Felvételi előkészítő

(Elemi algoritmusok; Tömbök)

1. Határozzuk meg egy adott ***n*** természetes számnál kisebb számok közül azt, amelyiknek a legtöbb valódi osztója van. A valódi osztó nagyobb mint 1, és kisebb, mint maga a szám! Írjuk ki a számot és valódi osztóinak darabszámát.

Példa

Bemenet	Kimenet
1000	840 30

Adott a következő sorozat, amelynek minden elemét – az elsőt kivéve – az előző elem segítségével generáljuk: 1, 11, 21, 1211, 111221, ...

A generálási szabály a következő:

- megszámoljuk, balról jobbra haladva az előző érték számjegyeit
- az új értéket úgy kapjuk meg, hogy beírjuk a régi érték összes számjegyeinek előfordulási számát és az illető számjegyet.

Határozzuk meg az ***n***-edik (***n*** $\leq$ 20) elemét a sorozatnak!

Példa

Bemenet	Kimenet
6	312211

2. Döntsük el egy adott számról, hogy *völgyszám*-e vagy *hegyszám*. Egy *völgyszám* számjegyei csökkenő sorrendben követik egymást egy bizonyos elemmel bezárólag, azután *szigorúan* növekvő sorozatot alkotnak. A *hegyszám* számjegyei *szigorúan* növekvő sorrendben követik egymást egy bizonyos elemmel bezárólag, azután csökkenő sorozatot alkotnak.

Írjunk ki egy megfelelő üzenetet aszerint, hogy az adott szám *völgyszám*-e vagy *hegyszám*. Ha a szám nem völgyszám és nem hegyszám, vágjuk le a szám *első* néhány számjegyét amíg az így kapott szám *völgyszámmá* vagy *hegyszámmá* válik. Írjuk ki az adott szám azon részét, amely völgyszám vagy hegyszám, ha létezik ilyen, vagy megfelelő üzenetet, ha nem.

Példák

Bemenet	Kimenet
13752	hegyszam
85369	volgyszam
913752	13752
1234	nincs benne hegyszam vagy volgyszam

3. Adott a következő sorozat, amelynek minden elemét – az elsőt kivéve – az előző elem segítségével generáljuk: 1, 11, 21, 1211, 111221, ...

A generálási szabály a következő:

- megszámoljuk, balról jobbra haladva az előző érték számjegyeit
- az új értéket úgy kapjuk meg, hogy beírjuk a régi érték összes számjegyeinek előfordulási számát és az illető számjegyet.

Határozzuk meg az $\boldsymbol{n}$-edik ($\boldsymbol{n} \leq 20$) elemét a sorozatnak!

Példa

Bemenet	Kimenet
6	312211

4. Írjunk algoritmust, amely egy adott, 6-nál nagyobb *páros* számot felír két különböző páratlan prímszám összegeként (*Goldbach-sejtés*).

Példák

Bemenet	Kimenet
10 24	10 = 3 + 7 24 = 5 + 19

5. Egy természetes számot *majdnem prím*nek nevezünk, ha egyenlő *két különböző prímszám szorzatával*. Például, a 15 *majdnem prím*, mivel egyenlő a 3 és 5 prímszámok szorzatával. Legyen egy $\boldsymbol{n}$ természetes szám ($1 \leq \boldsymbol{n} \leq 100\,000$) és egy $\boldsymbol{n}$ elemű $\boldsymbol{x}$ sorozat, amelyben az elemek 1-nél szigorúan nagyobb és 30 000-nél kisebb természetes számok.

Írjatok programot, amely meghatározza az adott sorozat leghosszabb tömbszakaszát, amely csak *majdnem prím*eket tartalmaz, és kiírja az illető tömbszakasz kezdőindexét (***balMax***) és végsőindexét (***jobbMax***). Ha több ilyen tömbszakasz létezik, a *legelső* leghosszabb tömbszakasz kezdőindexét és végsőindexét kell kiírnotok. Ha a sorozatban nem létezik egyetlen *majdnem prím* sem, ***balMax*** és ***jobbMax*** értéke egyenlő lesz -1-gyel.

