BLG 233E DATA STRUCTURES AND LABORATORY

EXPERIMENT 10 – TREE TRAVERSING

IMPORTANT REMINDERS

1. It is not allowed to use USB sticks during the lab sessions.
2. You should unplug your ethernet cables during the lab sessions.
3. Any reference book or help material (C++) is allowed.

In this experiment, you are required to implement three search strategies on a **binary tree**. First, you should create your binary tree with N nodes. Each node in the tree will store an integer value (You may randomly generate numbers while creating nodes). You should not add an element to the depth (d) if depth (d-1) has an empty spot.
User will enter an integer value to be searched and this value will be searched with three approaches which are explained below.
You will apply the following search strategies to your tree.

a. Inorder Traversal

Write a recursive inorder traversal function that searches the given value on a tree. Think of what may go wrong with this type of search when the tree size is large. Comment on this issue.

b. Breadth First Search

BFS strategy uses queue data structure to traverse the tree. The pseudo code of BFS algorithm for a tree is given below.

```
Algorithm BFS()
        initialize an empty queue Q;
        insert root in Q;
        while Q is not empty
                u = front of Q;
                remove u from Q;
                for each children v of u
                        insert v in Q;
                print u;
```

c. Depth First Search

DFS strategy uses stack data structure to traverse the tree. The pseudo code of DFS algorithm for a tree is given below.

```
Algorithm DFS()
        initialize an empty stack S;
        insert root in S;
        while S is not empty
                u = top of S;
                remove u from S;
                for each children v of u
                        insert v in S;
                print u;
```

Your search should also stop when the searched value is found. At the end of your program you should give neccessary messages to the user and print out the number of search steps that your algorithms has taken. You should also comment on the results of the algorithms in terms of number of search steps that each algorithm has taken.