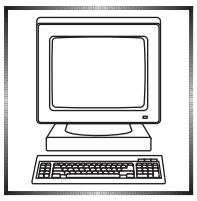


Informatikából kitűzött feladatok

I. 490. Az udvaron N diák ($5 \leqslant N \leqslant 35$) kieséses játékot játszik. A játék elején mindenki választ magának egy pozitív egész számot (mindenki különbözőt). A játék körökből áll, a köröket játék közben számolják, a játék a legelső körrel indul. Egy-egy kör végére néhány játékos kieshet, így ők a következő körökben már nem játszanak. A játék akkor ér véget, amikor két egymást követő körben nem esik ki egyetlen játékos sem. Ekkor a bennmaradók a játék győztesei.

A játék egy-egy körében a következőket teszik a játékosok:

1. számuk szerint növekvő sorrendbe állnak külön-külön a páros és a páratlan számmal rendelkezők;
2. a két sort összefésülik úgy, hogy egy új sor keletkezzen:
   a) a páratlan sorszámú körökben az első (legkisebb) páratlan számú diák kerül az új sor elejére (ha van ilyen diák);
   b) a páros sorszámú körökben az első (legkisebb) páros számú diák kerül az új sor elejére (ha van ilyen játékos);
   c) a többiek felváltva csatlakoznak az egyik és a másik sorból, amíg mindkét sor elég hosszú;
   d) majd a hosszabb sorból jönnek egymás után a megmaradt játékosok;
3. ezután minden olyan játékos kiesik, akinek a száma ebben az új sorban kisebb a mellette álló mindkét játékos számánál (a sorban most első és utolsó játékos tehát nem eshet ki).

Készítsünk programot, amely megadja, hogy egy adott játék hányadik körben ér véget, és kik a győztesei.

A standard bemenet első sorában a játszók N száma, második sorában a játékosok által választott N darab szám szerepel. A standard kimenet első sorába írjuk ki a körök számát, második sorába a győztes versenyzők számát növekvő sorrendben.

Példa:

Bemenet	**Kimenet**
`8` `2 32 5 10 18 9 7 11`	`4` `10 18 32`

Beküldendő egy `i490.zip` tömörített állományban a program forráskódja és egy rövid leírás, ami megadja, hogy a forrásállomány melyik fejlesztői környezetben fordítható.

I. 491. A Monor városi sportcsarnok egy multifunkciós csarnok, mely sport- és rendezvényközpontként működik. Az épület hasznos alapterülete 4166 m^2, az ülőhe-

lyek száma 1030, de nagy rendezvény esetén az épület maximális befogadóképessége (a küzdőtér használatával) 1500 fő.

A `monor.txt` pontosvesszővel tagolt, UTF-8 kódolású állományban a sportcsarnok látogatottsági statisztikája áll rendelkezésre 2016. szeptember 18. óta. Az A oszlopban az évek, a B oszlopban a hozzá kapcsolódó hónapok, a második sorban pedig a napok kaptak helyet.

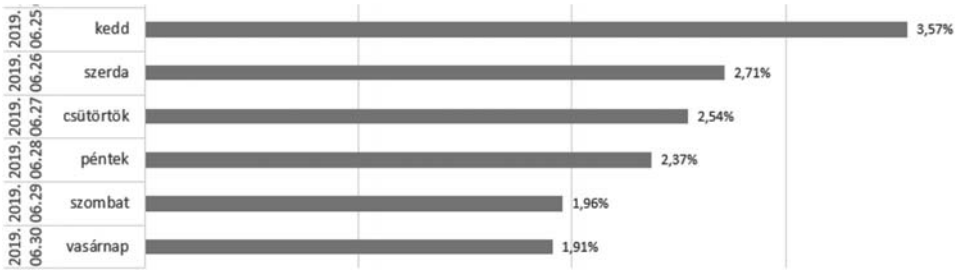
	A	B	C	D	E	F	G	H
1								
2			1	2	3	4	5	6
3	2016	szeptember						
4	2016	október	4	6	11	12	21	18
5	2016	november	49	80	34	27	34	63
6	2016	december	59	18	14	98	30	27
7	2017	január	49	39	63	51	91	53

1. Töltsük be a `monor.txt` szövegfájlt a táblázatkezelő egy munkalapjára az A1-es cellától kezdődően.
2. Munkánkat `monor` néven mentsük el a táblázatkezelő alapértelmezett formátumában. Az üres cellák azt jelzik, hogy nincsen adat. A táblázatot úgy készítsük el, hogy 2020. januárig feltölthető legyen, amint rendelkezésre állnak az adatok. A képletek kialakításánál és a számításoknál is készüljünk föl ezekre az adatokra.
3. Töltsük fel a B oszlopot a hónapnevekkel, a második sort a napok számaival.
4. Függvény segítségével töltsük fel az A oszlopot a megfelelő évekkel a 43. sorig.
5. Az AH3:AH43 tartomány celláiban számítsuk ki, mennyi volt az adott hónapban a látogatók száma.
6. Az AL3:AL7 tartományban függvény segítségével adjuk meg az összes látogatási adat figyelembevételével, hogy mennyi volt a látogatók összes száma az alábbi létszámsávokban.

