Kedves Versenyző! A megoldások értékelésénél csak **a programok futási eredményeit** vesszük tekintetbe. Ezért igen fontos a **specifikáció pontos betartása.** 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. Ha a programnak valamilyen állományra van szüksége, akkor azt mindig az aktuális könyvtárba kell rakni. Az állomány neve minden esetben rögzített.

1. feladat: Futó a sakktáblán (20 pont)

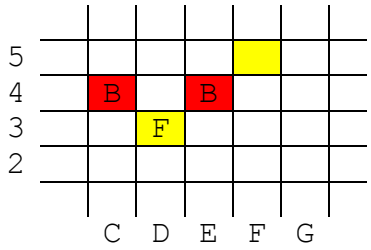
Egy sakktáblán elhelyezünk világos és sötét bábukat, valamint adott helyre egy világos futót. Készíts programot **futo.cbp (main.cpp)**, amely megadja azt minimális számú lépésből álló lépéssorozatot, amellyel a futó egy adott másik helyre eljuthat úgy, hogy más bábut nem léphet át, nem is üthet le!

A **futo.be** állomány első sorában a futó kezdeti helye található egy betűvel (A..H) és egy számjeggyel (1..8) kódolva, a következő sorokban a többi bábu helyzetét adjuk meg (mindegyikben egyet), s az utolsó sor tartalmazza a célhelyet.

A **futo.ki** állomány első sorába a futó lépései számát, a második sorba a futó kezdeti pozícióját, a további sorokba pedig a futó lépései utáni helykoordinátákat kell írni, soronként egy helyet! Ha nincs megoldás, akkor az egyetlen sorba -1-et kell kiírni!

Példa:

Bemenet	Kimenet
D3	3
C4	D3
E4	E2
F5	G4
	F5



2. feladat: Kapcsoló (25 pont)

Három különböző nyomtatót szeretnénk számítógépünkhöz kapcsolni, de a gépnek csak két portja van. Sikerült vásárolni egy olyan 2→3 kapcsolót, amelynek két bemenete a két porthoz, három kimenete pedig a három nyomtatóhoz köthető. Így mind a három nyomtatóra nyomtathatunk, csak a nyomtatás előtt egy kapcsológombbal a megfelelő port→nyomtató kapcsolatot ki kell alakítani, ha a kívánt nyomtató éppen nincs egyik porthoz sem kötve a 2→3 kapcsoló által.

Adott nyomtatandó anyagoknak egy sorozata, megjelölve, hogy az anyagot melyik nyomtatón kell kinyomtatni. A nyomtatókat az A, B és C karakterekkel azonosítjuk. A nyomtatást a megadott sorrendben kell elvégezni, és egyszerre csak egy anyagot nyomtathatunk. Kezdetben egyik nyomtató sincs porthoz kapcsolva.

Készíts programot **kapcsolo.cbp (main.cpp)**, amely kiszámítja, hogy minimálisan hány kapcsolás szükséges a bemenetként megadott nyomtatási igény végrehajtásához!

A **kapcsolo.be** szöveges állomány első sorában a nyomtatandó anyagok száma (1≤N≤1000000 – 1 millió) van. A következő sorban N karakterrel adjuk meg az N anyag nyomtatóazonosítóját (A, B és C betűk, elválasztójel nincs közöttük).

A **kapcsolo.ki** szöveges állomány egyetlen sorába a nyomtatáshoz szükséges kapcsolások minimális számát kell írni!

Példa:

Bemenet	Kimenet
10	4
AA**B**A**CA**BAAB	

3. feladat: Blokk-fa (30 pont)

Egy blokk-fa az alábbi szerkezetű blokkok sorozata:

- elemek száma (N),
- elemeket nagyság szerint rendezve tartalmazó tömb (maximum MAXN elemű, 1-től N-ig van kitöltve),
- blokksorszámokat tartalmazó tömb (0-tól MAXN-ig indexelt tömb, 0-tól N-ig van kitöltve).

A 0-ás indexű blokksorszám olyan blokk sorszáma, amelyben levő értékek kisebbek a blokk 1. elemének értékénél; az 1-es indexű blokksorszám pedig olyané, amelynek elemei nagyobbak a blokk 1. eleménél, de nem nagyobbak a 2.-nál; ... A blokksorszám 0, ha nincs az adott értékek közé eső elem.

Készíts programot **bfa.cbp (main.cpp)**, amely beolvas a bfa.be állományból egy blokk-fát, majd kiírja a blokk-fa elemei számát, valamint a magasságát!

Korlátok: MAXN≤1000, a blokksorszámokat 1-től folyamatosan adjuk ki, legfeljebb 10000 lehet.

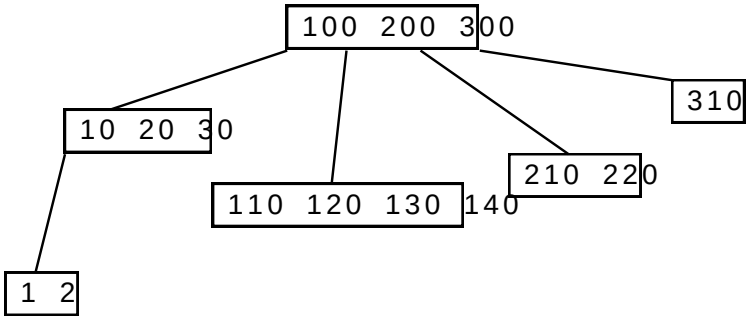
A **bfa.be** állomány sorhármasokat tartalmaz. Mindegyik első sora az adott blokkban levő elemek számát tartalmazza, a második magukat az elemeket egy-egy szóközzel elválasztva, a harmadik pedig a következő blokkok sorszámait. (Az i. blokk az állomány i. sorhármasában található.)

A **bfa.ki** állomány első sorába a blokk-fa elemei számát, a második sorába a magasságát kell írni!

Példa:

Bemenet:

```
3
100 200 300
2 3 4 5
3
10 20 30
6 0 0 0
2
110 120 130 140
0 0 0 0 0
4
210 220
0 0 0
1
310
0 0
2
1 2
0 0
```



Kimenet:

```
15
3
```

4. feladat: Folyók (25 pont)

Magyarország összes folyójáról tudjuk, hogy melyik másik folyóba folyik bele (a folyók N száma legfeljebb 100000000 – 100 millió). Készíts programot **folyok.cbp (main.cpp)**, amely két konkrét folyóról megadja, hogy összefolynak-e valahol Magyarország területén, s ha igen, akkor melyiknek hány folyón kell átjutnia az összefolyásig!

A **folyok.be** állomány első sorában a folyók száma és a két megvizsgálandó folyó sorszáma van, egy-egy szóközzel elválasztva, minden további sora két folyó sorszámot tartalmaz egy szóközzel elválasztva, az első amelyikbe folyik bele a második.

A **folyok.ki** állományba két sort kell írni! Az első tartalma IGEN legyen, ha a két folyó összefolyik valahol, illetve NEM, ha nem folynak össze! Ha az első sor IGEN, akkor a második sorba két szám kerüljön egy szóközzel elválasztva: azon folyók száma, amelyeken keresztül a két adott folyó összefolyik! Ha nem folynak össze, akkor nincs második sor.

Példa:

Bemenet:	Kimenet:
6 6 3 1 4 5 2 2 3 5 6	IGEN 1 2