Példák

Bemenet	Kimenet
8 24 34 35 11 8 77 35 26	6 8

($\boldsymbol{x}_6 = 77 = 7 * 11$, $\boldsymbol{x}_7 = 35 = 5 * 7$, $\boldsymbol{x}_8 = 26 = 2 * 13$).

Bemenet	Kimenet
3 24 11 8	-1 -1

6. Ismert, hogy bármely természetes szám ábrázolható az úgynevezett Fibonacci számrendszerben, ahol a számjegyek 1 és 0:

Egy, a Fibonacci számrendszerben felírt szám alakja: $x_{Fib} = c_n c_{n-1} \ldots c_3 c_2 c_1$, ahol c_i **1** vagy **0**.

Az x szám értékét a 10-es számrendszerben a következőképpen számítjuk ki:

$x_{10} = c_n \cdot F_n + \ldots + c_3 \cdot \mathbf{3} + c2 \cdot \mathbf{2} + c_1 \cdot \mathbf{1}$, ahol F_i a Fibonacci sorozat i-dik eleme (az $\mathbf{F_0} = \mathbf{0}$ elemet nem vesszük figyelembe és a sorozat egyetlen **1**-essel kezdődik).

Példa:

$x_{Fib} = 10101001 = 1 \cdot 34 + 0 \cdot 21 + 1 \cdot 13 + 0 \cdot 8 + 1 \cdot 5 + 0 \cdot 3 + 0 \cdot 2 + 1 \cdot 1 = 53_{10}$

Azért, hogy az ábrázolás egyedi legyen, a fenti szabályhoz hozzátesszük még a következő követelményt: az ábrázolás ne tartalmazzon egymás után két darab 1-est.

Írjatok programot, amely megszámolja azokat az adott n ($1 < n \le 1\,000\,000$) számnál kisebb (esetleg n-nel egyenlő) számokat, amelyek felírva Fibonacci számrendszerben palindromszámok.

Példa:

Bemenet	Kimenet	*Magyarázat*
15	6	6 darab 15-nél kisebb, adott tulajdonságú szám létezik: 1, 4, 6, 9, 12, 14, amelyeknek alakja felírva Fibonacci számrendszerben: 1, 101, 1001, 10001, 10101, 100001.

Maximális végrehajtási idő: 0,1 szekundum

7. A ***hrs*** sorozat elemeit a következő szabályok alapján számítjuk ki:

$$\begin{cases} \boldsymbol{r_1} = 1 \\ \boldsymbol{s_1} = 2 \end{cases}$$

$$\begin{cases} \boldsymbol{r_2} = \boldsymbol{r_1} + \boldsymbol{s_1} = 1 + 2 = 3 \\ \boldsymbol{s_2} = az\ a\ legkisebb, 0\text{-}nál\ nagyobb\ \ természetes\ szám, amely\ nem\ szerepel\ az\ \boldsymbol{r_1}, \boldsymbol{r_2}, \boldsymbol{s_1}\ \text{elemek között, tehát 4.} \end{cases}$$

…

$$\begin{cases} \boldsymbol{r_n} = \boldsymbol{r_{n-1}} + \boldsymbol{s_{n-1}} \\ \boldsymbol{s_n} = a\ legkisebb, 0\text{-}nál\ nagyobb\ természetes\ szám, amely\ nem\ szerepel\ az\ \boldsymbol{r_1}, \boldsymbol{r_2}, \ldots, \boldsymbol{r_n}\ \text{és}\ \boldsymbol{s_1}, \boldsymbol{s_2}, \ldots, \boldsymbol{s_{n-1}}\ \text{elemek között.} \end{cases}$$

Követelmény

Számítsátok ki, adott n esetében r_n és s_n értékét.

Bemeneti adatok

A `hrs.in` bemeneti állomány első sorában az n természetes szám található.

Kimeneti adatok

A `hrs.out` kimeneti állomány egyetlen sorába két természetes számot kell írnotok, amelyek közül az első az ***r*** sorozat ***n***. eleme és a második az ***s*** sorozat ***n***. eleme.

Megszorítások és pontosítások

- $2 \leq \boldsymbol{n} \leq 65\,000$.

Példa

`hrs.in`	`hrs.out`
`15`	`150 20`