Name - Shantanu Deshpande

class - SYCSE

Roll no. - 7

Batch - S1

Sub - D.S.

Practical No. 7

Aim: Implementation of DFS and BFS

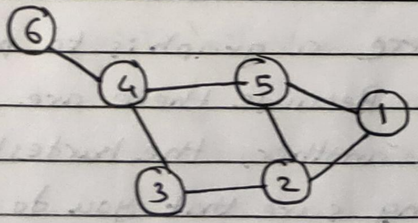
Theory:

Graph:

A graph G can be defined as a pair (V,E), where V is a set of vertices, and E is a set of edges between the vertices E.

Directed graph: A graph whose edges are ordered pairs of vertices.

Undirected graph: A graph whose edges are unordered pairs of vertices.



There are different ways to store graphs in a computer system

1. Adjacency list :-

Much like the incidence list, each vertex has a list of which vertices it is adjacent to. This causes redundancy in an undirected graph: for example, if vertices A and B are adjacent, A's adjacency list contain B, while B's list contains A. Adjacency queries are faster, at the cost of extra storage space.

2. Adjacency matrix :-

This is the n by n matrix A, where n is the number of vertices in the graph. If there is an edge from some vertex x to some vertex y, then the element a_{xy} is 1 (or in general the number of xy edges), otherwise it is 0. In computing, this matrix makes it easy to find subgraphs, and to reverse a directed graph.

- Graph Traversal Techniques :-

To traverse a graph is to process every node in the graph exactly once. Because there are many paths leading from one mode to another, the hardest part about traversing a graph is making sure that you do not process some node twice.

1. Depth - first search:-

DFS follows the following rules:

i) Select an unvisited node s, visit it and treat as the current node.

ii) Find an unvisited neighbor of the current node, visit it, and make it the new current node;

iii) If the current node has no unvisited neighbors, backtrack to the its parent and make that the new current node;

iv) Repeat the above two steps until no more nodes can be visited.

v) If there are still unvisited nodes, repeat from step 1.

2. Breadth - first Search:-

BFS follows the following rules:

i) Select an unvisited node s, visit it, have it be the root in a BFS tree being formed. Its level is called the current level.

ii) From each node x in the current level, in the order in which the level nodes were visited. visit all the unvisited neighbors of x. The newly visited nodes from this level form a new level that becomes the next current level.

iii) Repeat the previous step until no more nodes canbe visited.

iv) If there are still unvisited nodes, repeat from Step 1.

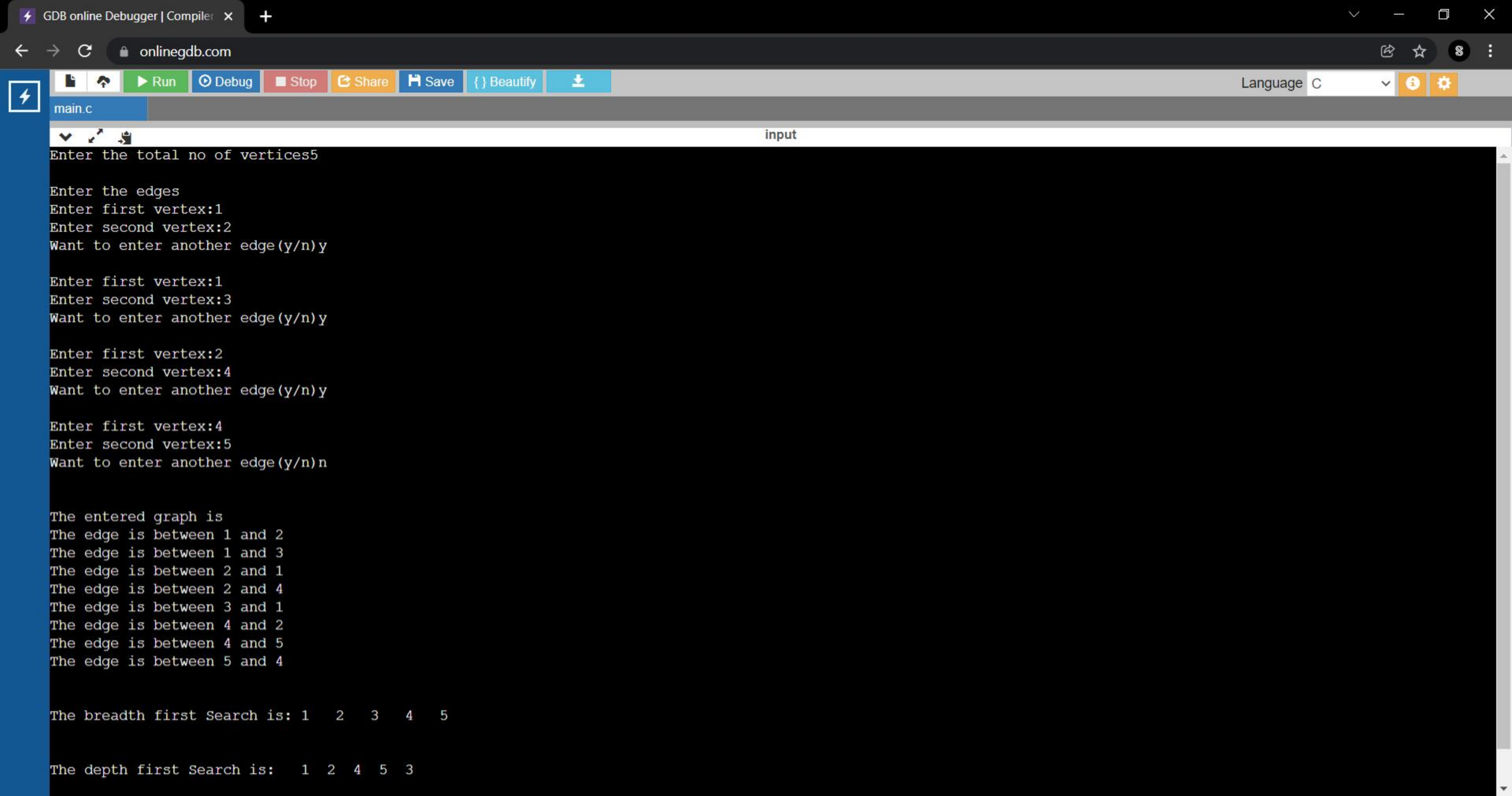
Conclusion:- Thus we have implemented DFS and BFS.

