C Auto

```c
bool validPath(int n, int** edges, int edgesSize, int* edgesColSize, int source, int destination) {
    if (source == destination) return true;

    int* graphSize = calloc(n, sizeof(int));
    int** graph = malloc(n * sizeof(int*));

    for (int i = 0; i < edgesSize; i++) {
        int u = edges[i][0];
        int v = edges[i][1];
        graphSize[u]++;
        graphSize[v]++;
    }

    for (int i = 0; i < n; i++) {
        graph[i] = malloc(graphSize[i] * sizeof(int));
        graphSize[i] = 0;
    }

    for (int i = 0; i < edgesSize; i++) {
        int u = edges[i][0];
        int v = edges[i][1];
        graph[u][graphSize[u]++] = v;
        graph[v][graphSize[v]++] = u;
    }

    int* queue = malloc(n * sizeof(int));
    int front = 0, back = 0;

    bool* visited = calloc(n, sizeof(bool));

    queue[back++] = source;
    visited[source] = true;

    while (front < back) {
        int cur = queue[front++];

        for (int i = 0; i < graphSize[cur]; i++) {
            int next = graph[cur][i];

            if (!visited[next]) {
                if (next == destination) {
                    for (int j = 0; j < n; j++) free(graph[j]);
```

```
                free(graph);
                free(graphSize);
                free(queue);
                free(visited);
                return true;
            }

            visited[next] = true;
            queue[back++] = next;
        }
    }
}
for (int i = 0; i < n; i++) free(graph[i]);
free(graph);
free(graphSize);
free(queue);
free(visited);

return false;
}
```

ved

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