

## Magic City

Primarul oraşului Taşkent vrea să reamenajeze celebrul parc de distracţii Magic City. Vi se dă un număr întreg pozitiv  $K$  şi sarcina dumneavoastră este să proiectaţi un parc care să îndeplinească următoarele condiţii.

- Alegeţi un număr întreg  $N$ , numărul de atracţii. Atracţiile vor fi etichetate cu numere întregi de la 0 la  $N - 1$ .
- Adăugaţi alei bidirecţionale între perechi de atracţii **distincte**. Este permisă existenţa a mai multor alei între aceeaşi pereche de atracţii. **Parcul nu trebuie să fie conex**, ceea ce înseamnă că nu este obligatoriu să existe drum între oricare două atracţii folosind alei.
- Pentru fiecare  $i$  astfel încât  $0 \leq i < N$ , atribuiţi atracţiei  $i$  un tip  $T[i]$ . Există  $2K$  tipuri disponibile numerotate de la 0 la  $2K - 1$ . Fiecare tip ar trebui atribuit cel puţin unei atracţii. Acelaşi tip poate fi atribuit unor atracţii diferite.

Studiile de piaţă au arătat că:

1. Fiecare vizitator preferă să viziteze trei atracţii fără a vizita două atracţii de acelaşi tip consecutiv.
2. Vizitatorilor nu le plac atracţiile care sunt conectate la multe alei pietonale.

Primarul vrea să satisfacă **toţi** potenţialii vizitatori şi impune două condiţii proiectului tău.

Numim un triplet ordonat de **tipuri**  $(t_1, t_2, t_3)$  pentru  $0 \leq t_1, t_2, t_3 < 2K$  **interesant** dacă  $t_1 \neq t_2$  şi  $t_2 \neq t_3$ . Observaţi că  $t_1$  ar putea fi egal cu  $t_3$ . Prin urmare, există  $2K \cdot (2K - 1)^2$  triplete interesante.

**Condiţia 1:** Pentru fiecare triplet interesant  $(t_1, t_2, t_3)$ , trebuie să existe trei atracţii  $a_1, a_2, a_3$  (cu  $0 \leq a_1, a_2, a_3 < N$ ) astfel încât:

- Tipurile  $a_1, a_2, a_3$  se potrivesc  $t_1, t_2, t_3$ . Adică,  $T[a_1] = t_1$ ,  $T[a_2] = t_2$  şi  $T[a_3] = t_3$ .
- Există o alee între  $a_1$  şi  $a_2$ .
- Există o alee între  $a_2$  şi  $a_3$ .

Este irelevant dacă există sau nu o alee între  $a_1$  şi  $a_3$ . De asemenea, observaţi că atunci când  $t_1 = t_3$ , atracţiile  $a_1$  şi  $a_3$  pot coincide.

**Condiţia 2:** Fiecare atracţie poate fi conectată la **cel mult**  $K$  alei.

Această problemă constă în 50 de subtask-uri **output only** cu **punctaj parţial**. Fiecare subtask corespunde unei valori specifice a lui  $K$ , iar tu trebuie să proiectezi un parc care să îndeplinească toate condiţiile de mai sus pentru acea valoare a  $K$ . Scorul tău depinde de numărul de atracţii din soluţia ta - mai puţine atracţii pot duce la un scor mai mare.

## Detalii de implementare

Există două metode pentru a trimite soluția și puteți folosi oricare dintre ele pentru fiecare subsarcină:

- **Apel de funcție**
- **Fișier de ieșire**

Pentru a trimite soluția printr-un **apel de funcție**, trebuie să implementați următoarea funcție.

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

- $K$  : jumătate din numărul de tipuri de atracții și de asemenea, numărul maxim permis de alei conectate la fiecare atracție.
- Această funcție este apelată exact o singură dată pentru fiecare subtask.

Funcția ar trebui să returneze o pereche  $(T, E)$  care să descrie un parc de distracții. Fie  $M$  numărul de alei din parcul tău.

- $T$  : un tablou de lungime  $N$  care descrie tipurile de atracții.
- $E$  : o matrice de lungime  $M$  care descrie aleile pietonale. Pentru fiecare  $0 \leq j < M$  ,  $E[j] = (U[j], V[j])$  reprezintă o alee pietonală bidirecțională între atracția  $U[j]$  și atracția  $V[j]$  .

Pentru a trimite soluția printr-un **fișier de ieșire**, creați și trimiteți un fișier text în următorul format:

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

Soluția dumneavoastră trebuie să îndeplinească următoarele restricții pentru a fi considerată **validă**:

- $N \leq 2000$
- $0 \leq T[i] < 2K$  pentru fiecare  $0 \leq i < N$  .
- $0 \leq U[j], V[j] < N$  și  $U[j] \neq V[j]$  pentru fiecare  $0 \leq j < M$  .
- Condițiile 1 și 2 trebuie îndeplinite.

## Restricții

- $1 \leq K \leq 50$

## Punctaj

Există 50 de subtask-uri, câte unu pentru fiecare număr întreg  $K$  de la 1 la 50 . Pentru subtaskul  $i$  ( $1 \leq i \leq 50$ ), valoarea lui  $K$  este  $i$  .

Fiecare subtask are un scor  $S$  și o dimensiune țintă a parcului  $P$  conform tabelului de mai jos.

| Subtask-uri | $S$ | $P$          |
|-------------|-----|--------------|
| 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   | $12 \cdot K$ |

Pentru fiecare subtask, dacă soluția ta nu descrie un parc de distracții valid, atunci scorul soluției tale va fi 0 (raportat ca `Output isn't correct` în CMS).

Altfel, scorul este calculat pe baza lui  $N$  și a parametrilor  $S$  și  $P$  după cum urmează:

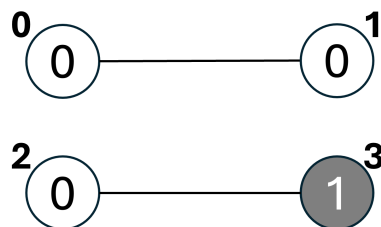
| Stare              | Scor                                       |
|--------------------|--------------------------------------------|
| $N \leq P$         | $S$                                        |
| $P < N \leq 2P$    | $(0.4 + 0.3 \cdot \frac{2P-N}{P}) \cdot S$ |
| $2P < N \leq 2000$ | $(0.1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

## Exemplu

Considerați următorul apel.

```
construct(1)
```

În acest exemplu,  $K = 1$ , deci există  $2K = 2$  tipuri de atracții. Figura de mai jos prezintă o soluție validă cu  $N = 4$  atracții și  $M = 2$  alei. Atracțiile 0, 1, 2 sunt de tipul 0, iar atracția 3 este de tipul 1.



Există două triplete interesante:

- Pentru tipul triplet  $(0, 1, 0)$  , putem alege  $(a_1, a_2, a_3) = (2, 3, 2)$  .
- Pentru tipul triplet  $(1, 0, 1)$  , putem alege  $(a_1, a_2, a_3) = (3, 2, 3)$  .

Aceasta înseamnă că este îndeplinită Condiția 1.

Procedura poate returna perechea  $([0, 0, 0, 1], [(0, 1), (2, 3)])$  . Observați că soluția furnizată pentru acest exemplu ar putea să nu fie optimă pentru  $K = 1$  .

## Grader local

Format de intrare:

$K$

Format de ieșire:

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

Observați că rezultatul evaluării graderului local corespunde formatului necesar pentru fișierul de ieșire.