

```
1  bool validPath(int n, int** edges, int edgesSize, int* edgesColSize, int source, int destination) {
2      if (source == destination) return true;
3
4      int* graphSize = calloc(n, sizeof(int));
5      int** graph = malloc(n * sizeof(int*));
6
7      for (int i = 0; i < edgesSize; i++) {
8          int u = edges[i][0];
9          int v = edges[i][1];
10         graphSize[u]++;
11         graphSize[v]++;
12     }
13
14     for (int i = 0; i < n; i++) {
15         graph[i] = malloc(graphSize[i] * sizeof(int));
16         graphSize[i] = 0;
17     }
18
19     for (int i = 0; i < edgesSize; i++) {
20         int u = edges[i][0];
21         int v = edges[i][1];
22         graph[u][graphSize[u]++] = v;
23         graph[v][graphSize[v]++] = u;
24     }
25
26     int* queue = malloc(n * sizeof(int));
27     int front = 0, back = 0;
28
29     bool* visited = calloc(n, sizeof(bool));
30
31     queue[back++] = source;
32     visited[source] = true;
33
34     while (front < back) {
35         int cur = queue[front++];
36
37         for (int i = 0; i < graphSize[cur]; i++) {
38             int next = graph[cur][i];
39
40             if (!visited[next]) {
41                 if (next == destination) {
42                     for (int j = 0; j < n; j++) free(graph[j]);
43                     return true;
44                 }
45                 visited[next] = true;
46                 queue[back++] = next;
47             }
48         }
49     }
50
51     free(graph);
52     free(queue);
53     free(graphSize);
54
55     return false;
56 }
```

```
43     free(graph);
44     free(graphSize);
45     free(queue);
46     free(visited);
47     return true;
48 }
49
50     visited[next] = true;
51     queue[back++] = next;
52 }
53 }
54 }
55 for (int i = 0; i < n; i++) free(graph[i]);
56 free(graph);
57 free(graphSize);
58 free(queue);
59 free(visited);
60
61 return false;
62 }
```

Accepted

Runtime: 0 ms

Case 1

Case 2

Input

```
n =
```

```
3
```

```
edges =
```

```
[[0,1],[1,2],[2,0]]
```

```
source =
```

```
0
```

```
destination =
```

```
2
```

Output

```
true
```

Expected

```
true
```



Contribute a testcase

Accepted Runtime: 0 ms

Case 1

Case 2

Input

n =

6

edges =

`[[0,1],[0,2],[3,5],[5,4],[4,3]]`

source =

0

destination =

5

Output

`false`

Expected

`false`

 Contribute a testcase