

Name : Hrithik Paul

Roll no : 20CS011017

Jisu Id: JISU/2020/0513

## Assignment 20:

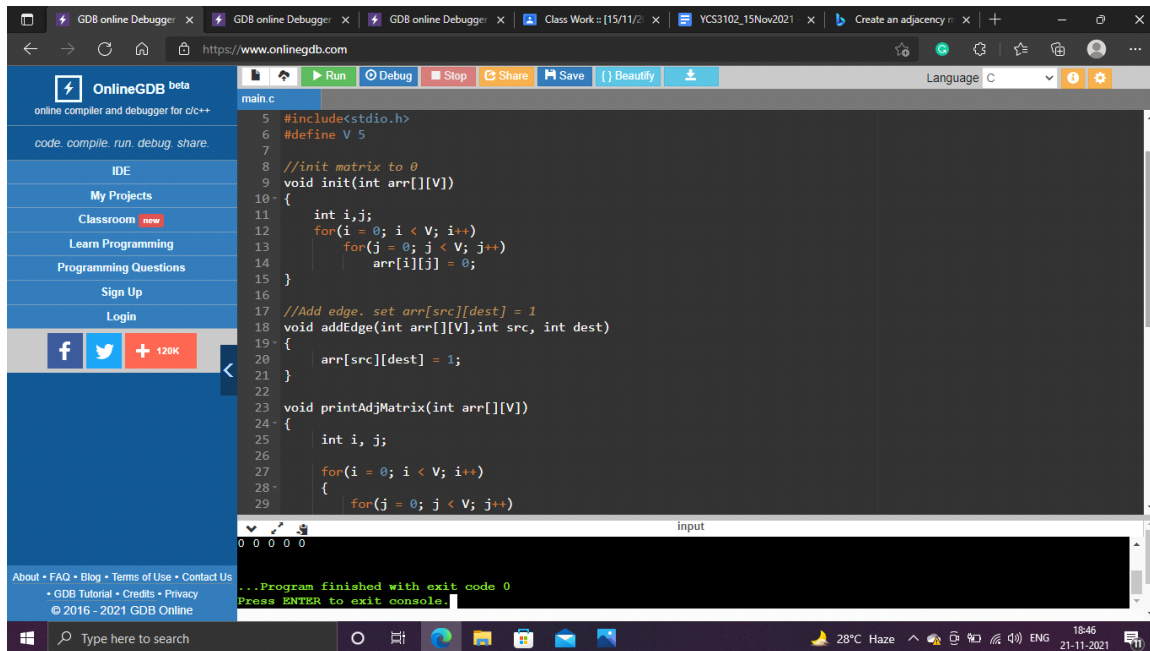
Create an adjacency matrix of a graph from the user given edge set

### Algo:

Step 1: Create a matrix A of size  $N \times N$  and initialise it with zero.

Step 2: Iterate over each given edge of the form (u,v) and assign 1 to  $A[u][v]$ . Also, If graph is undirected then assign 1 to  $A[v][u]$ .

### Output ss:



```
main.c
#include<stdio.h>
#define V 5
//init matrix to 0
void init(int arr[][V])
{
    int i,j;
    for(i = 0; i < V; i++)
        for(j = 0; j < V; j++)
            arr[i][j] = 0;
}
//Add edge. set arr[src][dest] = 1
void addEdge(int arr[][V],int src, int dest)
{
    arr[src][dest] = 1;
}
void printAdjMatrix(int arr[][V])
{
    int i, j;
    for(i = 0; i < V; i++)
    {
        for(j = 0; j < V; j++)
        {
            printf("%d ", arr[i][j]);
        }
        printf("\n");
    }
}

0 0 0 0 0

...Program finished with exit code 0
Press ENTER to exit console.
```

```
22
23 void printAdjMatrix(int arr[][V])
24 {
25     int i, j;
26
27     for(i = 0; i < V; i++)
28     {
29         for(j = 0; j < V; j++)
30         {
31             printf("%d ", arr[i][j]);
32         }
33         printf("\n");
34     }
35 }
36
37 //print the adjMatrix
38 int main()
39 {
40     int adjMatrix[V][V];
41
42     init(adjMatrix);
43     addEdge(adjMatrix,0,1);
44     addEdge(adjMatrix,0,2);
45     addEdge(adjMatrix,0,3);
46     addEdge(adjMatrix,1,3);
47 }
48
49 input
50 0 0 0 0 0
51
52 ...Program finished with exit code 0
53 Press ENTER to exit console.
```

```
30
31     printf("%d ", arr[i][j]);
32 }
33     printf("\n");
34 }
35 }
36
37 //print the adjMatrix
38 int main()
39 {
40     int adjMatrix[V][V];
41
42     init(adjMatrix);
43     addEdge(adjMatrix,0,1);
44     addEdge(adjMatrix,0,2);
45     addEdge(adjMatrix,0,3);
46     addEdge(adjMatrix,1,3);
47 }
48
49 input
50 0 1 1 0
51 0 0 1 1
52 0 0 0 1
53 0 0 0 0
54 0 0 0 0
55
56 ...Program finished with exit code 0
57 Press ENTER to exit console.
```

