

Чарівне місто

Мер Ташкента хоче перепроєктувати відомий парк розваг «Чарівне місто». Вам задано додатне ціле число K , і ваше завдання — спроектувати парк, який відповідає наступним вимогам:

- Виберіть ціле число N – кількість атракціонів. Атракціони позначені цілими числами від 0 до $N - 1$.
- Додайте двонаправлені пішохідні доріжки, кожен між парою **окремих** атракціонів. Дозволяється мати більше однієї пішохідної доріжки між однією парою атракціонів. **Необов'язково мати можливість пересуватися між кожною парою атракціонів, використовуючи пішохідні доріжки.**
- Для кожного i такого, що $0 \leq i < N$, призначте атракціону i тип $T[i]$. Доступні $2K$ типи, пронумеровані від 0 до $2K - 1$. Кожен тип має бути призначений принаймні одному атракціону. Різні атракціон можуть мати один і той самий тип.

Дослідження ринку показало, що:

- Кожен відвідувач бажає відвідати рівно три атракціони, не відвідуючи послідовно два атракціони одного типу.
- Відвідувачам не подобаються атракціони, які є кінцевою точкою багатьох пішохідних доріжок.

Щоб задовольнити **всіх** потенційних відвідувачів, мер висуває дві умови до вашого дизайну. Впорядковану трійку **типів** називаємо (t_1, t_2, t_3) для $0 \leq t_1, t_2, t_3 < 2K$ **цікавою**, якщо $t_1 \neq t_2$ та $t_2 \neq t_3$. Зауважте, що t_1 може дорівнювати t_3 . Таким чином, існує $2K \cdot (2K - 1)^2$ цікавих трійок.

Умова 1: Для кожної цікавої трійки (t_1, t_2, t_3) , повинні існувати три атракціони a_1, a_2, a_3 (усі $0 \leq a_1, a_2, a_3 < N$) такі, що:

- Типи a_1, a_2, a_3 відповідають t_1, t_2, t_3 . Тобто, $T[a_1] = t_1, T[a_2] = t_2, T[a_3] = t_3$.
- Між a_1 та a_2 є доріжка.
- Між a_2 та a_3 є доріжка.

Не має значення, чи є доріжка між a_1 та a_3 . Також зауважте, що коли $t_1 = t_3$, атракціони a_1 та a_3 можуть бути однаковими.

Умова 2: Кожен атракціон може бути кінцевою точкою **щонайбільше** K доріжок.

Це завдання складається з 50 підзадач формату **output-only** з **частковим оцінюванням**. Кожне підзадача відповідає певному значенню K , і ви повинні спроектувати парк, який задовольняє всі вищезазначені умови для цього значення K . Ваш бал залежить від кількості атракціонів у вашому рішенні — менша кількість атракціонів дає вищий або однаковий бал..

Деталі реалізації

Існує два способи подання рішення, і ви можете використовувати будь-який з них для кожної підзадачі:

- **Procedure call**
- **Output file**

Щоб надіслати своє рішення через **procedure call**, вам слід реалізувати наступну процедуру.

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

- K : половина кількості типів атракціонів, а також максимальна кількість доріжок, кінцевою точкою яких може бути кожен атракціон.
- Ця процедура викликається рівно один раз для кожної підзадачі.

Процедура повинна повертати пару чисел (T, E) що описують парк розваг. Нехай M буде кількістю доріжок у вашому парку.

- T : масив довжини N , що описує типи атракціонів.
- E : масив довжини M , що описує доріжки. Для кожного $0 \leq j < M$, $E[j] = (U[j], V[j])$ представляє двонаправлену доріжку між різними атракціонами $U[j]$ та $V[j]$.

Щоб надіслати своє рішення через **output file**, створіть та надішліть текстовий файл у такому форматі:

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

Зверніть увагу, що ваше рішення має відповідати наступним обмеженням, щоб вважатися **правильним**:

- $N \leq 2000$
- $0 \leq T[i] < 2K$ для усіх $0 \leq i < N$, і кожен тип має бути призначений принаймні одному атракціону.
- $0 \leq U[j], V[j] < N$ та $U[j] \neq V[j]$ для усіх $0 \leq j < M$.
- Умови 1 та 2 мають бути виконані.

Обмеження

- $1 \leq K \leq 50$

Оцінювання

Існує 50 підзадач, по одній для кожного цілого числа K від 1 до 50. Для підзадачі i ($1 \leq i \leq 50$) значення K дорівнює i .

Кожна підзадача має оцінку S та цільову кількість атракціонів P згідно з таблицею нижче.

Підзадача	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	$12 \cdot K$

Для кожної підзадачі, якщо ваше рішення не описує правильний парк розваг, то оцінка вашого рішення буде 0 (повідомлятиметься як `Output isn't correct` в CMS).

В іншому випадку ваш бал рахується на основі N та параметрів S і P наступним чином:

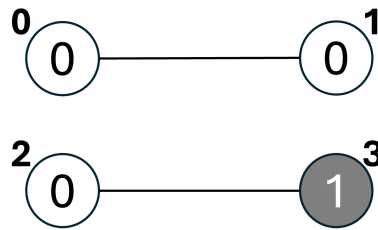
Умова	Бал
$N \leq P$	S
$P < N \leq 2P$	$(0.4 + 0.3 \cdot \frac{2P-N}{P}) \cdot S$
$2P < N \leq 2000$	$(0.1 + 0.3 \cdot \frac{2P}{N}) \cdot S$

Приклад

Розглянемо наступний виклик:

```
construct(1)
```

У цьому прикладі $K = 1$, отже, існує $2K = 2$ типів атракціонів. На рисунку нижче показано правильне рішення з $N = 4$ атракціонами та $M = 2$ пішохідними доріжками. Атракціони 0, 1, 2 мають тип 0, а атракціон 3 має тип 1.



Є дві цікаві трійки:

- Для трійки типу $(0, 1, 0)$ ми можемо вибрати $(a_1, a_2, a_3) = (2, 3, 2)$.
- Для трійки типу $(1, 0, 1)$ ми можемо вибрати $(a_1, a_2, a_3) = (3, 2, 3)$.

Це означає, що Умова 1 виконана.

Процедура може повернути пару $([0, 0, 0, 1], [(0, 1), (2, 3)])$. Зверніть увагу, що надане рішення для цього прикладу може бути неоптимальним для $K = 1$.

Приклад градера

Формат вхідних даних:

K

Формат вихідних даних:

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

Зверніть увагу, що вивід приклада градера відповідає необхідному формату вихідного файлу.