

«Волшебный город»

Мэр Ташкента хочет перепроектировать знаменитый парк развлечений «Волшебный город». Вам дано положительное целое число K , и ваша задача — спроектировать парк следующим образом:

- Выберите целое число N , обозначающее количество аттракционов. Аттракционы пронумерованы от 0 до $N - 1$.
- Добавьте двусторонние пешеходные дорожки между парами **разных** аттракционов. Разрешается добавлять несколько дорожек между одной и той же парой аттракционов. **Не требуется**, чтобы между каждой парой аттракционов можно было переместиться по пешеходным дорожкам.
- Для каждого i , такого, что $0 \leq i < N$, присвойте аттракциону i тип $T[i]$. Имеется $2K$ доступных типов аттракционов, пронумерованных от 0 до $2K - 1$. Каждый тип должен быть присвоен как минимум одному аттракциону. Разные аттракционы могут иметь один и тот же тип.

Исследование рынка показало, что:

1. Каждый посетитель собирается посетить три аттракциона, не посещая два аттракциона одного и того же типа подряд.
2. Посетители не любят аттракционы, из которых выходит много дорожек.

Чтобы удовлетворить **всех** потенциальных посетителей мэр выдвигает два условия к вашему проекту.

Мы называем упорядоченную тройку **типов** (t_1, t_2, t_3) для $0 \leq t_1, t_2, t_3 < 2K$ **интересной**, если $t_1 \neq t_2$ и $t_2 \neq t_3$. Заметим, что t_1 может быть равна t_3 . Таким образом, существует $2K \cdot (2K - 1)^2$ интересных троек.

Условие 1: Для каждой интересной тройки (t_1, t_2, t_3) должны существовать три аттракциона a_1, a_2, a_3 (при условии, что $0 \leq a_1, a_2, a_3 < N$), такие, что:

- Типы a_1, a_2, a_3 совпадают с t_1, t_2, t_3 . То есть $T[a_1] = t_1, T[a_2] = t_2$ и $T[a_3] = t_3$.
- Между a_1 и a_2 имеется пешеходная дорожка.
- Между a_2 и a_3 имеется пешеходная дорожка.

Не имеет значения, имеется ли пешеходная дорожка между a_1 и a_3 . Также обратите внимание, что при условии $t_1 = t_3$ аттракционы a_1 и a_3 могут совпадать.

Условие 2: Каждый аттракцион может быть соединен **не более чем с K** пешеходными дорожками.

Эта задача состоит из 50 **output-only подзадач**, с **частичным оцениванием**. Каждая подзадача соответствует конкретному значению K , и вам необходимо спроектировать парк, удовлетворяющий всем вышеуказанным условиям для данного значения K . Ваш балл зависит от количества аттракционов в вашем решении — меньшее количество аттракционов может привести к более высокому баллу.

Детали реализации

Существует два способа сдачи решения, и вы можете использовать любой из них для каждой подзадачи:

- **Вызов функции**
- **Вывод в файл**

Чтобы сдать решение посредством **вызова функции**, вам необходимо реализовать следующую функцию.

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

- K : половина количества типов аттракционов, а также максимально допустимое количество дорожек, исходящих из каждого аттракциона.
- Эта функция вызывается ровно один раз для каждой подзадачи.

Функция должна возвращать пару (T, E) , описывающую парк развлечений. Пусть M — количество дорожек в вашем парке.

- T : массив длины N , описывающий типы аттракционов.
- E : массив длины M , описывающий пешеходные дорожки. Для каждого $0 \leq j < M$ элемент $E[j] = (U[j], V[j])$ представляет двунаправленную пешеходную дорожку между двумя различными аттракционами $U[j]$ и $V[j]$.

Чтобы отправить своё решение в виде **файла вывода**, создайте и отправьте текстовый файл в следующем формате:

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

Обратите внимание, что ваше решение должно удовлетворять следующим ограничениям, чтобы считаться **допустимым**:

- $N \leq 2000$
- $0 \leq T[i] < 2K$ для каждого $0 \leq i < N$, и существует хотя бы один аттракцион каждого типа.

- $0 \leq U[j], V[j] < N$ и $U[j] \neq V[j]$ для каждого $0 \leq j < M$.
- Должны быть выполнены условия 1 и 2.

Ограничения

- $1 \leq K \leq 50$

Оценка

Существует 50 подзадач, по одной для каждого целого числа K из диапазона от 1 до 50. Для подзадачи i ($1 \leq i \leq 50$) значение K равно i .

Каждая подзадача имеет оценку S и целевое число аттракционов в парке P в соответствии с приведённой ниже таблицей.

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	$12 \cdot K$

Для каждой подзадачи, если ваше решение не описывает допустимый парк развлечений, то оценка вашего решения будет равна 0 (в CMS отображается как «Output isn't correct»).

В противном случае ваша оценка рассчитывается на основе N и параметров S и P следующим образом:

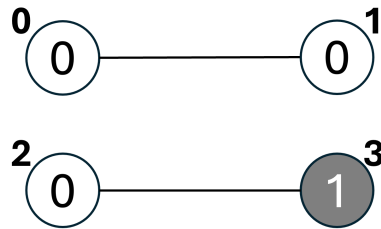
Условие	Оценка
$N \leq P$	S
$P < N \leq 2P$	$(0.4 + 0.3 \cdot \frac{2P-N}{P}) \cdot S$
$2P < N \leq 2000$	$(0.1 + 0.3 \cdot \frac{2P}{N}) \cdot S$

Пример

Рассмотрим следующий запрос.

```
construct(1)
```

В этом примере $K = 1$, поэтому существует $2K = 2$ типов аттракционов. На рисунке ниже показано допустимое решение с $N = 4$ аттракционами и $M = 2$ пешеходными дорожками. Аттракционы 0, 1, 2 относятся к типу 0, а аттракцион 3 — к типу 1.



Есть две интересные тройки:

- Для тройки типов $(0, 1, 0)$ можно выбрать $(a_1, a_2, a_3) = (2, 3, 2)$.
- Для тройки типов $(1, 0, 1)$ можно выбрать $(a_1, a_2, a_3) = (3, 2, 3)$.

Это означает, что условие 1 выполняется.

Функция может возвращать пару $([0, 0, 0, 1], [(0, 1), (2, 3)])$. Обратите внимание, что приведённое решение для этого примера может оказаться неоптимальным для задачи $K = 1$.

Пример грейдера

Формат входных данных:

```
K
```

Формат выходных данных:

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

Обратите внимание, что вывод примера грейдера соответствует требуемому формату выходного файла.