

Oszthatóság 17-tel

17-tel úgy vizsgálhatjuk meg az oszthatóságot, hogy a szám első számjegyétől az utolsó előtti számjegyéig képzett számból kivonjuk az utolsó számjegy ötszörösét. A folyamat ismételhető. Pl.: $132770 \rightarrow 13277 - (0 \cdot 5) = 13277 \rightarrow 1327 - (7 \cdot 5) = 1292 \rightarrow 129 - (2 \cdot 5) = 119$. 119 osztható 17-tel, tehát 132770 is osztható 17-tel. Az ismétlés megáll, ha a kapott szám nullánál kisebb vagy egyenlő lenne.

Írj programot, amely eldönti egy számról, hogy osztható-e 17-tel!

Bemenet

A *standard bemenet* első sorában a szám szerepel ($1 \leq N \leq 1\,000\,000\,000$), aminek a 17-tel oszthatóságát vizsgáljuk.

Kimenet

A *standard kimenet* első sorába az IGEN vagy a NEM szót kell írni, attól függően, hogy N osztható-e 17-tel! A második sorba a fenti módszerrel kiszámolt közbülső számok kerüljenek, a kiszámítás sorrendjében! Ha az utolsó szám 0, azt még ki kell írni! Üres sort kell kiírni, ha nincs egyetlen közbülső szám sem!

Példa

Bemenet	Kimenet
132770	IGEN 13277 1292 119
Bemenet	Kimenet
132771	NEM 13272 1317 96
Bemenet	Kimenet
51	IGEN 0
Bemenet	Kimenet
27	NEM {üres második sor}

Korlátok

Időlimit: 0.2 mp.

Memórialimit: 32 MB

Zenehallgatás

Petya összeállított egy lejátszási listát az N kedvenc dalából. Mindegyik dalnak ismerjük a hosszát másodpercben. A lejátszási listát végtelenítve hallgatja, tehát amint véget ért, előlről kezdi, és újra ugyanolyan sorrendben következnek a dalok. Felírt magának K időpontot, és kíváncsi arra, hogy ezekben az időpontokban melyik dal fog szólni.

Készíts programot, amely megadja, hogy az egyes időpontokban hányadik dal fog szólni!

Bemenet

A *standard bemenet* első sorában dalok száma ($1 \leq N \leq 100\,000$) és az időpontok száma ($1 \leq K \leq 100\,000$) van. A második sor i . száma az i . dal hossza másodpercben ($1 \leq T_i \leq 10\,000$). A harmadik sorban a K időpont van ($1 \leq P_i \leq 10^9$), amelyekre meg kell határozni, hogy abban az időpontban melyik dal szól.

Kimenet

A *standard kimenet* első sorába egyetlen sora K dal sorszámát tartalmazza, ahol az i -edik szám a P_i -edik másodpercben szóló dal sorszáma!

Példa

Bemenet	Kimenet
3 2	2 1
2 4 3	
6 10	

Korlátok

Időlimit: 0.2 mp.

Memórialimit: 32 MB

Pontozás

A pontok 30%-a szerezhető olyan tesztekre, ahol $K \leq 1000$ és $N \leq 1000$.

Kedves Versenyző! A megoldások értékelésénél csak a **programok futási eredményeit** vesszük tekintetbe. Ezért igen fontos a **feladatban leírtak pontos betartása**. A programokat csak a feladatkiírásban leírt szabályoknak megfelelő adatokkal próbáljuk ki, emiatt nem kell ellenőrizni, hogy a bemenő adatok helyesek-e.

1. feladat: Sípálya (25 pont)

Egy hójelentés N sípályán mért hóréteget tartalmazza.

