Degree

Problem

Submissions

Discussion Coming Soon

You are given a graph with N nodes and M bidirectional (undirected) edges. You have to find the degree of each node in this graph and print it.

Note: the degree of a node is the number of edges coming to it or going out from it. If the edges are undirected or bidirectional, we think of it as a single edge.

Input Format

First line N and M, the number of nodes in the graph and the number of edges respectively.

M lines, each containing u and v, meaning there is an undirected edge connecting node u and node v.

Output Format

Print N integers, the degree of node i

Constraints

$1 \leq N \leq 10^4$

$1 \leq M \leq \frac{N \times (N-1)}{2}$

$1 \leq u, v \leq N$

Sample Input 1

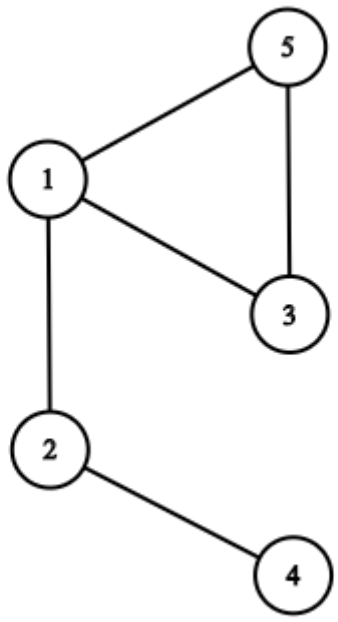
```
5 5
1 2
1 5
4 2
3 5
1 3
```

Sample Output 1

```
3 2 2 1 2
```

Explanation 1

This is the graph given in the sample 1.



Sample Input 2

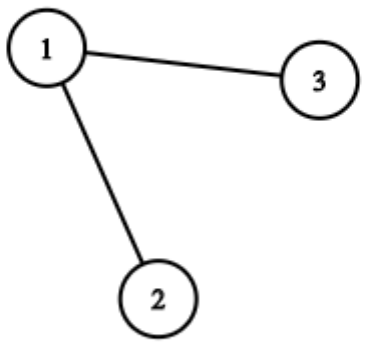
```
3 2
1 2
3 1
```

Sample Output 2

```
2 1 1
```

Explanation 2

This is the graph given in the sample 2, since node 1 has 2 neighbours degree of node 1 is 2. Since node 2 and node 3 has only one neighbour, their degree is 1.



C++ (GCC 9.2.0) Bright **Memory Limit (kB) :** 256000 **Time Limit (s) :** 1

```cpp
#include <bits/stdc++.h>

using namespace std;
#define int long long

int32_t main(){
    int m, n;
    cin >> n >> m;
    vector<int> adj[n + 1];
    for(int i = 0; i < m; i++){
        int x, y;
        cin >> x >> y;
        if (x <=n && y <=n){
            adj[x].push_back(y);
            adj[y].push_back(x);
        }

    }
    for(int i = 1; i <=n; i++)
        cout << adj[i].size() << " ";
```

```
21 }
```

Upload File

Test against custom test case

Run Code

Submit

Sample Test Case 0

Sample Test Case 1

Accepted

Input(stdin)

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

Output(stdin)

```
3 2 2 1 2
```

Expected Output

```
3 2 2 1 2
```