0 – 49
50 – 199
200 – 599
600 – 999
1000 –

7. Az AK8-as cellában adjuk meg függvény segítségével, hogy mennyi volt az egy napon belüli legnagyobb látogatószám a sportcsarnokban.
8. Feltételes formázás segítségével jelöljük a három legnagyobb látogatószámot tartalmazó cellát piros kitöltő színnel. Ha több egyforma érték is van, akkor mindegyiket jelöljük meg.

9. Készítsünk a táblázat alá a mintán látható diagramhoz hasonlót a 2019. júniusi adatokból (a júniusi teljes hónapról – a mintán kevesebb, mint egy hét látszik). A százalékos értékek azt mutatják, hogy az adott napon a júniusi összesített látogatói adathoz képest az emberek hány százaléka látogatott el a sportcsarnokba az adott napon. Mivel már végleges adatokról van szó, így nem szükséges a diagramnak az adatok változását követnie.



10. A fő táblázatot a könnyebb átláthatóság érdekében állítsuk be úgy, hogy az év, a hónap és a napok mindig láthatóak maradjanak görgetéskor. Nyomtatáskor a fő adatok egy oldalra férjenek el.

Források:
`www.monorisportcsarnok.hu` (utolsó letöltés 2019.09.10.);
`http://monorisportcsarnok.hu/stat/statistic.php?ev=2016&ho=9&l=m` (utolsó letöltés 2019.09.10.).

Beküldendő egy tömörített `i491.zip` állományban a munkafüzet, valamint egy rövid leírás, amelyben szerepel az alkalmazott táblázatkezelő neve és verziószáma.

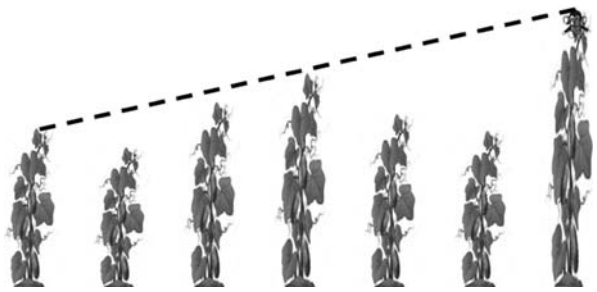
I. 492 (É). Ubullal, az Uborkanemesítő Intézet kismajmával egy korábbi feladatban (I. 398.) már találkoztunk. Azóta Ubul átköltözött az Intézet egyik kísérleti parcellájába, ahol N sorban és M oszlopban ($1 \leqslant N, M \leqslant 100$) ültetve helyezkednek el az uborkafák.

Az uborkafák magasságát a weblapunkról letölthető `ubifak.txt` nevű, UTF-8 kódolású, tabulátorokkal tagolt szöveges állomány centiméterben megadva tartalmazza. Egy uborkafa nagyon magas, de legfeljebb 100 méteres lehet. Az állomány első eleme az $(1; 1)$, utolsó eleme pedig az $(N; M)$ koordinátájú fa magasságát adja meg.

Készítsünk programot `i492` néven a következő feladatok megoldására. A program futása során a képernyőre való kiíráskor utaljunk a feladat sorszámára.

1. Olvassuk be a fájlból az uborkafák magasságát, és az adatokat tároljuk el.
2. Kérjük be egy fa koordinátáit (sorszám, oszlopszám) és írassuk ki a képernyőre az adott koordinátájú uborkafa magasságát.
3. Ubul az előző feladatban megadott fán ücsörög. Hány olyan fa van a parcellában, amely az előbb megadott koordinátájú fánál magasabb?

4. Szemléltessük a kilátást a `kilatas.txt` nevű állománnyal, amely N sorban és M oszlopban karaktereket tartalmaz (szóközök nélkül) a következő módon. Ubul előbb megadott helyét egy U betű jelöli. Az adott pontban lévő fánál magasabb fákat × jelöli, míg a többi fát egy-egy pont.
5. Ubul napközben legszívesebben a parcella legmagasabb fáján szeret ücsörögni. Hol van ez a fa és milyen magas? Írassuk ki a választ a képernyőre. Ha több ilyen van, mindegyik koordinátái jelenjenek meg.
6. Az éjszakát Ubul azon a fán töltötte, amely a sorok legnagyobb fái közül a legkisebb, hogy ne fázzon. Reggel át akar ugrálni arra a fára, amely az oszlopok legkisebb fái közül a legnagyobb. Ha Ubul mindig csak az adott sor vagy adott oszlop szomszédos fájára ugrik, legalább hány ugrással közelítheti meg ezt a fát? (Feltehetjük, hogy a két szélsőérték egyértelmű.)