Name- Shantanu Deshpande
Class-SYCSE
Roll no.-7
Batch-S1

Input-

```
#include<stdio.h>

int g[10][10],visited[10];
void bfs(int,int);
void bft(int);
void dfs(int,int);
void dft(int);

void main()
{
int i,j,n;
char ch='y';

printf("Enter the total no of vertices");
scanf("%d",&n);
printf("\nEnter the edges");
do
{
do
{
printf("\nEnter first vertex:");
scanf("%d",&i);
}while(i>n);
do
{
printf("Enter second vertex:");
scanf("%d",&j);
}while(j>n);
g[i][j]=1;
g[j][i]=1;
printf("Want to enter another edge(y/n)");

scanf("%c",&ch);scanf("%c",&ch);
}while(ch=='y'||ch=='Y');
printf("\n\nThe entered graph is");
for(i=1;i<=n;i++)
for (j=1;j<=n;j++)
if(g[i][j]==1)
printf("\nThe edge is between %d and %d ",i,j);

printf("\n\n\nThe breadth first Search is: 1");
bft(n);
for(i=1;i<=n;i++)
visited[i]=0;
printf("\n\n\nThe depth first Search is: ");
dft(n);

}
```

```
void bft(int n)
{
int i;
for(i=1;i<=n;i++)
if(visited[i]==0)
bfs(i,n);
}
void bfs(int v,int n)
{
int w,front=0,rear=-1,Q[10];
visited[v]=1;
while(1)
{
 for(w=1;w<=n;w++)
 {
 if(g[v][w])
 {
 if(visited[w]==0)
 {
 Q[++rear]=w;
 visited[w]=1;
 printf("   %d",w);
 }

 }
 }
if(rear<front)
return;
v=Q[front++];
}
}
void dft(int n)
{
int i;
for(i=1;i<=n;i++)
if(visited[i]==0)
dfs(i,n);
}

 void dfs(int v,int n)
 {
 int w;
 visited[v]=1;
 printf("  %d",v);
for(w=1;w<=n;w++)
 {
 if(g[v][w])
 {
 if(visited[w]==0)
 dfs(w,n);
}
 }
 }
```



```
Enter the total no of vertices5

Enter the edges
Enter first vertex:1
Enter second vertex:2
Want to enter another edge(y/n)y

Enter first vertex:1
Enter second vertex:3
Want to enter another edge(y/n)y

Enter first vertex:2
Enter second vertex:4
Want to enter another edge(y/n)y

Enter first vertex:4
Enter second vertex:5
Want to enter another edge(y/n)n


The entered graph is
The edge is between 1 and 2
The edge is between 1 and 3
The edge is between 2 and 1
The edge is between 2 and 4
The edge is between 3 and 1
The edge is between 4 and 2
The edge is between 4 and 5
The edge is between 5 and 4


The breadth first Search is: 1   2   3   4   5


The depth first Search is:   1  2  4  5  3
```