Magic City

The mayor of Tashkent wants to redesign the famous Magic City amusement park. You are given a positive integer K and your task is to design a park as follows:

- Choose an integer N, the number of attractions. The attractions are labeled with integers from 0 to $N-1$.
- Add bidirectional walkways, each between a pair of **distinct** attractions. It is allowed to have more than one walkway between the same pair of attractions. **It is not required to be able to travel between every pair of attractions using the walkways.**
- For each i such that $0 \leq i < N$, assign attraction i a type $T[i]$. There are $2K$ types numbered from 0 to $2K-1$. Every type should be assigned to at least one attraction. Different attractions may have the same type.

Market research revealed that:

1. Each visitor prefers to visit three attractions without visiting two attractions of the same type consecutively.
2. Visitors dislike attractions that are an endpoint of many walkways.

In order to satisfy **all** potential visitors, the mayor imposes two conditions on your design.

We call an ordered triple of **types** (t_1, t_2, t_3) for $0 \leq t_1, t_2, t_3 < 2K$ **interesting** if $t_1 \neq t_2$ and $t_2 \neq t_3$. Note that t_1 might be equal to t_3. Thus, there are $2K \cdot (2K-1)^2$ interesting triples.

Condition 1: For each interesting triple (t_1, t_2, t_3), there must exist three attractions a_1, a_2, a_3 (with $0 \leq a_1, a_2, a_3 < N$) such that:

- The types of a_1, a_2, a_3 match t_1, t_2, t_3. That is, $T[a_1] = t_1$, $T[a_2] = t_2$, and $T[a_3] = t_3$.
- There is a walkway between a_1 and a_2.
- There is a walkway between a_2 and a_3.

It is irrelevant whether or not there is a walkway between a_1 and a_3. Also note that when $t_1 = t_3$, the attractions a_1 and a_3 may be the same.

Condition 2: Each attraction can be an endpoint of **at most** K walkways.

This task consists of 50 **output-only** subtasks with **partial scoring**. Each subtask corresponds to a specific value of K, and you must design a park that satisfies all above conditions for that value of K. Your score depends on the number of attractions in your solution: fewer attractions yield a higher or equal score.

Implementation Details

There are two methods to submit your solution, and you may use either one for each subtask:

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

To submit your solution via a **procedure call**, you should implement the following procedure:

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

- K: half the number of attraction types, and also the maximum number of walkways that each attraction can be an endpoint of.
- This procedure is called exactly once for each subtask.

The procedure should return a pair (T, E) describing an amusement park. Let M be the number of walkways in your park.

- T: an array of length N describing the types of attractions.
- E: an array of length M describing the walkways. For each $0 \leq j < M$, $E[j] = (U[j], V[j])$ represents a bidirectional walkway between distinct attractions $U[j]$ and $V[j]$.

To submit your solution via an **output file**, create and submit a text file in the following format:

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

Note that your solution must satisfy the following constraints to be considered **valid**:

- $N \leq 2000$
- $0 \leq T[i] < 2K$ for each $0 \leq i < N$, and every type should be assigned to at least one attraction.
- $0 \leq U[j], V[j] < N$ and $U[j] \neq V[j]$ for each $0 \leq j < M$.
- Conditions 1 and 2 must be met.

Constraints

- $1 \leq K \leq 50$

Scoring

There are 50 subtasks, one for each integer K from 1 to 50. For Subtask i ($1 \leq i \leq 50$), the value of K is i.

Each subtask has a score S and a target number of attractions P according to the table below.

Subtasks	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	$12 \cdot K$

For each subtask, if your solution does not describe a valid amusement park, then the score of your solution will be 0 (reported as `Output isn't correct` in CMS).

Otherwise, your score is calculated based on N and the parameters S and P as follows:

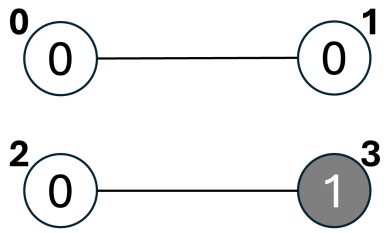
Condition	Score
$N \leq P$	S
$P < N \leq 2P$	$(0.4 + 0.3 \cdot \frac{2P-N}{P}) \cdot S$
$2P < N \leq 2000$	$(0.1 + 0.3 \cdot \frac{2P}{N}) \cdot S$

Example

Consider the following call:

```
construct(1)
```

In this example, $K = 1$, so there are $2K = 2$ types of attractions. The figure below shows a valid solution with $N = 4$ attractions and $M = 2$ walkways. Attractions $0, 1, 2$ are of type 0, and attraction 3 is of type 1.



There are two interesting triples:

- For the type triple $(0, 1, 0)$, we can choose $(a_1, a_2, a_3) = (2, 3, 2)$.
- For the type triple $(1, 0, 1)$, we can choose $(a_1, a_2, a_3) = (3, 2, 3)$.

This means that Condition 1 is satisfied.

The procedure can return the pair $([0, 0, 0, 1], [(0, 1), (2, 3)])$. Note that the provided solution for this example might not be optimal for $K = 1$.

Sample Grader

Input format:

```
K
```

Output format:

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

Note that the output of the sample grader matches the required format for the output file.