|  |  |
| --- | --- |
|  | **DEPARTMENT OF COMPUTER ENGINEERING** |

**Experiment No.**

|  |  |
| --- | --- |
| Semester | S.E. Semester III – Computer Engineering |
| Subject | Data Structures Lab (CSL301) |
| Subject Professor In-charge | Prof. Swapnil S. Sonawane |
| Assisting Teachers | Prof. Swapnil S. Sonawane |

|  |  |
| --- | --- |
| Student Name | Trisha Shah |
| Roll Number | 20102A0004 |

**Title:**

Implement Graph Traversal techniques:) Depth First Search b) Breadth First Search

**Objective:**

Students will be able to analyze and Implement appropriate searching techniques for a given problem.

**Explanation:**

Depth-first search is an algorithm for traversing or searching tree or graph data structures. The algorithm starts at the root node (selecting some arbitrary node as the root node in the case of a graph) and explores as far as possible along each branch before backtracking. So the basic idea is to start from the root or any arbitrary node and mark the node and move to the adjacent unmarked node and continue this loop until there is no unmarked adjacent node. Then backtrack and check for other unmarked nodes and traverse them. Finally, print the nodes in the path.

Algorithm:

1. Create a recursive function that takes the index of the node and a visited array.
2. Mark the current node as visited and print the node.
3. Traverse all the adjacent and unmarked nodes and call the recursive function with the index of the adjacent node.

There are many ways to traverse graphs. BFS is the most commonly used approach.

BFS is a traversing algorithm where you should start traversing from a selected node (source or starting node) and traverse the graph layerwise thus exploring the neighbour nodes (nodes which are directly connected to source node). You must then move towards the next-level neighbour nodes.

As the name BFS suggests, you are required to traverse the graph breadthwise as follows:

1. First move horizontally and visit all the nodes of the current layer
2. Move to the next layer

**Program Code:**

A] Depth First Search

#include<stdio.h>

int adj[10][10]={0},visited[10]={0},n;

void dfs(int node)

{

int i;

visited[node]=1;

printf("%d ",node);

for(i=0;i<n;i++)

{

if(adj[node][i]==1 && visited[i]==0)

{

dfs(i);

}

}

}

int main()

{

int e,i,v1,v2,node;

printf("\nEnter number of vertices=");

scanf("%d",&n);

printf("\nEnter number of edges=");

scanf("%d",&e);

for(i=0;i<e;i++)

{

printf("\nEnter edge=");

scanf("%d%d",&v1,&v2);

adj[v1][v2]=adj[v2][v1]=1;

}

printf("\nEnter starting node=");

scanf("%d",&node);

dfs(node);

return 0;

}

B] Breath First Search

#include<stdio.h>

int adj[10][10]={0},visited[10]={0},n;

void bfs(int node)

{

int q[20],f=-1,r=-1,i,nd;

visited[node]=1;

q[++r]=node;

while(f!=r)

{

nd=q[++f];

printf("\t%d",nd);

for(i=0;i<n;i++)

{

if(adj[nd][i]==1 && visited[i]==0)

{

visited[i]=1;

q[++r]=i;

}

}

}

}

int main()

{

int e,i,v1,v2,node;

printf("\nEnter nos of nodes=");

scanf("%d",&n);

printf("\nEnter nos of edges=");

scanf("%d",&e);

printf("\nEnter edge details\n");

for(i=0;i<e;i++)

{

printf("\nEnter edge=");

scanf("%d%d",&v1,&v2);

adj[v1][v2]=adj[v2][v1]=1;

}

printf("\nEnter starting vertex=");

scanf("%d",&node);

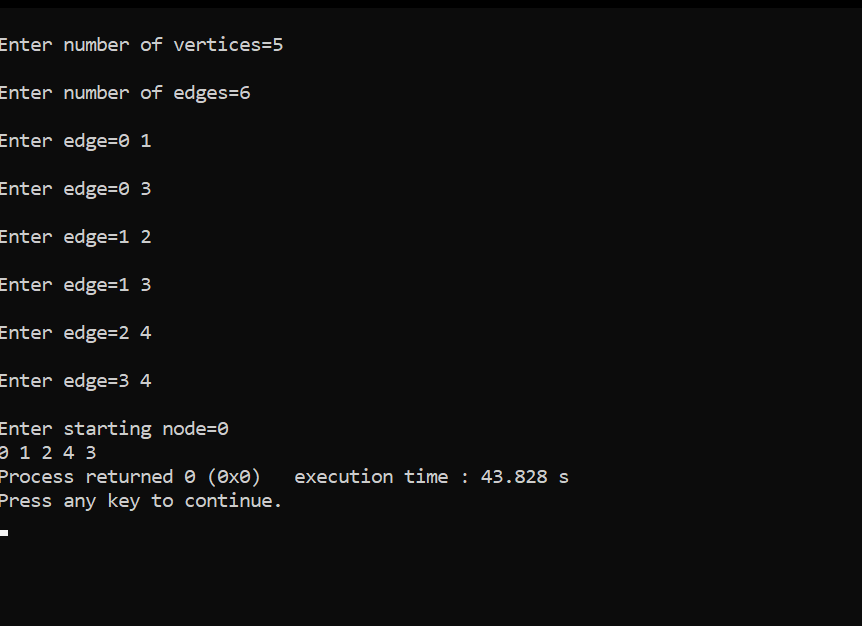
bfs(node);

return 0;

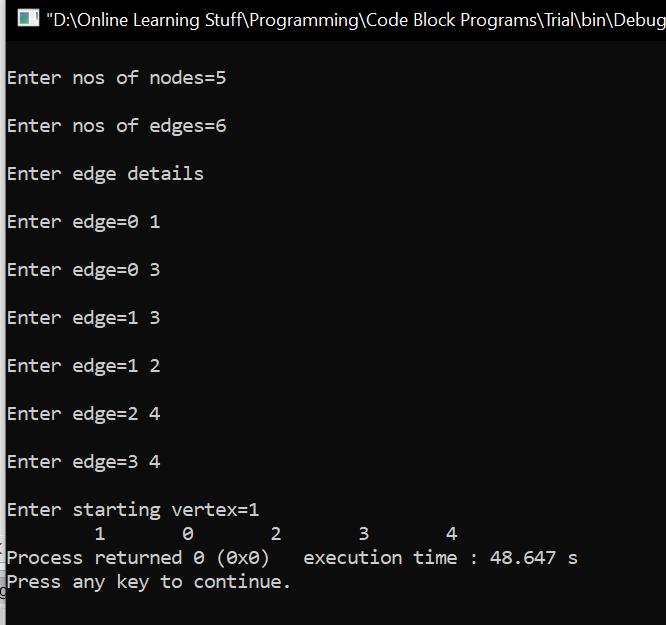
}

**Output:**

**DFS**

****

**BFS**

****

**Conclusion:**

Through this experiment, students were able to successfully implement DFS and BFS