). Tudjuk, hogy el lehet jutni bármely településről bármely településre. Egyes utakat felújítás idejére lezárnak. Add meg, hogy melyek azok az utak, amelyek közül bármelyiket lezárva, nem lehet eljutni bármely településről bármely településre!

A. (5,2), (1,3), (1,4), (2,3), (5,6), (2,6), (3,4)

B. (9,5), (6,5), (10,6), (10,9), (3,2), (3,10), (8,3), (1,8), (7,8), (4,1), (4,7), (7,1)

4. feladat: Hálózat (40 pont)

Egy hálózaton üzeneteket szeretnénk küldeni. Minden üzenetet egy n hosszú sorozatként kódolunk, ahol a sorozat elemei az 1, …, q számok. (Pl.: n=3, q = 4-re [2, 1, 4] vagy [1, 1, 1] üzenetek). A hálózatunk nem megbízható, néha az átvitel során megváltozik a sorozat néhány tagja. Legyen két üzenet távolsága azon pozíciók száma, ahol a két sorozat különbözik. Jelöljük u és v távolságát d(u, v)-vel. (Pl.: d([2, 1, 4], [1, 1, 1]) = 2, mert az első és a harmadik pozícióban tér el a két üzenet.) A hibák javítása érdekében nem használjuk az összes lehetséges üzenetet csak azok egy C részhalmazát, ezt nevezzük kódnak. Ha egy kódban nem található sorozatot fogadunk, azt lecseréljük a hozzá legközelebbi kódbeli sorozatra, ezzel megkísérelve a hiba javítását. Egy C kód távolsága a  érték. Egy kód t-hibafelismerő, hogy ha legfeljebb t hiba bekövetkezése esetén felismerjük, hogy az átvitel során hiba történt. Egy kód pontosan t-hibafelismerő, ha t-hibafelismerő, de nem (t+1)-hibafelismerő. Hasonlóan egy kód t-hibajavító, ha legfeljebb t hiba bekövetkezése esetén a sorozatot helyesen javítani tudjuk. Egy kód pontosan t-hibajavító, ha t-hibajavító, de nem (t+1)-hibajavító.

Legyen n=5, q=4, C={[4,2,3,4,1],[1,4,3,4,3],[1,3,1,1,2],[2,2,2,3,4]}!

A. d([4, 2, 3, 4, 1], [1, 4, 3, 4, 3])=?

B. d(C) = ?

C. Pontosan hány hibafelismerő C?

D. Pontosan hány hibajavító C?

E. Mennyi egy T kód d(T) távolsága, ha az pontosan t-hibafelismerő?

F. Mennyi lehet egy T kód d(T) távolsága, ha az pontosan t-hibajavító?

G. Legfeljebb hány elemű lehet egy T kód, ha n = 5, q = 3 és T 2-hibajavító? Adj meg egy ilyen kódot úgy, hogy az egyik sorozat a kódban a [2, 3, 1, 2, 3] legyen.

5. feladat: Mit csinál? (35 pont)

Az alábbi algoritmus az N elemű X vektor értékei alapján állít elő egy Y vektort. Az X vektor elemei 1 és M közötti egész számok.

Valami(N,M,X,Y)  
 D[1..M]:=0  
 Ciklus i=1-től N-ig  
 D[X[i]]:=D[X[i]]+1  
 Ciklus vége {\*}  
 E[1]:=1  
 Ciklus i=1-től M-1-ig  
 E[i+1]:=E[i]+D[i]  
 Ciklus vége {\*\*}  
 Ciklus i=1-től N-ig  
 Y[E[X[i]]]:=X[i]  
 E[X[i]]:=E[X[i]]+1  
 Ciklus vége {\*\*\*}  
Eljárás vége.

A. Mi lesz a D vektorban {\*}-nál, az E vektorban {\*\*}-nál és az Y vektorban {\*\*\*}-nál, ha N=10, M=3, X=(2,3,1,3,2,3,1,1,3,3)?

B. Mi lesz a D vektorban {\*}-nál, az E vektorban {\*\*}-nál és az Y vektorban {\*\*\*}-nál, ha N=10, M=10, X=(2,4,1,3,2,3,5,7,3,3)?

