

Varázslatos Város (Magic City)

Taskent polgármestere újra akarja tervezni a híres Varázslatos Város vidámparkot. Adott egy K pozitív egész szám, és a feladatod egy park megtervezése, a következők szerint:

- Válasszunk egy N egész számot, a látnivalók számát. A látnivalók 0 és $N - 1$ közötti egész számokkal lesznek azonosítva.
- Kétirányú járdákkal kötünk össze két-két látnivalót. Lehet több járda is ugyanazon két látnivaló között. **Nem szükséges, hogy bármely két látnivaló elérhető legyen egymásból a járdákon keresztül.**
- Minden i -re (ahol $0 \leq i < N$), az i . látnivalónak adjunk egy $T[i]$ típust. $2K$ típus közül választhatunk, amelyek 0 -tól $2K - 1$ -ig vannak számozva. Minden típust legalább egy látnivalóhoz hozzá kell rendelni. Különböző látnivalóknak lehet azonos típusa.

A piackutatás kimutatta, hogy:

- Minden látogató három látnivalót szeretne meglátogatni anélkül, hogy két azonos típusú látnivalót keresne fel egymás után.
- A látogatók nem szeretik azokat a látnivalókat, amelyekhez sok járda csatlakozik.

Ahhoz, hogy **minden** potenciális látogató igényeit kielégítse, a polgármester két feltételt szab a terveddel kapcsolatban.

Egy (t_1, t_2, t_3) **típusokból** álló rendezett hármas (ahol $0 \leq t_1, t_2, t_3 < 2K$) **érdekes** ha $t_1 \neq t_2$ és $t_2 \neq t_3$. Megjegyzendő, hogy t_1 egyenlő lehet t_3 -mal. Így $2K \cdot (2K - 1)^2$ érdekes hármas van.

1. feltétel: Minden (t_1, t_2, t_3) érdekes hármashoz kell lennie három látnivalónak: a_1, a_2, a_3 (ahol $0 \leq a_1, a_2, a_3 < N$), amelyekre:

- Az a_1, a_2, a_3 típusai megegyeznek a t_1, t_2, t_3 típusokkal. Azaz $T[a_1] = t_1$, $T[a_2] = t_2$ és $T[a_3] = t_3$.
- a_1 és a_2 között van egy járda.
- a_2 és a_3 között van egy járda.

Az lényegtelen, hogy van-e járda a_1 és a_3 között. Azt is vedd figyelembe, hogy amikor $t_1 = t_3$, akkor az a_1 és a_3 látnivalók lehetnek azonosak.

2. feltétel: Minden látnivalóhoz **legfeljebb** K járda csatlakozhat.

Ez a feladat 50 **output-only** részfeladatból áll, **részleges pontozással**. Minden részfeladat egy adott K értéknek felel meg, és olyan parkot kell tervezned, amely az adott K értékre vonatkozóan kielégíti a fenti összes feltételt. A pontszámod a megoldásodban szereplő látnivalók számától függ: kevesebb látnivaló több (vagy ugyanannyi) pontot eredményez.

Megvalósítás

A megoldás beküldésére kétféleképpen van lehetőség, és minden részfeladathoz a kettő közül bármelyiket használhatod:

- **Függvényhívás**
- **Kimeneti fájl**

A megoldás **függvényhívással** történő beküldéséhez a következő függvényt kell implementálni.

```
std::pair<std::vector<int>, std::vector<std::pair<int, int>>> construct(int K)
```

- K : a látványosságtípusok számának fele, valamint az egyes látványosságokhoz kapcsolódó járdák maximálisan megengedett száma.
- Ez a függvény minden részfeladatban pontosan egyszer hívódik meg.

A függvénynek egy (T, E) párt kell visszaadnia, amely egy vidámparkot ír le. Legyen M a parkban lévő járdák száma.