Készíts programot (`sipalya.pas,...`), amely beolvassa a sípályák számát ($1 \leq N \leq 20$) és az egyes pályákon a hóréteg vastagságát ($0 \leq V(i) \leq 100$), majd

A. megadja, hogy melyik sípályán a legnagyobb a hóréteg;

B. megad egy sípályát, ahol a hóréteg legalább 100 cm vastag;

C. megadja azokat a sípályákat, ahol nem lehet síelni (azaz a hóréteg vastagsága 0)!

Példa

Bemenet:

Kimenet:

Pályák száma: 7

Legnagyobb: 3

1. pálya: 0

Van 100 cm hó: 2

2. pálya: 150

Nem lehet síelni 3 pályán: 1 4 6

3. pálya: 200

4. pálya: 0

5. pálya: 57

6. pálya: 0

7. pálya: 30

2. feladat: Pénz (30 pont)

Bergengóciában N-féle pénzértmet használnak: $P(1), P(2), \dots, P(N)$ forintost.

Írj programot (`penz.pas, ...`), amely megadja, hogy mely 1 és M forint közötti összegek fizethetők ki egyetlen pénzértmével, melyek legfeljebb 2 pénzértmével és melyek legfeljebb 3 pénzértmével!

A program olvassa be a pénzértmék számát ($1 \leq N \leq 10$), és a kifizetendő összeget ($1 \leq M \leq 1000$)! Ezután olvassa be a pénzértmék értékét ($1 \leq P_i \leq M$)!

A program írja ki a legfeljebb 1, 2, majd a 3 pénzértmével kifizethető összegeket!

Példa:

Bemenet:

Kimenet:

Érték száma: 2

1 érmével: 1 5

Maximális összeg: 100

2 érmével: 1 2 5 6 10

1. érme: 1

3 érmével: 1 2 3 5 6 7 10 11 15

2. érme: 5

3. feladat: Nyelv (20 pont)

Egy programozási nyelven az elágazások az IF szóval kezdődnek és a FI szóval végződnek. Minden program legalább egy, legfeljebb 100 szóból áll

Készíts programot (`nyelv.pas, ...`), amely beolvas szavanként egy szöveget, majd megadja, hogy az elágazások egymásba ágyazása helyes-e! Ha nem helyes, akkor megadja az első hiba okát is. (Pl. hibás egymásba ágyazás az alábbi: ... IF ... FI ... FI ... IF ...)

Megjegyzés: a többi szó bármi lehet, ellenőrzésükkel nem kell foglalkozni.

Bemenet:

Kimenet:

Szavak száma: 7

Hibás: FI utasítás hiányzik

1. szó: IF

2. szó: ALMA

3. szó: IF

4. szó: FI

5. szó: FI

6. szó: IF

7. szó: BARACK

Elérhető összpontszám: 75 pont + 25 pont az 1. fordulóból

Kedves Versenyző! A megoldások értékelésénél csak a **programok futási eredményeit** vesszük tekintetbe. Ezért igen fontos a **specifikáció pontos betartása**. Ha például a feladat szövege adatok valamilyen állományból történő beolvasását írja elő, és a program ezt nem teljesíti, akkor a feladatra nem adunk pontot (akkor sem, ha egyébként tökéletes lenne a megoldás); az objektív értékelés érdekében ugyanis a pontozóknak a programszövegekben egyetlen karaktert sem szabad javítaniuk, s az előre megadott javítási útmutatótól semmiben nem térhetnek el. A programokat csak a feladatkiírásban leírt szabályoknak megfelelő adatokkal próbáljuk ki, emiatt nem kell ellenőrizni, hogy a bemenő adatok helyesek-e, illetve a szükséges állományok léteznek-e (sőt ezért plusz pont sem jár). Ha a programnak valamilyen állományra van szüksége, akkor azt mindig az aktuális könyvtárba kell rakni. Az állományok neve minden esetben rögzített. **Csak olyan programokat értékelünk, amelyek 1 percen belül adnak végeredményt!**

1. feladat: Falu (18 pont)

Ismerjük egy megye települései (falvak, városok) közötti utak hosszát. Zsákfalunak nevezzük azt a falut, ahova csak egyetlen út vezet (és onnan tovább már nem lehet menni, csak visszafelé). A településeket sorszámmal azonosítjuk.