7. Látja-e Ubul a megadott koordinátájú fáról az adott sorban, illetve az adott oszlopban lévő szélső fák tetejét? Mind a négy eset eredményét írassuk ki a képernyőre. (Feltételezhetjük, hogy a fák egyenlő távolságra vannak egymástól.)

Beküldendő egy tömörített `i492.zip` állományban a program forráskódja és rövid dokumentációja, amely tartalmazza a megoldás rövid leírását, és megadja, hogy a forrásállomány melyik fejlesztői környezetben fordítható.

I/S. 38. Egy munkahelyen N ember dolgozik, akiket 0-tól $(N-1)$-ig sorszámokkal azonosítunk. Mindenkinek pontosan egy közvetlen főnöke van, kivéve a 0. sorszámú embert, a cégvezetőt. Minden emberre teljesül, hogy ha vesszük a közvetlen főnökét, majd annak a közvetlen főnökét és így tovább, amíg lehet, akkor végül a cégvezetőhöz jutunk. Egy A sorszámú embernek beosztottja minden olyan ember, ahonnan az előbbi módon, a közvetlen főnökökön végighaladva egy idő után az A sorszámú emberhez jutunk. Egy A sorszámú ember tudja kezelni egy B sorszámú beosztottját, ha a cégnél töltött éveik számának különbsége legfeljebb K. Egy A sorszámú ember jó főnök, ha minden beosztottját tudja kezelni (ha valakinek nincs beosztottja, akkor jó főnök). Készítsünk programot, amely megadja a jó főnökök számát.

Standard bemenet: az első sor tartalmazza N-et és K-t. A következő sor $N-1$ darab számot tartalmaz, az i-edik szám az i-edik sorszámú ember közvetlen fő-

nökének sorszámát. A következő sor N darab számot tartalmaz, az i-edik szám az $(i-1)$-edik sorszámú ember cégnél töltött éveinek számát.

Standard kimenet: a jó főnökök száma.

Korlátok: $3 \leqslant N \leqslant 10^5$, $0 \leqslant K \leqslant 10^9$, $1 \leqslant$ a cégnél eltölött évek száma $\leqslant 10^9$. Időkorlát: 0,3 mp.

Értékelés: a pontok 50%-a kapható, ha $N \leqslant 1000$.

Példa:

Bemenet (a / jel a sortörést helyettesíti)	**Kimenet**
7 3 2 0 2 0 4 4 6 1 5 7 10 7 8	4

Beküldendő egy `is38.zip` tömörített állományban a megfelelően dokumentált és kommentezett forrásprogram, amely tartalmazza a megoldás lépéseit, valamint megadja, hogy a program melyik fejlesztői környezetben futtatható.

S. 137. A Lamök bolygón összeírták egy elektronikus szótárba az univerzum összes szavát ABC-sorrendben. Sajnos a rendszert támadás érte, ezáltal nemcsak a szavak sorrendje, hanem az egyes szavakon belül a betűk sorrendje is összekeveredett. A bolygó lakói szeretnék minél hamarabb visszaállítani az eredeti szótárat, ezért a hibás szótár összes szavára meg akarják határozni, hogy minimum és maximum hányadik lehetett az eredeti szótárban. Sajnos ez túl nehéz feladatnak bizonyult számukra, ezért a te segítségedet kérik: készíts programot, amely megadja, hogy egy-egy szó legalább és legföljebb hányadik lehetett az eredeti szótárban.

A standard bemenet első sora tartalmazza a szótár szavainak N számát. Ezután N sor következik: a bemenet $(i+1)$-edik sora tartalmazza a hibás szótár i-edik szavát. A szavak csak az angol ABC kisbetűit tartalmazzák.

A standard kimenet N sort tartalmaz: az i-edik sorba írjuk ki, hogy a hibás szótár i-edik szava az eredeti szótárban minimum és maximum hányadik lehetett.

Korlátok: $1 \leqslant N \leqslant 10^5$, $1 \leqslant$ egy szó hossza $\leqslant 20$. Időkorlát: 0,3 mp.

Értékelés: a pontok 50%-a kapható, ha $N \leqslant 10^4$.

Példa (a / jel sortörést helyettesít):

Bemenet	**Kimenet**
5 / mlaa / tenebem / osr / lama xyz	1 3 / 1 4 / 3 4 / 1 3 / 5 5

Beküldendő egy `s137.zip` tömörített állományban a megfelelően dokumentált és kommentezett forrásprogram, amely tartalmazza a megoldás lépéseit, valamint megadja, hogy a program melyik fejlesztői környezetben futtatható.

A feladatok megoldásai regisztráció után a következő címen tölthetők fel:
`https://www.komal.hu/munkafuzet`
Beküldési határidő: 2019. november 10.