C. Fogalmazd meg általánosan, hogy mi lesz a D vektorban {\*}-nál, az E vektorban {\*\*}-nál és az Y vektorban {\*\*\*}-nál!

6. feladat: Dobozok (24 pont)

Egy üzemben kocka alakú dobozokat gyártanak. Egy doboz akkor tehető bele egy másik dobozba, ha a mérete (a kocka oldalhossza) kisebb. A dobozokat egy teherautóra pakolhatják, de egymás mellé két doboz nem fér el, azaz csak egymásba rakva vihetők el. A dobozok egyesével érkeznek, és a teherautóra csak érkezési sorrendben pakolhatók, azaz minden doboznál azt dönthetjük el, hogy beletesszük a teherautón levő dobozokból a legbelső belsejébe vagy nem tesszük fel a teherautóra.

A dobozok mérete az érkezés sorrendjében: 5,3,6,4,8,7,3,3,1,2.

Töltsd ki az alábbi táblázatot, amelyben az i-edik szám azt jelöli, hogy maximum hány doboz lehet a teherautón, ha a legutoljára felrakott doboz az i-edik, valamint, hogy közvetlenül milyen sorszámúba rakhattuk (utóbbi szám 0, ha nem raktuk bele másikba)!

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| felső | 1. | 2. | 3. | 4. | 5. | 6. | 7. | 8. | 9. | 10. |
| max. | 1 |  |  |  |  |  |  |  |  |  |
| alatta | 0 |  |  |  |  |  |  |  |  |  |

Példa:

Ha három dobozunk lenne, rendre 2,3,1 mérettel, akkor a táblázat második sorában az 1,1,2 számok állnának, a harmadik sorban pedig a 0,0,1 vagy 0,0,2 (az 1 méretű doboz az előző kettő bármelyikébe beletehető, az elsőbe a második nem).

7. feladat: A Rek bolygó programozói (26 pont)

Nemrég rendezték a programozók intergalaktikus találkozóját, ahol a földi programozók találkoztak a Rek bolygóról érkező kollégáikkal. A földiek meg akarták mutatni a kedvenc programjaikat, de kiderült, hogy a Rek bolygón nem ismerik a ciklusokat és az értékadást. Nincs is rá szükségük, mert mindent rekurzióval valósítanak meg (vagyis olyan függvényeket írnak, amik meghívják saját magukat). A függvények értéke náluk egyszerűen az utoljára kiszámolt kifejezés értéke lesz.

Egy programot már sikerült átírnia a tolmácsnak úgy, hogy az idegenek is értsék, de a többiben néhány helyen bizonytalan. Írd be a hiányzó kifejezéseket! Szerencsére már csak a függvény paramétereket, számokat és műveleteket kell használnod. A programok paraméterei mind pozitív egészek lehetnek.

|  |  |
| --- | --- |
| Földi program | Rek program |
| faktoriális(N):  a ≔ 1  Ciklus i=1-től N-ig  a ≔ a \* i  Ciklus vége  faktoriális ≔ a Függvény vége. | faktoriális(N):  ha N = 1  1  különben  N \* faktoriális(N-1) Függvény vége. |

Írd át az alábbi algoritmusokat a Rek bolygó nyelvére az alábbi kódrészek kiegészítésével!

|  |  |
| --- | --- |
| valami(F):  a ≔ 1  b ≔ 1  Ciklus i=3-tól F-ig  a ≔ a + b  b ≔ a – b  Ciklus vége  valami ≔ a Függvény vége. | valami(F):  ha < 3  különben  valami()+valami() Függvény vége. |
| valami(A,B,C):  s ≔ 0  Ciklus i=A-tól B-ig  ha i mod C = 0  s ≔ s + 1  Ciklus vége  valami ≔ s Függvény vége. | valami(A,B,C):  ha >  különben ha A mod C = 0  +valami(,,)  különben  valami(,,) Függvény vége. |
| valami(A):  c ≔ 0  b ≔ 1  Ciklus amíg b\*b ≤ A  c ≔ c + b \* b  b ≔ b + 1  Ciklus vége  valami ≔ c Függvény vége. | valami(A):  valami2(,1,0) Függvény vége.  valami2(A,B,C):  ha >  különben  valami2(,,) Függvény vége. |

Elérhető összpontszám: 200 pont