Megoldások

A feladatsor létrejöttében közreműködtek: Bakonyi Viktória, Busa Máté, Csertán András, Deák Bence, Gáspár Attila, Horváth Gyula, Németh Zsolt, Noszály Áron, Sárközi Gergely, Szente Péter, Zsakó László

17

Ötlet: Zsakó László

Kidolgozó: Zsakó László

Témák: implementáció (elágazás, ciklus, tömb)

A feladat erősen implementációs jellegű, a megoldáshoz elegendő a feladatszövegben definiált algoritmus lekódolása. Gyakori buktatók voltak a második üres sor hiánya vagy az algoritmus nem pontos megvalósítása (pl. `N>17`-ig ismételni `N>0` helyett). Érdekességképpen meggondolhatjuk, hogy az algoritmus `O(log(N))` futási idejű, mivel lépésenként legalább `1`-el csökken az `N` tízes számrendszerbeli hossza.

Torony1

Ötlet: Zsakó László

Kidolgozó: Zsakó László

Témák: rekurzív sorozatok

A válasz a kérdésre legyen F(N)! Ekkor:

$$F(N) = \begin{cases} 1 & ha \quad N = 0 \\ 3 & ha \quad N = 1 \\ 3 * F(N-1) + F(N-2) & ha \quad N > 1 \end{cases}$$

Miért pont 3*F(N-1)+F(N-2)? Osztályozzuk az N magas tornyokat! A tetejükön piros, sárga, zöld vagy fehér építőelem van, ezeket levéve N-1, N-1, N-1 vagy N-2 nagyságú tornyokhoz jutunk, amikből összesen F(N-1)+F(N-1)+F(N-1)+F(N-2)=3*F(N-1)+F(N-2) van. Lépésenkénti maradékszámítás kihagyásával a nagy értékű eredmények hibásak lesznek (ha nincs tetszőlegesen nagy szám ábrázolására alkalmas típusunk). Az F értékek kiszámítására nem célszerű nyelvbe épített rekurziót használnunk, mivel naívan (ún. memorizáció nélkül) nagyon lassú, exponenciális futási idejű megoldást kapunk.

Érdemes lehet ciklust használni, két különböző módon is gondolkodhatunk (F(i)-t miből számíthatjuk ki, illetve F(i)-ből mit számíthatunk ki):

```
F[0]=1; F(1):=3
Ciklus i=2-től N-ig
  F[i]=(3*F[i-1]+F[i-2]) mod 20210108
Ciklus vége
```

```
F[0]=1
Ciklus i=0-tól N-1-ig
  F[i+1]=(F[i+1]+3*F[i]) mod 20210108
  F[i+2]=(F[i+2]+F[i]) mod 20210108
Ciklus vége
```

Az F kezdetben mindkét esetben 0-kal van feltöltve, az eredmény mindkettőben F(N), bár a második még F(N+1)-be is számol.

Tehát összességében O(N) időben megoldhatjuk a feladatot. Megjegyzendő, O(log(N)) idejű megoldás is adható, például a rekurzió mátrixának gyorshatványozásával.

Kártya

Ötlet: Busa Máté

Kidolgozó: Busa Máté

Témák: matematika

Az, hogy mindenkinek ugyanannyi kártyája legyen az ajándékozás után, pontosan akkor lehetséges, ha $A_1+A_2+...+A_N$ osztható `N`-nel. Ha nem osztható, akkor tehát rögtön adódik, hogy a válasz `NEM` és még további $N-(A_1+A_2+...+A_N) \bmod N$ kártyát kell szerezniük. Egyébként a válasz `IGEN` és mindegyiküknek $\frac{A_1+A_2+\cdots+A_N}{N}$ kártyája lesz az ajándékozás után, tehát az `i`. gyerek $A_i - \frac{A_1+A_2+\cdots+A_N}{N}$ kártyát ajándékozott el, feltéve, hogy ez pozitív mennyiség.

Utóbbit úgy is kifejezhetjük, hogy az `i`. gyereknek $max\left(0, A_i - \frac{A_1+A_2+\cdots+A_N}{N}\right)$ kártyát kell elajándékoznia!

Unoka

Ötlet: Zsakó László

Kidolgozó: Deák Bence

Témák: fák (levéltől gyökér felé ábrázolva)

Minden személyre tároljuk a közvetlen felmenőjét, legyen ez az `i`-re `apa[i]>0`, ha pedig nincsen felmenője rögzítve, akkor legyen `apa[i]=0`! Legyen továbbá `apa[0]=0`! Legyen az `i`. ember unokáinak száma `unoka[i]`, kezdetben `unoka[i]=0`! Ekkor a megoldás:

```
Ciklus i=1-től N-ig
  unoka[apa[apa[i]]]:=unoka[apa[apa[i]]]+1
Ciklus vége
```

Minden emberre `unoka[i]` éppen az unokáinak számát adja meg, hiszen az `i`. csúcsra `apa[i]` vagy az apja, vagy `0`, ha nem létezik, míg `apa[apa[i]]` a nagyapja, vagy `0`, ha nem létezik. Ezután már csak az `unoka` tömbben kell a maximális értékű elemet és a hozzátartozó indexet megtalálni (a speciális `0`-s indexet kivéve).