- T : egy N hosszúságú tömb, amely a látnivalók típusait írja le.
- E : egy M hosszúságú tömb, amely a járdákat írja le. Minden $0 \leq j < M$ esetén $E[j] = (U[j], V[j])$ egy kétirányú járdát jelöl $U[j]$ és $V[j]$ látnivalók között ($U[j] \neq V[j]$).

A megoldás **kimeneti fájlként** történő beküldéséhez hozz létre és küldj be egy szövegfájlt a következő formátumban:

```
N M
T[0] T[1] ... T[N-1]
U[0] V[0]
U[1] V[1]
...
U[M-1] V[M-1]
```

A megoldásnak a következő feltételeknek kell megfelelnie ahhoz, hogy **érvényes** legyen:

- $N \leq 2000$
- $0 \leq T[i] < 2K$ minden $0 \leq i < N$ esetén, és minden típust legalább egy látnivalóhoz hozzá kell rendelni.
- $0 \leq U[j], V[j] < N$ és $U[j] \neq V[j]$ minden egyes $0 \leq j < M$ esetén.
- Az 1. és 2. feltételnek teljesülnie kell.

Korlátok

- $1 \leq K \leq 50$

Pontozás

50 részfeladat van, minden 1 és 50 közötti K egész számhoz egy részfeladat tartozik. Az i . részfeladat esetében a K értéke i ($1 \leq i \leq 50$).

Minden részfeladhoz tartozik egy S pontszám és egy P cél látnivalószám az alábbi táblázat szerint.

Részfeladat	S	P
1	1	2
2	8	12
3	9	24
4	9	40
5	9	50
6-10	4	$12 \cdot K$
11-12	3	$12 \cdot K$
13 - 50	1	$12 \cdot K$

Minden részfeladat esetében, ha a megoldásod nem érvényes vidámparkot ír le, akkor a megoldás pontszáma 0 lesz (a CMS-ben `Output isn't correct` hibaként jelenik meg).

Egyébként a pontszámod az alábbiak szerint kerül kiszámításra az N , valamint az S és P paraméterek alapján:

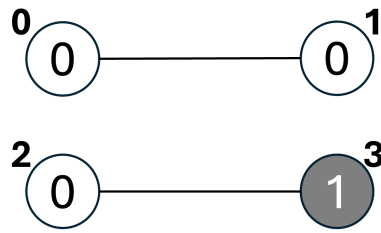
Feltétel	Pontszám
$N \leq P$	S
$P < N \leq 2P$	$(0.4 + 0.3 \cdot \frac{2P-N}{P}) \cdot S$
$2P < N \leq 2000$	$(0.1 + 0.3 \cdot \frac{2P}{N}) \cdot S$

Példa

Tekintsük a következő hívást:

```
construct(1)
```

Ebben a példában $K = 1$, tehát $2K = 2$ típusú látnivaló van. Az alábbi ábra egy érvényes megoldást mutat $N = 4$ látnivaló és $M = 2$ járda esetén. A 0, 1, 2 látnivalók 0 típusúak, a 3 látnivaló pedig 1 típusú.



Két érdekes hármas van:

- $(0, 1, 0)$ típusokból álló hármas esetén választhatjuk az $(a_1, a_2, a_3) = (2, 3, 2)$ látnivalókat.
- $(1, 0, 1)$ típusokból álló hármas esetén választhatjuk az $(a_1, a_2, a_3) = (3, 2, 3)$ látnivalókat.

Ez azt jelenti, hogy az 1. feltétel teljesül.

A függvény tehát visszaadhatja a $([0, 0, 0, 1], [(0, 1), (2, 3)])$ párt. Vedd figyelembe, hogy a példához megadott megoldás esetleg nem optimális $K = 1$ esetén.

Mintaértékelő

Bemeneti formátum:

K

Kimeneti formátum:

```
N M
T[0] T[1] ... T[N-1]
U[0] V[0]
U[1] V[1]
...
U[M-1] V[M-1]
```

Megjegyezzük, hogy a mintaértékelő kimenete megegyezik a kimeneti fájlhoz szükséges formátummal.