***To Implement the methods of Graph like BFS DFS etc***

*Name :- SUJAL NIMJE*

*Roll no :- 64*

*Subject :- DS, practical no : - 7*

*Code :-*

#include <stdio.h>

#include <stdlib.h>

#include "../queue/circular.h"

int size = 3;

int graph[3][3] = {

    {0, 1, 1},

    {0, 0, 1},

    {1, 1, 0}};

int vis[3];

void dfs(int curr)

{

    printf("%d ", curr);

    vis[curr] = 1;

    for (int j = 0; j < size; j++)

    {

        int edge = j;

        if (vis[edge] == 0 && graph[curr][j] == 1)

        {

            dfs(edge);

        }

    }

}

void isUniversalNode(int node)

{

    for (int i = 0; i < size; i++)

    {

        if ((graph[node][i] == 0 || graph[i][node] == 0) && i != node)

        {

            printf("%d is not a universal node\n", node);

            return;

        }

    }

    printf("%d is a universal node \n");

}

void outerDegOfNode(int node)

{

    int count = 0;

        for (int j = 0; j < size; j++)

        {

            if (graph[node][j] == 1)

            {

                count++;

            }

        }

        printf("outer deg of %d node is : %d\n", node, count);

}

void bfs(int start)

{

    struct Queue \*q = (struct Queue \*)malloc(sizeof(struct Queue));

    allocateMemory(q, 10);

    enqueue(q, start);

    while (!isEmpty(q))

    {

        int curr = dqueue(q);

        if (vis[curr] == 0)

        {

            printf("%d ", curr);

            vis[curr] = 1;

            for (int i = 0; i < size; i++)

            {

                if (graph[curr][i] == 1)

                {

                    enqueue(q, i);

                }

            }

        }

    }

}

void main()

{

    for (int i = 0; i < size; i++)

    {

        vis[i] = 0;

    }

    printf("the DFS sequence is : ");

    dfs(0);

    for (int i = 0; i < size; i++)

    {

        vis[i] = 0;

    }

    printf("\n");

    printf("the BFS sequence is : ");

    bfs(0);

    printf("\n");

    outerDegOfNode(0);

    isUniversalNode(0);

}

*Output*

