# D. Deputies

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

The Trinitarian kingdom has exactly $n = 3k$ cities. All of them are located on the shores of river Trissisipi, which flows through the whole kingdom. Some of the cities are located on one side of the river, and all the rest are on the other side.

Some cities are connected by bridges built between them. Each bridge connects two cities that are located on the opposite sides of the river. Between any two cities exists no more than one bridge.

The recently inaugurated King Tristan the Third is busy distributing his deputies among cities. In total there are $k$ deputies and the king wants to commission each of them to control exactly three cities. However, no deputy can be entrusted to manage the cities, which are connected by a bridge — the deputy can set a too high fee for travelling over the bridge to benefit his pocket, which is bad for the reputation of the king.

Help King Tristan the Third distribute the deputies between the cities, if it is possible.

## Input

The first line contains two integers $n$ and $m$ — the number of cities and bridges ($3 \leq n < 10^5$, $n = 3k$, $0 \leq m \leq 10^5$). Next $m$ lines describe the bridges. The $i$-th line contains two integers $a_i$ and $b_i$ — the numbers of cities that are connected by the $i$-th bridge ($1 \leq a_i, b_i \leq n$, $a_i \neq b_i$, $1 \leq i \leq m$).

It is guaranteed that no bridge connects a city with itself and that any two cities are connected with no more than one bridge.

## Output

If distributing the deputies in the required manner is impossible, print in a single line "NO" (without the quotes).

Otherwise, in the first line print "YES" (without the quotes), and in the second line print which deputy should be put in charge of each city. The $i$-th number should represent the number of the deputy (from $1$ to $k$), who should be in charge of city numbered $i$-th in the input — overall there should be $n$ numbers.

If there are multiple solutions, print any of them.

## Examples

### input

```
6 6
1 2
4 1
3 5
6 5
2 6
4 6
```

### output

```
YES
1 2 1 2 2 1
```

### input

```
3 1
1 2
```

### output

```
NO
```