Készíts programot (`falu.pas`, ...), amely megadja:

- A. a zsákfalvak számát;
- B. azt a települést, ahova a legtöbb út vezet szomszédos településről;
- C. az egymáshoz legközelebbi 2, nem szomszédos települést (ha több ilyen van, akkor bármelyik megadható)

A `falu.be` szöveges állomány első sorában a települések ($2 \leq N \leq 1000$) és az utak száma van ($1 \leq M \leq 100000$), egy szóközzel elválasztva. A következő M sor mindegyikében három egész szám van, egy-egy szóközzel elválasztva: egy-egy út két végpontjának sorszáma és a köztük levő út hossza.

A `falu.ki` szöveges állomány első sorába a zsákfalvak számát; a második sorba a legtöbb utas település sorszámát (ha több van, bármelyik megadható), a harmadikba pedig a két legközelebbi település sorszámát (szóközzel elválasztva, ha több egyforma távolság is van, bármelyik településpár megadható) kell írni!

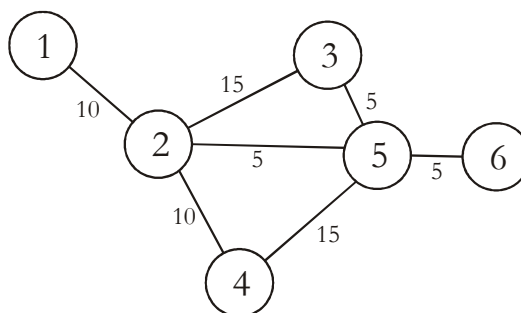
Példa:

`falu.be`

```
6 7
1 2 10
2 3 15
2 4 10
2 5 5
3 5 5
4 5 15
5 6 5
```


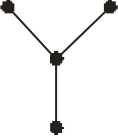
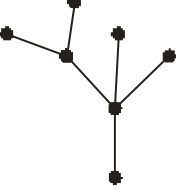
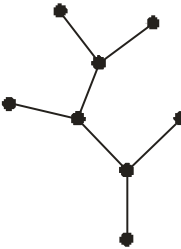
`falu.ki`

```
2
2
3 6
```



2. feladat: Fa (18 pont)

Minden fát leírhatunk egy karaktersorozattal. Ebben a leírásban X betűk és zárójelek fognak szerepelni. Az X ágat jelent, az ágak végi elágazásokat pedig zárójelbe tesszük.

			
X	X(X)(X)	X(X(X)(X))(X)(X)	X(X(X)(X(X)(X)))(X)

Írj programot (`fak.pas`, ...), amely megadja:

- A. a fa magasságát (a földtől milyen messze van a legmesszebb levő ágvég);
- B. a fa elágazásai számát (a törzs nem számít elágazásnak);
- C. egy helyen a legnagyobb elágazásszámot!

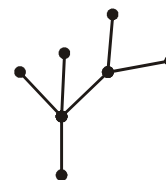
A `fak.be` szöveges állomány egyetlen sorában a fát leíró szöveg van (hossza legfeljebb 10000 karakter).

A `fak.ki` szöveges állomány első sorába a fa magasságát, a második sorába a fa elágazásai számát, a harmadik sorába pedig a legnagyobb elágazásszámot kell kiírni!

Példa:

`fak.be`
X (X) (X) (X (X) (X))

`fak.ki`
3
5
3

3. feladat: Pakolás (18 pont)

Egy raktárban egyetlen hosszú sorban ládák vannak. Minden láda kocka alakú, de méretük különböző lehet. A ládák egymásra rakásával akarnak helyet felszabadítani. A biztonsági előírás szerint több ládát is lehet egymásra rakni, de minden ládát csak nálánál nagyobbra lehet helyezni. Továbbá, az i -edik helyen lévő ládát csak akkor lehet rárakni a j -edik helyen lévő torony tetejére, ha az i -edik és j -edik helyek között már nincs láda (j lehet akár kisebb, akár nagyobb, mint i). Minden ládát legfeljebb egyszer lehet mozgatni.

