

魔幻之城

塔什干市的市长想要重新设计著名的魔幻之城游乐园。给定一个正整数 K ，你的任务是设计一个游乐园如下：

- 选择一个整数 N 作为景点的数量。景点用 0 到 $N - 1$ 的整数进行编号。
- 添加若干条双向步道，每条步道连接两个**不同**的景点。在同一对景点之间可以有不止一条步道。**不要求通过这些步道就能在任意两个景点之间通行。**
- 对于每个满足 $0 \leq i < N$ 的 i ，为景点 i 分配一个类型 $T[i]$ 。共有 $2K$ 种类型，编号为从 0 到 $2K - 1$ 。每种类型必须至少分配给一个景点。不同景点可以有相同类型。

市场调研表明：

1. 每位游客希望游览三个景点，但不要连续游览两个相同类型的景点。
2. 如果一个景点和很多步道相连，游客往往不喜欢。

为了满足**所有**潜在游客的需求，市长对你的设计提出了两个要求。

对于满足 $0 \leq t_1, t_2, t_3 < 2K$ 、由**类型**构成的有序三元组 (t_1, t_2, t_3) ，若满足 $t_1 \neq t_2$ 且 $t_2 \neq t_3$ ，我们称之为**有趣**的三元组。注意 t_1 可能等于 t_3 。因此，共有 $2K \cdot (2K - 1)^2$ 个有趣的三元组。

条件 1： 对于每个有趣的三元组 (t_1, t_2, t_3) ，必须存在三个景点 a_1, a_2, a_3 （有 $0 \leq a_1, a_2, a_3 < N$ ），满足：

- a_1, a_2, a_3 的类型与 t_1, t_2, t_3 匹配。即 $T[a_1] = t_1$ ， $T[a_2] = t_2$ ，且 $T[a_3] = t_3$ 。
- a_1 与 a_2 之间有一条步道。
- a_2 与 a_3 之间有一条步道。

a_1 与 a_3 之间是否存在步道无关紧要。另外请注意，当 $t_1 = t_3$ 时， a_1 和 a_3 可以是同一个景点。

条件 2： 每个景点**最多和 K 条**步道相连。

本任务包含 50 个**提交答案型**子任务，并且有**部分分**。每个子任务对应一个特定的 K 值，你必须为每个 K 值设计一个满足上述所有条件的游乐园。你的得分取决于你答案中的景点数量：景点越少，得分越高或持平。

实现细节

有两种提交答案的方式，你可以为每个子任务选择其中一种：

- 函数调用
- 输出文件

通过**函数调用**提交答案时，你要实现以下函数：

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

- K ：类型总数的一半，同时也是每个景点允许连接的最大步道数。
- 对每个子任务，该函数恰好被调用一次。

该函数应返回一个二元组 (T, E) ，以给出游乐园的设计。假定你的游乐园中步道的数量为 M 。

- T ：长度为 N 的数组，给出景点的类型。
- E ：长度为 M 的数组，给出所有步道。对于每个满足 $0 \leq j < M$ 的 j ， $E[j] = (U[j], V[j])$ 表示不同景点 $U[j]$ 与 $V[j]$ 之间的一条双向步道。

通过**输出文件**提交答案时，你要创建并提交一个如下格式的文本文件：

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

注意，你的答案必须满足以下**约束条件**才能被视为**符合要求**：

- $N \leq 2000$
- 对于每个 $0 \leq i < N$ ，都有 $0 \leq T[i] < 2K$ ，而且每个类型应该分配给至少一个景点。
- 对于每个 $0 \leq j < M$ ，都有 $0 \leq U[j], V[j] < N$ 且 $U[j] \neq V[j]$ 。
- 必须满足条件 1 和条件 2。

约束条件

- $1 \leq K \leq 50$

评分

共有 50 个子任务，分别对应从 1 到 50 的整数 K 。对于子任务 i ($1 \leq i \leq 50$)， K 的值为 i 。

每个子任务都有一个分值 S 和景点的目标数量 P ，如下表所示：

子任务	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	$12 \cdot K$

对于每个子任务，如果你的答案未给出一个符合要求的游乐园，则你的得分为 0（在 CMS 中报告为 **Output isn't correct**）。

否则，你的得分将根据下表由 N 以及参数 S 和 P 计算：

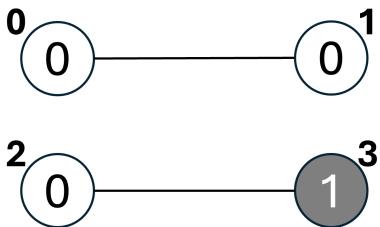
条件	分数
$N \leq P$	S
$P < N \leq 2P$	$(0.4 + 0.3 \cdot \frac{2P-N}{P}) \cdot S$
$2P < N \leq 2000$	$(0.1 + 0.3 \cdot \frac{2P}{N}) \cdot S$

例子

考虑以下调用：

```
construct(1)
```

在这个例子中， $K = 1$ ，因此有 $2K = 2$ 种景点类型。 下图展示了一个包含 $N = 4$ 个景点和 $M = 2$ 条步道的符合要求的答案。 景点 0,1,2 的类型为 0，景点 3 的类型为 1。



这里共有两个有趣的三元组：

- 对于类型三元组 $(0, 1, 0)$ ，我们可以选择 $(a_1, a_2, a_3) = (2, 3, 2)$ 。
- 对于类型三元组 $(1, 0, 1)$ ，我们可以选择 $(a_1, a_2, a_3) = (3, 2, 3)$ 。

这表明条件 1 已满足。

该函数可以返回二元组 $([0, 0, 0, 1], [(0, 1), (2, 3)])$ 。 请注意，此处提供的答案可能不是 $K = 1$ 时的最优解。

评测程序示例

输入格式：

K

输出格式：

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

请注意，评测程序示例的输出结果满足输出文件的格式要求。