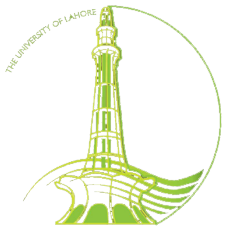

THE UNIVERSITY OF LAHORE
ISLAMABAD CAMPUS

Data Structuers and algorithms (CS09203)

Lab Report

Name:	Irtaza Kashir Raja
Registration #:	SEU-F16-127
Lab Report #:	10
Dated:	16-04-2018
Submitted To:	Mr. Usman Ahmed

The University of Lahore, Islamabad Campus
Department of Computer Science & Information Technology

Experiment # 10
BFS Graph and its representationsl

Objective
The objective of this session is to show the representation of graphs using C++.

Software Tool
1. Code Blocks with GCC compiler.

1 Theory

Breadth First Traversal (or Search) for a graph is similar to Breadth First Traversal of a tree. The only catch here is, unlike trees, graphs may contain cycles, so we may come to the same node again. To avoid processing a node more than once, we use a boolean visited array. For simplicity, it is assumed that all vertices are reachable from the starting vertex.

2 Task

2.1 Task 1

Impement Breadth First Traversal (or Search) for a graph

2.2 Procedure: Task 1

```
#include<iostream>
#include<queue>
using namespace std;

struct Node {
        char data;
```

```
        Node *left;
        Node *right;
};

void LevelOrder(Node *root) {
        if(root == NULL) return;
        queue<Node*> Q;
        Q.push(root);

        while(!Q.empty()) {
                Node* current = Q.front();
                Q.pop();
                cout<<current->data<<" ";
                if(current->left != NULL) Q.push(current->left);
                if(current->right != NULL) Q.push(current->right);
        }
}
Node* Insert(Node *root,char data) {
        if(root == NULL) {
                root = new Node();
                root->data = data;
                root->left = root->right = NULL;
        }
        else if(data <= root->data) root->left = Insert(root->left,data);
        else root->right = Insert(root->right,data);
        return root;
}

int main() {

        Node* root = NULL;
        root = Insert(root,'M'); root = Insert(root,'B');
        root = Insert(root,'Q'); root = Insert(root,'Z');
        root = Insert(root,'A'); root = Insert(root,'C');

        LevelOrder(root);
}
```

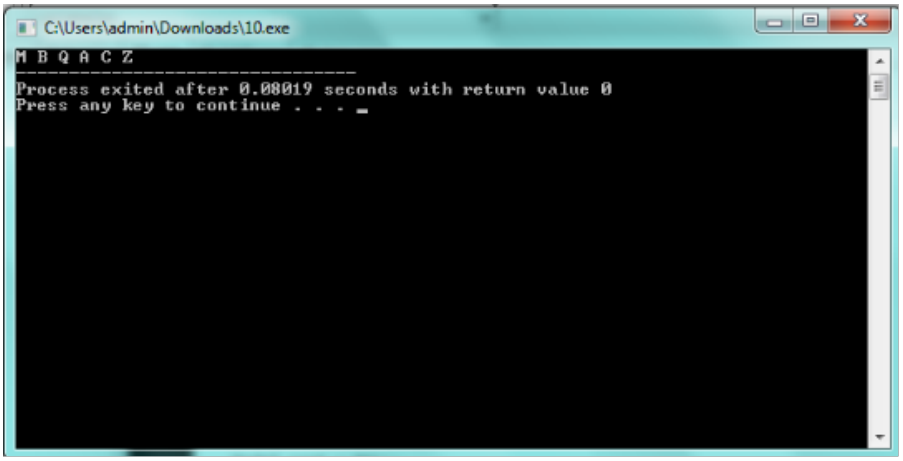

Figure 1: output