## Assignment 21:

**Algo:**Step 1: SET STATUS = 1 (ready state)

for each node in G

Step 2: Enqueue the starting node A

and set its STATUS = 2

(waiting state)

Step 3: Repeat Steps 4 and 5 until

QUEUE is empty

Step 4: Dequeue a node N. Process it

and set its STATUS = 3

(processed state).

Step 5: Enqueue all the neighbours of

N that are in the ready state

(whose STATUS = 1) and set

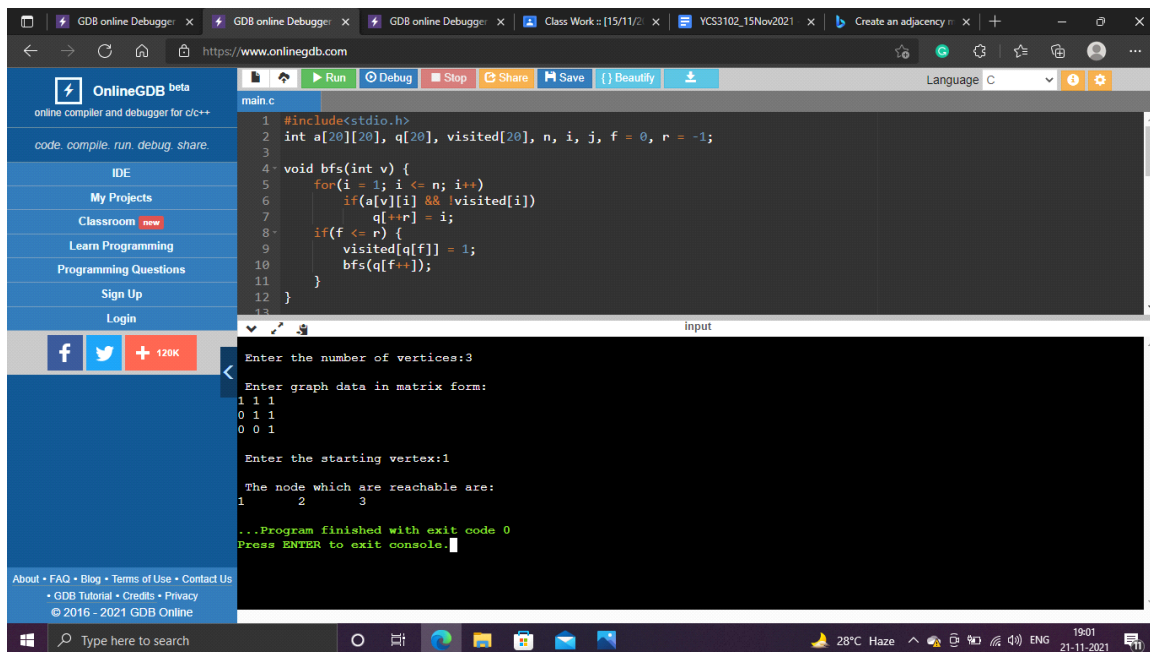
their STATUS = 2

(waiting state)

[END OF LOOP]

