Магичен град

Кметът на Ташкент иска да направи нов дизайн на известния увеселителен парк "Магичен град". Вашата задача е по дадено цяло положително число K, да направите дизайн на парк по следния начин:

- Избирате цяло число N, брой на атракциите. Обозначавате атракциите с целите числа от 0 до $N-1$.
- Добавяте двупосочни пътеки между двойки **различни** атракции. Възможно е да има повече от една пътека между една и съща двойка атракции. **Не е задължително да може да се придвижите между всяка двойка атракции по пътеките.**
- За всяко i, такова че $0 \leq i < N$, определяте, че атракция i е от тип $T[i]$. Има $2K$ типа, номерирани с числата от 0 до $2K-1$. Всеки тип трябва да бъде сложен на поне една атракция. Различните атракции може да са от един и същ тип.

Пазарно проучване е показало, че:

1. Всеки посетител предпочита да посети три атракции, като няма да посети две атракции от един и същи тип последователно.
2. Посетителите не харесват атракции, които са крайни за много пътеки.

За да удовлетвори **всички** потенциални посетители, кметът поставя две условия за вашия дизайн.

Наричаме наредени тройки от **типове** (t_1, t_2, t_3) за $0 \leq t_1, t_2, t_3 < 2K$ **интересни**, ако $t_1 \neq t_2$ и $t_2 \neq t_3$. Забележете, че t_1 може да е равно на t_3. Това означава, че има $2K \cdot (2K-1)^2$ интересни тройки.

Условие 1: За всяка интересна тройка (t_1, t_2, t_3) трябва да съществуват три атракции a_1, a_2, a_3 ($0 \leq a_1, a_2, a_3 < N$), такива че:

- Типовете на a_1, a_2, a_3 съответстват на t_1, t_2, t_3, т.е. $T[a_1] = t_1$, $T[a_2] = t_2$ и $T[a_3] = t_3$.
- Има пътека между a_1 и a_2.
- Има пътека между a_2 и a_3.

Няма значение дали има пътека между a_1 и a_3. Обърнете внимание, че когато $t_1 = t_3$, аткракциите a_1 и a_3 може да са една и съща атракция.

Условие 2: Всяка атракция може да е крайна на **най-много** K пътеки.

Тази задача се състои от 50 **output-only** подзадачи с **частично оценяване**. Всяка подзадача съответства на конкретна стойност на K и трябва да направите дизайн на парк,

удовлетворяващ всички досегашни условия за тази стойност на K. Вашият резултат зависи от броя на атракциите във Вашето решение: по-малко атракции дават по-добър или същия резултат.

Детайли по имплементацията

Има два начина за изпращане на решението Ви, като може да използвате, който и да е от двата за всяка подзадача:

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

За да изпратите решението си чрез **procedure call**, трябва да имплементирате следната функция.

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

- K: половината от броя типове атракции и също максималният брой пътеки, на които може да е край всяка атракция.
- Тази функция се извиква точно веднъж на подзадача.

Функцията трябва да върне двойка (T, E), описващи увеселителния парк. Нека M е броят на пътеките във вашия парк.

- T: масив с дължина N, описващ типовете на атракциите.
- E: масив с дължина M, описващ пътеките. За всяко $0 \leq j < M$, $E[j] = (U[j], V[j])$, описва двупосочна пътека между различни атракции $U[j]$ и $V[j]$.

За да изпратите решението си чрез **output file**, създайте и изпратете текстов файл в следния формат:

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

Забележете, че Вашето решение трябва да удовлетворява следните ограничения, за да се счита за **валидно**:

- $N \leq 2000$
- $0 \leq T[i] < 2K$ за всяко $0 \leq i < N$ и всеки тип трябва да бъде сложен на поне една атракция.
- $0 \leq U[j], V[j] < N$ и $U[j] \neq V[j]$ за всяко $0 \leq j < M$.
- Условия 1 и 2 трябва да са удовлетворени.

Ограничения

- $1 \leq K \leq 50$

Оценяване

Има 50 подзадачи, по една за всяко цяло число K от 1 до 50. За подзадача i ($1 \leq i \leq 50$), стойността на K е i.

Всяка подзадача има резултат S и целеви брой на атракциите P спрямо таблицата по-долу.

Подзадачи	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	$12 \cdot K$

За всяка подзадача, ако Вашето решение не описва валиден увеселителен парк, то резултатът на решението Ви ще е 0 (отчетено като `Output isn't correct` в CMS).

В противен случай, Вашият резултат ще бъде изчислен спрямо N и параметрите S и P по следния начин:

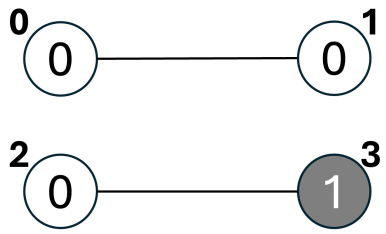
Случай	Резултат
$N \leq P$	S
$P < N \leq 2P$	$(0.4 + 0.3 \cdot \frac{2P-N}{P}) \cdot S$
$2P < N \leq 2000$	$(0.1 + 0.3 \cdot \frac{2P}{N}) \cdot S$

Пример

Нека разгледаме следното извикване:

```
construct(1)
```

В този пример $K = 1$, така че има $2K = 2$ типа атракции. Следната фигура по-долу показва валидно решение с $N = 4$ атракции и $M = 2$ пътеки. Атракции $0, 1, 2$ са от тип 0 и атракция 3 е от тип 1.



Имаме две интересни тройки:

- За тройката от типове $(0, 1, 0)$, можем да изберем $(a_1, a_2, a_3) = (2, 3, 2)$.
- За тройката от типове $(1, 0, 1)$, можем да изберем $(a_1, a_2, a_3) = (3, 2, 3)$.

Това означава, че Условие 1 е удовлетворено.

Функцията може да върне двойката $([0, 0, 0, 1], [(0, 1), (2, 3)])$. Забележете, че това решение на този пример може да не е оптималното за $K = 1$.

Локален грейдър

Формат на входа:

```
K
```

Формат на изхода:

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

Забележете, че изходът на локалния грейдър съответства на нужния формат на output file.