Készíts programot (`pakol.pas`, ...), amely kiszámítja, hogy legkevesebb hány toronyba lehet a ládákat összepakolni!

A `pakol.be` szöveges állomány első sorában a ládák N ($2 \leq N \leq 30000$) száma van. A második sor pontosan N pozitív egész számot tartalmaz egy-egy szóközzel elválasztva, a ládák méreteit. A második sorban lévő számok mindegyike 1 és 30000 közötti érték.

A `pakol.ki` szöveges állomány első és egyetlen sora egy egész számot tartalmazzon, azt a legkisebb M számot, hogy a bementben megadott ládasor összepakolható M számú toronyba!

Példa:

`pakol.be`
10
1 2 4 6 7 5 3 2 5 3

`pakol.ki`
2

4. feladat: Játék (21 pont)

Tekintsük azt az egyszemélyes játékot, amelyet N sorból és M oszlopból álló négyzetácsos táblán játszanak. Minden mező vagy üres, vagy csapda. Egy bábut kell mozgatni a táblán. A bábu kezdetben a tábla bal felső sarkában van, és a jobb alsó sarokba kell eljuttatni az alábbi lépés-szabályt betartva:

- Csak olyan mezőre lehet lépni, ahova még nem lépett a bábu.
- Csapda mezőre nem lehet lépni.
- Csak a négy szomszédos mező valamelyikére lehet lépni.
- Egy lépésben csak jobbra, vagy lefelé lehet lépni.

Készíts programot (`jatek.pas`, ...), amely kiszámítja, hogy hányféle képen lehet eljuttatni a bábut a bal felső sarokból a jobb alsóba!

A `jatek.be` szöveges állomány első sorában két egész szám van, a sorok N és oszlopok M száma ($1 \leq N \leq 10$, $1 \leq M \leq 10$). A további N sor mindegyike M egész számot tartalmaz egy-egy szóközzel elválasztva. Minden szám vagy 0, vagy 1. A sorban az i -edik szám 1, akkor a megfelelő mező csapda, egyébként a mező üres.

A `jatek.ki` szöveges állomány egyetlen sora egy egész számot tartalmazzon, azt, hogy hány féleképpen lehet eljuttatni a bábut a bal felső sarokból a jobb alsóba!

Példa:

<code>jatek.be</code>	<code>jatek.ki</code>
5 6	7
0 0 0 0 0 0	
0 1 0 0 1 0	
0 0 1 0 0 0	
1 0 1 0 0 0	
0 0 0 0 1 0	

Elérhető összpontszám: 75 pont + 25 pont az 1. fordulóból

Kedves Versenyző! A megoldások értékelésénél csak a **programok futási eredményeit** vesszük tekintetbe. A programokat csak a feladatkiírásban leírt szabályoknak megfelelő adatokkal próbáljuk ki, emiatt nem kell ellenőrizni, hogy a bemenő adatok helyesek-e.

1. feladat: Lövészverseny (25 pont)

Az időjárás előrejelzésben ismerjük előre N ($2 \leq N \leq 100$) nap várható minimális és maximális hőmérsékletét. Készíts programot (`idojaras.pas`, `idojaras.c`, ...), amely beolvassa N értékét és a $2*N$ db hőmérsékletet, majd megadja:

- A. azt a K napos időtartamot (ha van), amelyben az előrejelzés szerint folyamatosan fagy lesz;
- B. azt a két szomszédos napot, ahol a legnagyobbat változik a hőmérséklet;
- C. azokat a napokat, ahol a napi minimális hőmérséklet a napi átlaghőmérsékletek átlaga fölötti!