Step 6: EXIT

**Output :**



The screenshot shows a web browser window with the URL <https://www.onlinegdb.com>. The page features a sidebar on the left with navigation links: IDE, My Projects, Classroom (marked 'new'), Learn Programming, Programming Questions, Sign Up, and Login. Below these are social media icons for Facebook, Twitter, and a '+120K' badge. The main area displays a C++ code editor with the following code:

```
1 #include<stdio.h>
2 int a[20][20], q[20], visited[20], n, i, j, f = 0, r = -1;
3
4 void bfs(int v) {
5     for(i = 1; i <= n; i++)
6         if(a[v][i] && !visited[i])
7             q[f++] = i;
8     if(f <= r) {
9         visited[q[f]] = 1;
10        bfs(q[f]);
11    }
12 }
13
```

Below the code editor is an input/output console. The input section shows the user entering '3' for the number of vertices and a 3x3 adjacency matrix:

```
Enter the number of vertices:3
Enter graph data in matrix form:
1 1 1
0 1 1
0 0 1
```

The output section shows the starting vertex as '1' and the reachable nodes as '1 2 3':

```
Enter the starting vertex:1
The node which are reachable are:
1 2 3
...Program finished with exit code 0
Press ENTER to exit console.
```

**Assignment 22:**

**Algo:**

Step 1: SET STATUS = 1 (ready state) for each node in G

Step 2: Push the starting node A on the stack and set its STATUS = 2 (waiting state)

Step 3: Repeat Steps 4 and 5 until STACK is empty

Step 4: Pop the top node N. Process it and set its STATUS = 3 (processed state)

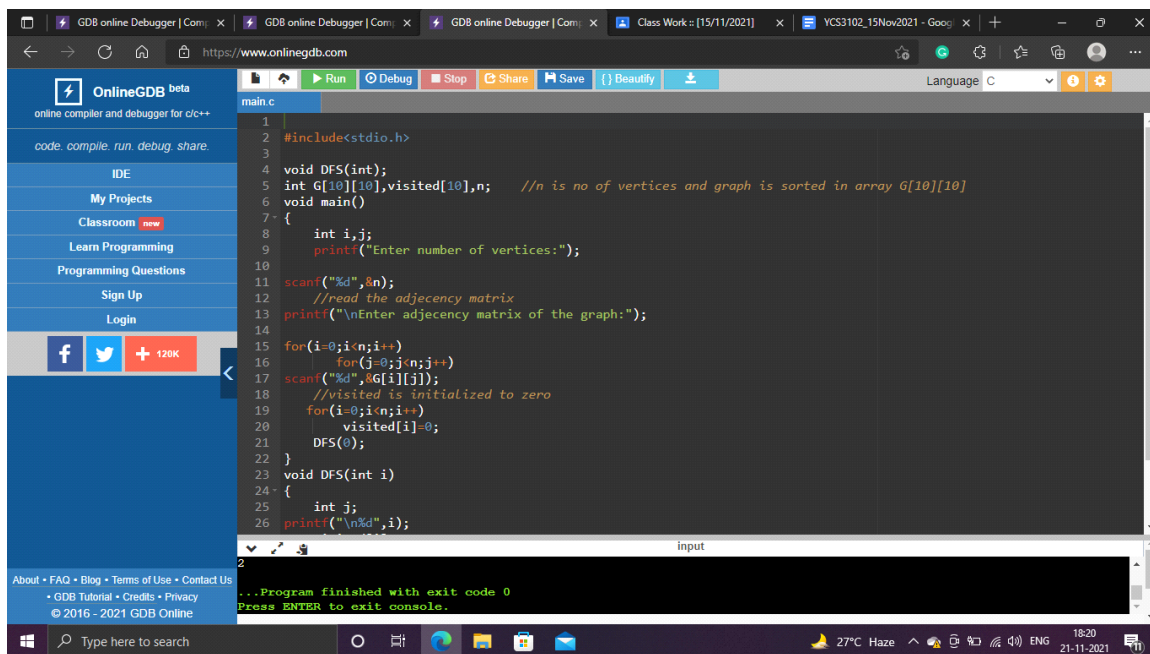
Step 5: Push on the stack all the neighbours of N that are in the ready state (whose STATUS = 1) and set their

