Boardgame Expo

Every year, a big Boardgame Expo takes place in Cluj-Napoca, showcasing a wide selection of new games. The main attraction this year is a game called BoardOina.

There are n players lined up in a queue, waiting to try out the game. Players are numbered from 0 to $n-1$ in their order in the queue. Player 0 is at the front of the queue and player $n-1$ is at the back.

There are m *distinct* **friendship relations** between m pairs of players in the queue. Specifically, for each i from 0 to $m-1$, inclusive, player $x[i]$ and player $y[i]$ are friends, where $0 \le x[i] < y[i] < n$. Friendship relations are symmetric.

Consider a sequence of k *consecutive* players in the queue starting at player s (for any s and k such that $0 \le s < n$ and $1 \le k \le n-s$). This sequence of players forms a **friend group** of size k if for all pairs of two players, they are connected by a sequence of friendship relations within that friend group. Specifically, players $s, s+1, \ldots, s+k-1$ form a friend group of size k if, for each u and v such that $s \le u < v < s+k$, there exists a sequence of players $p[0], \ldots, p[l-1]$ such that:

- $l \ge 2$;
- $s \le p[j] < s+k$ for each j from 0 to $l-1$, inclusive;
- $p[0] = u$ and $p[l-1] = v$;
- players $p[j]$ and $p[j+1]$ are friends for each j from 0 to $l-2$, inclusive.

Note that in the case of $k=1$, player s alone forms a friend group of size 1.

BoardOina can be played by any number of players. However, to make the game more successful, the organizers only let friend groups play it.

Only one group can play at a time. For each game, a friend group starting at the player at the front of the queue is formed, and starts playing the game. The players in this friend group are removed from the queue. This process is repeated until the queue becomes empty. Formally, we say that the queue **can be partitioned into** g **friend groups** if there exists an array of group sizes, $K = [K[0], K[1], \ldots, K[g-1]]$, such that each of the following conditions holds.

- $g > 0$ and $K[j] > 0$ (for each j such that $0 \le j < g$);
- $K[0] + K[1] + \ldots + K[g-1] = n$;
- for each j between 0 and $g-1$, inclusive, players $s[j], s[j]+1, \ldots, s[j]+K[j]-1$ form a friend group of size $K[j]$, where $s[0] = 0$ and otherwise

$s[j] = K[0] + K[1] + \ldots + K[j-1]$.

The organizers want to *minimize* the number of friend groups that play the game. That is, they want to partition the queue into g friend groups such that it is not possible to partition the queue into $g-1$ (or less) friend groups.

Your task is to find a partitioning of the queue into a minimum number of friend groups, and report the array of group sizes.

Implementation Details

You should implement the following procedure.

```
std::vector<int> partition_players(int n, int m, std::vector<int> X,
      std::vector<int> Y)
```

- n: the number of players in the queue.
- m: the number of friendship relations.
- x, y: arrays of length m describing friendship relations.
- This procedure should return an array of group sizes, representing a partition of the player queue into a minimum number of friend groups.
- This procedure is called exactly once for each test case.

Constraints

- $2 \leq n \leq 100\,000$
- $0 \leq m \leq 200\,000$
- $0 \leq x[i] < y[i] < n$ (for each i such that $0 \leq i < m$)
- Friendship relations are distinct. In other words, $x[i] \neq x[j]$ or $y[i] \neq y[j]$ (for each i and j such that $0 \leq i < j < m$).
- If there are multiple solutions with minimum number of groups, you can return any valid solution.

Subtasks

1. (5 points) $y[i] = x[i] + 1$ for each i from 0 to $m-1$, inclusive.
2. (7 points) $y[i] \leq x[i] + 2$ for each i from 0 to $m-1$, inclusive.
3. (6 points) $n \leq 300$ and $m \leq 600$
4. (15 points) $n \leq 2\,000$ and $m \leq 4\,000$
5. (34 points) There are no friendship relations which are *cyclic*. That is, for any sequence of *distinct* players $p[0], p[1], \ldots, p[l-1]$, such that $l \geq 3$ and for each $0 \leq j < l-1$ players $p[j]$ and $p[j+1]$ are friends, players $p[0]$ and $p[l-1]$ are **not** friends.
6. (33 points) No additional constraints.

Examples

Example 1

Consider the following call:

```
partition_players(5, 3, {0, 1, 3}, {1, 4, 4})
```

In this example, players 0 and 1, players 1 and 4, and players 3 and 4 are friends.

Player 2 has no friends in the queue, hence there must be a friend group formed by player 2 alone, which means that the minimum number of friend groups is $g = 3$. On the other hand, players 0 and 1, as well as players 3 and 4 can form a friend group of size 2.

Therefore, the queue can be partitioned into 3 friend groups of sizes 2, 1 and 2, so the procedure may return $[2, 1, 2]$.

Example 2

Consider the following call:

```
partition_players(7, 6, {0, 4, 2, 1, 2, 3}, {1, 5, 4, 5, 5, 6})
```

In this example, players 0 and 1, players 4 and 5, players 2 and 4, players 1 and 5, players 2 and 5 and players 3 and 6 are friends.

The only friend of player 3 is player 6, so any friend group containing player 3 is either

- a friend group of size 1 containing player 3 alone, or
- a friend group containing both player 3 and player 6.

A friend group in the second case must also contain players 4 and 5. This is not possible as the only friend of player 6 is player 3, so player 3 is not connected to players 4 and 5 by a sequence of friendship relations.

Therefore, player 3 must be placed in a friend group of size 1. Similarly, player 6 must also be placed in a friend group of size 1, therefore the number of friend groups in a partition is at least 4.

Players 0, 1 and 2 do not form a friend group of size 3, as neither player 0 or player 1 is connected to player 2 by a sequence of friendship relations within the group. That would have not been the case if 5 was in the group, but since 3 and 4 will definitely be in different groups, that will never happen. That is, the number of friend groups in a partition is at least 5.

On the other hand, players 0 and 1, and players 4 and 5 form two friend groups of size 2. Therefore, the queue can be partitioned into 5 friend groups of sizes 2, 1, 1, 2 and 1. The procedure may return $[2, 1, 1, 2, 1]$.

Sample Grader

The sample grader reads the input in the following format:

- line 1: $n\ m$
- line $2 + i$ ($0 \le i < m$): $x[i]\ y[i]$

Let the elements of the array returned by `partition_players` be $K[0], K[1], \ldots, K[g-1]$ for some nonnegative g. The output of the sample grader is in the following format:

- line 1: g
- line 2: $K[0]\ K[1]\ \ldots\ K[g-1]$