//DFS Traversal:

WEEK 9:

#include<stdio.h>

#include<stdlib.h>

int arr[10][10],vis[10];

int n;

void dfs(int a);

void main()

{

    printf("enter the number of vertices");

    scanf("%d",&n);

    printf("n");

    printf("enter adjacency matrix");

    for(int i=0;i<n;i++)

    {

        for(int j=0;j<n;j++)

        {

            scanf("%d",&arr[i][j]);

        }

    }

    for(int i=0;i<n;i++)

    {

        vis[i]=0;

    }

    printf("dfs traversal");

    for(int i=0;i<n;i++)

    {

        if(vis[i]==0)

        {

            dfs(i);

        }

    }

}

void dfs(int a)

{

    printf("%d",a);

    vis[a]=1;

    for(int i=0;i<n;i++)

    {

        if(arr[a][a]==1&&arr[i]==0)

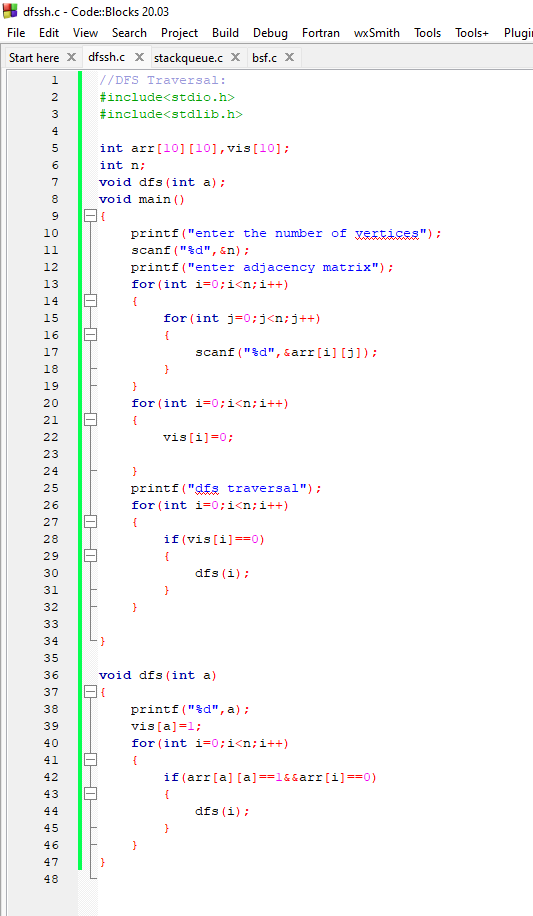
        {

            dfs(i);

        }

    }

}



//BFS Traversal:

#include <stdio.h>

#include <stdlib.h>

#define MAX 100 // Maximum number of vertices

int queue[MAX], front = -1, rear = -1; // Queue for BFS

int visited[MAX]; // To track visited nodes

// Function to add an element to the queue

void enqueue(int vertex) {

    if (rear == MAX - 1) return; // Queue overflow

    if (front == -1) front = 0;

    queue[++rear] = vertex;

}

// Function to remove an element from the queue

int dequeue() {

    if (front == -1 || front > rear) return -1; // Queue underflow

    return queue[front++];

}

// Function to perform BFS

void bfs(int graph[MAX][MAX], int n, int start) {

    for (int i = 0; i < n; i++) visited[i] = 0; // Reset visited array

    enqueue(start);

    visited[start] = 1;

    printf("BFS Traversal: ");

    while (front <= rear) {

        int current = dequeue();

        printf("%d ", current);

        for (int i = 0; i < n; i++) {

            if (graph[current][i] == 1 && !visited[i]) {

                enqueue(i);

                visited[i] = 1;

            }

        }

    }

    printf("\n");

}

int main() {

    int graph[MAX][MAX], n, start;

    printf("Enter the number of vertices: ");

    scanf("%d", &n);

    printf("Enter the adjacency matrix:\n");

    for (int i = 0; i < n; i++) {

        for (int j = 0; j < n; j++) {

            scanf("%d", &graph[i][j]);

        }

    }

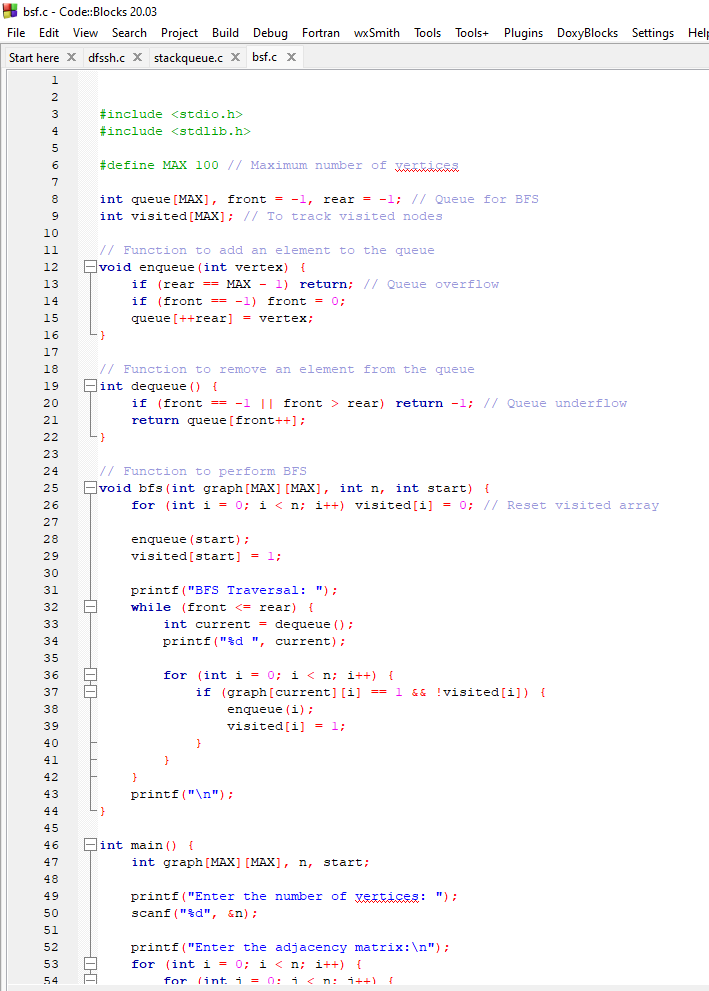
    printf("Enter the starting vertex for BFS: ");

