Problem statement:

Breadth-First Search in an Unweighted Graph

The provided text describes Breadth-First Search (BFS), a fundamental graph traversal algorithm used to explore the vertices of a graph layer by layer. In an unweighted graph, BFS is commonly used to compute the shortest path (in terms of number of edges) from a source node to all other reachable nodes.

BFS operates by visiting all neighbors of a node before moving on to the next level of neighbors. It uses a queue to maintain the list of vertices to be processed next, and a visited set or array to ensure each node is visited only once.

The algorithm guarantees that the first time a node is reached, the shortest path to that node has been found. This makes BFS ideal for applications such as shortest path finding in unweighted graphs, level-order traversal in trees, and finding connected components.

Problem Statement

In this problem, you are given an undirected, unweighted graph represented by a set of nodes and edges. Your task is to implement BFS starting from a given source node and print the shortest distance from the source to each reachable node.

Input Format

- The first line contains two integers `n` and `m`, the number of nodes and edges in the graph.
- The next `m` lines each contain two integers `u` and `v`, representing an undirected edge between nodes `u` and `v`.
- The last line contains a single integer `s`, the source node from which BFS should begin.

Constraints

- $1 \leq n \leq 10^5$
- $1 \leq m \leq 2 \times 10^5$
- $1 \leq u, v, s \leq n$
- All edges are undirected.

Output Format

Print the shortest distance from the source node to every other reachable node, one per line. If a node is unreachable from the source, print `-1`.

Sample Input

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

Sample Output

```
0
1
1
2
2
3
```

Explanation

Starting from node 1, BFS visits nodes 2 and 3 in the first layer, then nodes 4 and 5 in the second, and finally node 6 in the third. The printed output shows the shortest distance from node 1 to each node (0 for itself, 1 for direct neighbors, and so on).