STATUS = 2 (waiting state)

[END OF LOOP]

Step 6: EXIT

**Output:**



The screenshot displays the OnlineGDB web interface. The left sidebar contains navigation links: IDE, My Projects, Classroom, Learn Programming, Programming Questions, Sign Up, and Login. The main editor area shows a C program named 'main.c' implementing a Depth-First Search (DFS) algorithm. The code includes headers, defines a graph G, and uses a visited array to track nodes. The main function prompts the user for the number of vertices and the adjacency matrix, then calls the DFS function starting from node 0. The DFS function recursively visits nodes and prints their values. The bottom console shows the program's output, indicating it finished successfully with exit code 0.

```
1 #include<stdio.h>
2
3 void DFS(int);
4 int G[10][10],visited[10],n; //n is no of vertices and graph is sorted in array G[10][10]
5 void main()
6 {
7     int i,j;
8     printf("Enter number of vertices:");
9
10
11     scanf("%d",&n);
12     //read the adjacency matrix
13     printf("\nEnter adjacency matrix of the graph:");
14
15     for(i=0;i<n;i++)
16         for(j=0;j<n;j++)
17             scanf("%d",&G[i][j]);
18     //visited is initialized to zero
19     for(i=0;i<n;i++)
20         visited[i]=0;
21     DFS(0);
22 }
23 void DFS(int i)
24 {
25     int j;
26     printf("\n%d",i);
27 }
```

Input

2

...Program finished with exit code 0  
Press ENTER to exit console.

OnlineGDB beta  
online compiler and debugger for c/c++

code. compile. run. debug. share.

IDE

My Projects

Classroom **new**

Learn Programming

Programming Questions

Sign Up

Login

f t + 120K

About • FAQ • Blog • Terms of Use • Contact Us  
• GDB Tutorial • Credits • Privacy  
© 2016 - 2021 GDB Online

main.c

```
6 void main()
7 {
8     int i,j;
9     printf("Enter number of vertices:");
10
11     scanf("%d",&n);
12     //read the adjacency matrix
13     printf("\nEnter adjacency matrix of the graph:");
14
15     for(i=0;i<n;i++)
16         for(j=0;j<n;j++)
17             scanf("%d",&G[i][j]);
18     //visited is initialized to zero
19     for(i=0;i<n;i++)
20         visited[i]=0;
21     DFS(0);
22 }
23 void DFS(int i)
24 {
25     int j;
26     printf("\n%d",i);
27     visited[i]=1;
28     for(j=0;j<n;j++)
29         if(!visited[j]&&G[i][j]==1)
30             DFS(j);
31 }
```

input

2

...Program finished with exit code 0  
Press ENTER to exit console.

27°C Haze 18:20 21-11-2021

OnlineGDB beta  
online compiler and debugger for c/c++

code. compile. run. debug. share.

IDE

My Projects

Classroom **new**

Learn Programming

Programming Questions

Sign Up

Login

f t + 120K

About • FAQ • Blog • Terms of Use • Contact Us  
• GDB Tutorial • Credits • Privacy  
© 2016 - 2021 GDB Online

main.c

```
6 void main()
7 {
8     int i,j;
9     printf("Enter number of vertices:");
10
11     scanf("%d",&n);
12     //read the adjacency matrix
13     printf("\nEnter adjacency matrix of the graph:");
14
15     for(i=0;i<n;i++)
16         for(j=0;j<n;j++)
17             scanf("%d",&G[i][j]);
18     //visited is initialized to zero
19     for(i=0;i<n;i++)
20         visited[i]=0;
21     DFS(0);
22 }
23 void DFS(int i)
24 {
25     int j;
26     printf("\n%d",i);
27     visited[i]=1;
28     for(j=0;j<n;j++)
29         if(!visited[j]&&G[i][j]==1)
30             DFS(j);
31 }
```

input

Enter number of vertices:4

Enter adjacency matrix of the graph:0 1 1 1  
0 0 0 1  
1 0 0 0  
0 1 0 1

0  
1  
3  
2

...Program finished with exit code 0  
Press ENTER to exit console.

27°C Haze 18:20 21-11-2021