Példa:

Bemenet:

Napok száma?5

Fagy hány napon keresztül?2

- 1. nap minimuma, maximuma: -9 -2
- 2. nap minimuma, maximuma: -1 4
- 3. nap minimuma, maximuma: -5 -4
- 4. nap minimuma, maximuma: -6, -1
- 5. nap minimuma, maximuma: 5 8

Kimenet:

Folyamatos fagy: 3 4

Legnagyobb változás: 4 5

Átlag fölöttiek: 2 5

2. feladat: Tördelés (25 pont)

Wikipédián található az alábbi leírás a webcímekről (most csak azt engedjük meg, ami ebben a leírásban szerepel):

Egy tipikus, egyszerű webcím így néz ki:

<http://hu.wikipedia.org:80/wiki>

Ennek részei:

- A `http` (vagy `https`) a használandó protokoll. A protokoll neve után kettőspont (`:`) írandó.
- A `hu.wikipedia.org` a célgép tartományneve. Ez elé két perjel (`//`) írandó.
- A `80` a célgép azon hálózati portszáma, amin kérésünket várja; ez elé kettőspont (`:`) írandó. Ezt a részt gyakran teljesen elhagyhatjuk, például esetünkben a `http` protokoll alapértelmezett portszáma a `80`.
- A `/wiki` a kért elérési út a célgépen. Ez a rész mindig a perjellel (`/`) kezdődik.

A legtöbb böngésző nem is igényli, hogy a „`http://`” részt begépeljük egy weblap eléréséhez, hiszen az esetek döntő többségében úgyis ezt használjuk. Egyszerűen begépelhetjük a lap címét, például: „`hu.wikipedia.org/wiki/Bit`”. A főlap megtekintéséhez általában elég a tartomány nevét beírni, például „`hu.wikipedia.org`”.

A webcím a példákban szereplőtől eltérő jeleket (pl. szóközt, relációkat, ...) nem tartalmazhat. A webcímek egyéb részeket is tartalmazhatnak, `http` esetében például az elérési út után, egy kérdőjel (`?`) mögé helyezve keresési kérdés szerepelhet, ami egy `get` metódusú HTML űrlapból származik.

Példa:

<http://hu.wikipedia.org/w/wiki.phtml?title=Bit&action=history>

Készíts programot (`url.pas`, `url.c`, ...), amely egy beolvasott webcímet (legfeljebb 100 karakteres) a fenti szempontok szerint ellenőriz! Hay valamely szempont szerint hibás, akkor megadja, hogy m a hiba a szövegben.

Példa:Bemenet: `http://www.njszt.hu\tehetseg`

Kimenet: Hibás karakter: \

3. feladat: Elszigetelt falu (25 pont)

Egy megyében N ($1 \leq N \leq 100$) falu van. A falvakat M ($3 \leq M \leq 1000$) út köti össze, ismerjük minden út hosszát. A legelszigeteltebb falunak azt nevezzük, amelytől a legközelebbi szomszédja a lehető legtávolabb van.

Készíts programot (`falu.pas`, `falu.c`, ...), amely megadja a legelszigeteltebb falut! Ha több megoldás van, akkor bármelyik kiírható.

Példa:

Bemenet:

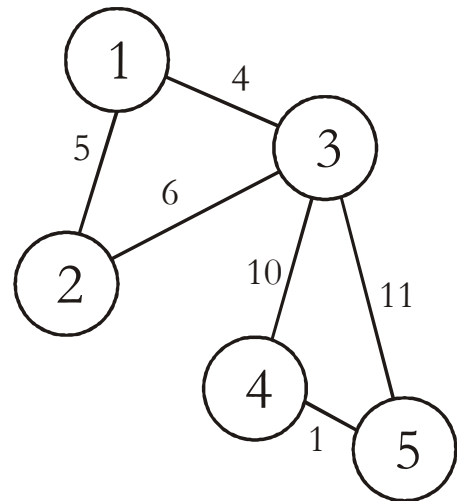
Falvak száma?5

Utak száma?6

```
1. út kezdete, vége, hossza?1 2 5
2. út kezdete, vége, hossza?2 3 6
3. út kezdete, vége, hossza?3 1 4
4. út kezdete, vége, hossza?3 4 10
5. út kezdete, vége, hossza?4 5 1
6. út kezdete, vége, hossza?3 5 11
```

Kimenet:

A legelszigeteltebb: 2

**Elérhető összpontszám: 75 pont + 25 pont a 2. fordulóból**

Kedves Versenyző! A megoldások értékelésénél csak a **programok futási eredményeit** vesszük tekintetbe. Ezért igen fontos a **feladatban leírtak pontos betartása**. A programokat csak a feladatkiírásban leírt szabályoknak megfelelő adatokkal próbáljuk ki, emiatt nem kell ellenőrizni, hogy a bemenő adatok helyesek-e.