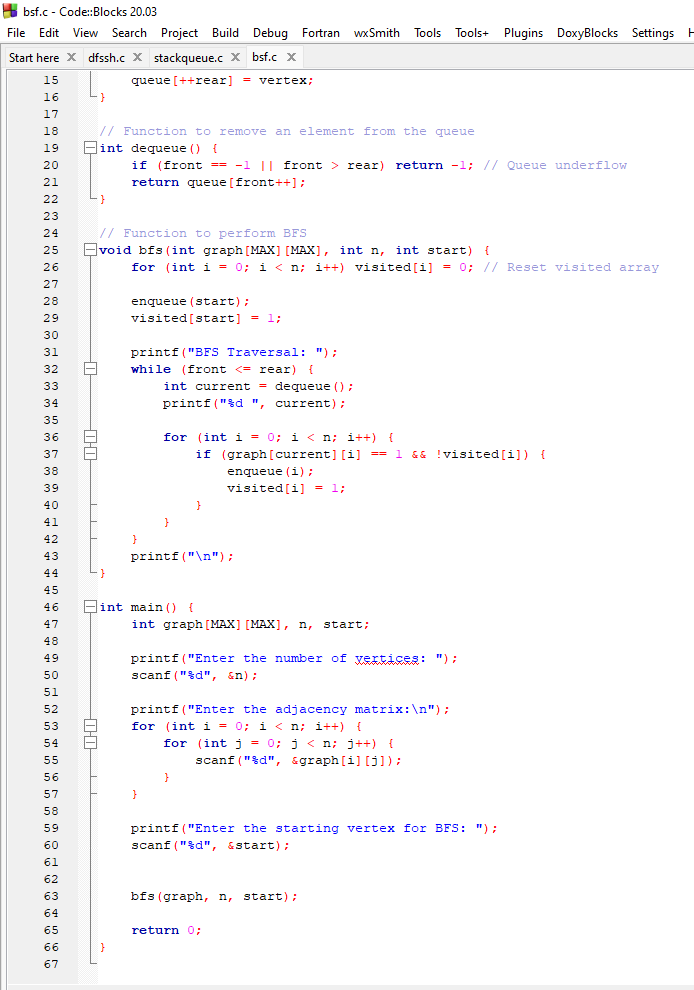
    scanf("%d", &start);

    bfs(graph, n, start);

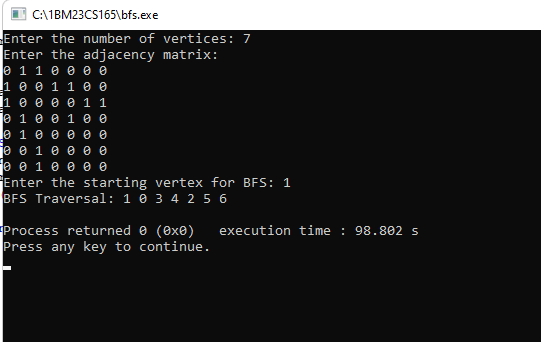
    return 0;

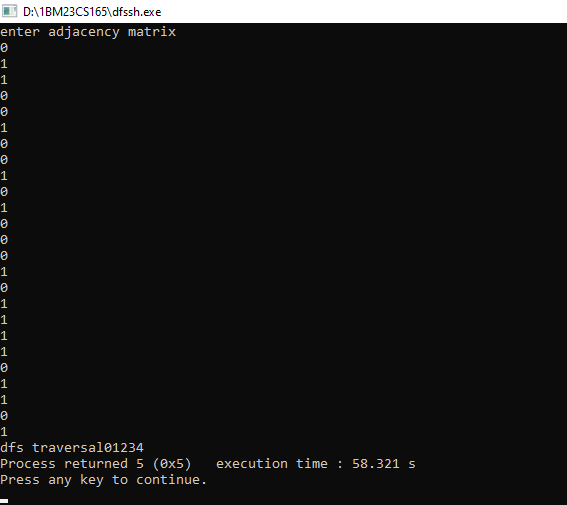
}





1.Output for BFS:



2.Output for DFS: