Pepcoder And Reversing(0/1 BFS)

You are given 2 integers N and M , N is the number of vertices, M is the number of edges. You'll also be given ai and bi where ai and bi represents an edge from a vertex ai to a vertex bi.
You have to find the minimum number of edges you have to reverse in order to have atleast one path from vertex 1 to N, where the vertices are numbered from 1 to N.

First line contains two space separated integers,N and M. Then M lines follow, each line has 2 space separated integers ai and bi.

```python
import collections


def introTo01BFS(edges, V):
    graph = collections.defaultdict(list)

    for u, v in edges:
        graph[u].append([v, 0])
        graph[v].append([u, 1])
    # print(graph)
    queue = collections.deque()
    queue.appendleft([0, 1])
    visited = [False] * (V + 1)

    while len(queue) > 0:
        wt, node = queue.popleft()
        if node == V:
            return wt
        visited[node] = True
        for nbr, wt in graph[node]:
            if visited[nbr] is True:
                continue
            if wt == 0:
                queue.appendleft([wt + 0, nbr])
            elif wt == 1:
                queue.append([wt + 1, nbr])
    return -1


edges = [[1, 2], [3, 2], [6, 2], [3, 4], [7, 4], [7, 5], [5, 6]]
print(introTo01BFS(edges, 7))
```