1. feladat: Kártya (25 pont)

Egy kártyajátékban az egyes lapoknak számértékük van. Minden lapot egy színnel és egy figurával adunk meg. A színek: piros, zöld, tök, makk. A figurák: 7-es, 8-as, 9-es, 10-es, alsó, felső, király, ász. A számot tartalmazó figurák annyit érnek, amennyi a ráírt szám. Az alsó 2-t, a felső 3-at, a király 4-et, az ász 11-et ér. A piros lapoknál az értéket duplán kell számítani.

Készíts programot (`kartya.pas`,...), amely beolvassa egy játékos N ($1 \leq N \leq 4$) kártyáját, majd megadja, hogy a lapok összesen hány pontot érnek!

Példa

Bemenet:

Kártyák száma? 3

1. kártya színe? piros

1. kártya figurája? alsó

2. kártya színe? tök

2. kártya figurája? 7-es

3. kártya színe? tök

3. kártya figurája? ász

Kimenet:

A kártyák értéke: 22

2. feladat: Bábu (25 pont)

Egy játéktáblán a 0. időegységben L bábu van. Mindegyiket elindítjuk valamerre. Egy időegység alatt mindegyik a neki megfelelő távolságra mozdul el, a tábla széléről visszafordulnak. Lehetséges, hogy előbb-utóbb két bábu összeütközik: ugyanarra a helyre lépnének vagy átlépnének egymáson.

Írj programot (`babu.pas`, ...), amely megadja, hogy K időegységben belül mikor ütközik legelőször két bábu!

A program olvassa be a tábla szélességét ($1 \leq N \leq 100$), a bábuk számát ($1 \leq L \leq 10$) és az időtartamot ($1 \leq K \leq 100\,000$)! Ezután olvassa be a bábuk kezdő helyét ($1 \leq S_i \leq N$) és mozgás irányát ($X_i \in \{J, B\}$ – jobbra, balra)!

A program írja ki az első ütközés időpontját! Ha K időegységben belül nincs ütközés, akkor -1-et kell kiírni!

Példa:

A tábla hossza: 10

A bábuk száma: 2

Az időtartam: 10

1 bábu helye: 3, iránya: B

2. bábu helye: 8, iránya: B

Ütközés időpont: 5

		😊					😊		
--	--	---	--	--	--	--	---	--	--

Magyarázat: A bábuk helyzete időegységenként:

1: 2, 7

2: 1, 6

3: 2, 5

4: 3, 4

5: 4, 3

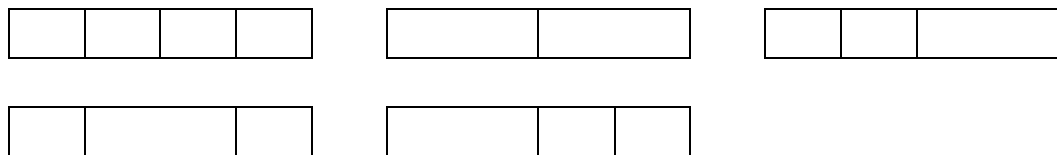
Az 5. időegységre az 1-es és a 2-es bábu egymáson átlépett volna, azaz az 5. időegységben ütköztek.

3. feladat: Járdakövezés (25 pont)

Egy N ($1 \leq N \leq 80$) egység hosszú járdát 1 és 2 méretű lapokkal szeretnénk kikövezni. Hányféleképpen lehet ezt megtenni?

Készíts programot (`lapok.pas, ...`), amely kiszámítja, hogy egy N egység hosszú járdát hányféleképpen lehet kikövezni 1 és 2 méretű lapokkal!

Példa: $N=4$ egység hosszú járdát 5-féleképpen lehet kikövezni, a kikövezési lehetőségei:



Bemenet:

Kimenet:

A járda hossza? 4

A kikövezések száma: 5

Elérhető összpontszám: 75 pont + 25 pont az 1. fordulóból

Kedves Versenyző! A megoldások értékelésénél csak a **programok futási eredményeit** vesszük tekintetbe. Ezért igen fontos a **specifikáció pontos betartása**. Ha például a feladat szövege adatok valamilyen állományból történő beolvasását írja elő, és a program ezt nem teljesíti, akkor a feladatra nem adunk pontot (akkor sem, ha egyébként tökéletes lenne a megoldás); az objektív értékelés érdekében ugyanis a pontozóknak a programszövegekben egyetlen karaktert sem szabad javítaniuk, s az előre megadott javítási útmutatótól semmiben nem térhetnek el. A programokat csak a feladatkiírásban leírt szabályoknak megfelelő adatokkal próbáljuk ki, emiatt nem kell ellenőrizni, hogy a bemenő adatok helyesek-e, illetve a szükséges állományok léteznek-e (sőt ezért plusz pont sem jár). Ha a programnak valamilyen állományra van szüksége, akkor azt mindig az aktuális könyvtárba kell rakni. Az állományok neve minden esetben rögzített. **Csak olyan programokat értékelünk, amelyek 1 percen belül adnak végeredményt!**

1. feladat: Kép (20 pont)

Ugyanarról a területről két időpontban készítettünk fényképet. A fényképek négy széléről le szeretnénk vágni azt a részt, amelyek egyformák.

Készíts programot (`kep.pas`, ...), amely megadja hogy a kép 4 széléről maximum mekkora téglalapok vághatók le!

A `kep.be` szöveges állomány első sorában a fényképek sorai és oszlopai száma van ($1 \leq N, M \leq 1000$), egy szóközzel elválasztva. A következő N sorban az első kép, az azt követő N sorban a második kép képpontjai vannak. Minden sor M képpont leírását tartalmazza, egymástól egy-egy szóközzel elválasztva. A képpontokat egy 0 és 255 közötti fényességértékkel adjuk meg.

A `kep.ki` szöveges állomány első sorába a legnagyobb balról, alulról, jobbról, illetve felülről levágható téglalap szélességét kell írni!

Példa:

<code>kep.be</code>	<code>kep.ki</code>
8 10	1 1 3 2
1 1 1 1 1 1 1 1 1 1	
2 2 2 2 2 3 3 3 3 3	
2 2 2 2 2 2 2 2 2 2	
2 2 2 2 2 2 5 5 5 5	
1 1 1 1 1 1 1 1 1 1	
1 1 1 1 1 1 1 1 1 1	
1 1 1 1 1 1 1 1 1 1	
0 0 0 0 0 0 0 0 0 0	

1 1 1 1 1 1 1 1 1 1	
2 2 2 2 2 3 3 3 3 3	
2 2 9 9 2 2 2 2 2 2	
2 2 2 2 2 2 5 5 5 5	
1 1 1 1 1 1 1 1 1 1	
1 3 1 1 3 1 1 1 1 1	
1 1 1 1 1 1 5 1 1 1	
0 0 0 0 0 0 0 0 0 0	

2. feladat: Játék (20 pont)

Egy játéktáblán a 0. időegységben L bábu van. Mindegyiket elindítjuk valamerre. Egy időegység alatt mindegyik a neki megfelelő távolságra mozdul el, a tábla szélére érve megállnak. Lehetséges, hogy előbb-utóbb két bábu összeütközik: ugyanarra a helyre lépnének vagy átlépnenek egymáson.

Írj programot (`tabla.pas`, ...), amely megadja, hogy K időegységen belül mikor ütközik legelőször két bábu!

A `tabla.be` szöveges állomány első sorában a játéktábla sorai és oszlopai száma ($1 \leq N, M \leq 100$), a bábuk száma ($1 \leq L \leq 10$) és az időtartam ($1 \leq K \leq 100\,000$) van, egyetlen szóközzel elválasztva. A következő L sor egy-egy bábu leírását tartalmazza: a kezdő helyét ($1 \leq S_i \leq N$, $1 \leq O_i \leq M$) és mozgás irányát ($X_i \in \{F, L, J, B\}$ – fel, le, jobbra, balra), egy-egy szóközzel elválasztva.

A `tabla.ki` szöveges állomány egyetlen sorába az első ütközés időpontját kell írni! Ha K időegységen belül nincs ütközés, akkor -1-et kell kiírni!

Példa:

<code>tabla.be</code>	<code>tabla.ki</code>
7 10 3 100	3
4 3 J	
2 6 F	
4 8 B	

Magyarázat: A bábuk helyzete időegységenként:

1: (4, 4), (1, 6), (4, 7)
2: (4, 5), (1, 6), (4, 6)
3: (4, 6), (1, 6), (4, 5)

A 3. időegységre az 1-es és a 3-as bábu egymáson átlépett volna, azaz a 3. időegységben ütköztek.

3. feladat: Ütemezés (20 pont)

Egy vállalkozó alkatrészek gyártásával foglalkozik. Minden alkatrészen kétféle műveletet kell elvégeznie, **A** és **B** műveletet. Mindkét művelet elvégzésére egy-egy munkagépe van, amelyek egymástól függetlenül tudnak dolgozni, de egy alkatrészen egyszerre csak egyik művelet végezhető. Az alkatrészen a két műveletetszőleges sorrendben el lehet végezni. Minden legyártandó alkatrésze ismert, hogy mennyi időt igényel az **A**, valamint a **B** művelet elvégzése.

Készíts programot (`utemez.pas`, ...), amely kiszámítja, hogy legkevesebb mennyi idő alatt lehet legyártani az összes alkatrészt!

A `utemez.be` szöveges állomány első sorában az alkatrészek N ($2 \leq N \leq 1000$) száma van. Az alkatrészeket az $1, \dots, N$ számokkal azonosítjuk. A második és a harmadik sor pontosan N egész számot tartalmaz egy-egy szóközzel elválasztva, a legyártandó alkatrészek elvégzendő **A**, illetve **B** műveletek idejét. A második sorban az i -edik szám az i -edik alkatrészen végzendő **A** művelet ideje. A harmadik sorban az i -edik szám pedig az i -edik alkatrészen végzendő **B** művelet ideje. A második és harmadik sorban lévő számok mindegyike 1 és 5000 közötti érték.

A `utemez.ki` szöveges állomány első sora egy egész számot tartalmazzon, azt a legkisebb T időt, amely alatt a két gép le tudja gyártani az összes alkatrészt!

Példa:

<code>utemez.be</code>	<code>utemez.ki</code>
3	50
4 20 15	
11 30 3	

4. feladat: Járdakövezés (15 pont)

Egy N egység hosszú járdát 1, 2 és 3 méretű lapokkal szeretnénk kikövezni. Hányféleképpen lehet ezt megtenni?

Készíts programot (`kovezes.pas`, ...), amely kiszámítja, hogy egy N egység hosszú járdát hányféleképpen lehet kikövezni 1, 2 és 3 méretű lapokkal!

A `kovezes.be` szöveges állomány egyetlen sorában a járda hossza ($1 \leq N \leq 70$) van.

A kovezes.ki szöveges állomány egyetlen sora a lehetséges kikövezések számát tartalmazza!

Példa:

kovezes.be

4

--	--

kovezes.ki

7

--	--

--	--	--

--	--

--	--	--

--	--	--

--	--	--	--

Elérhető összpontszám: 75 pont + 25 pont az 1. fordulóból