Laboratory 6

Binary Trees

I. OBJECTIVES:

In this laboratory, you will

- understand and apply the concepts of binary trees.
- implement binary tree operations and functions using linked structure.
- create a program that will simulate applications of binary trees such as finding duplicate numbers and traversal methods; preorder, inorder and postorder.

II. DISCUSSION:

A **binary tree** is a finite set of elements that is either empty or is partitioned into three disjoint subsets. The first subset contains a single element called the **root** of the tree. The other two subsets are themselves binary tree, called the **left** and **right subtrees** of the original tree. A left or right subtree can be empty. Each element of a binary tree is called **node** of the tree. A conventional method of picturing a binary tree is shown in figure 6-1.

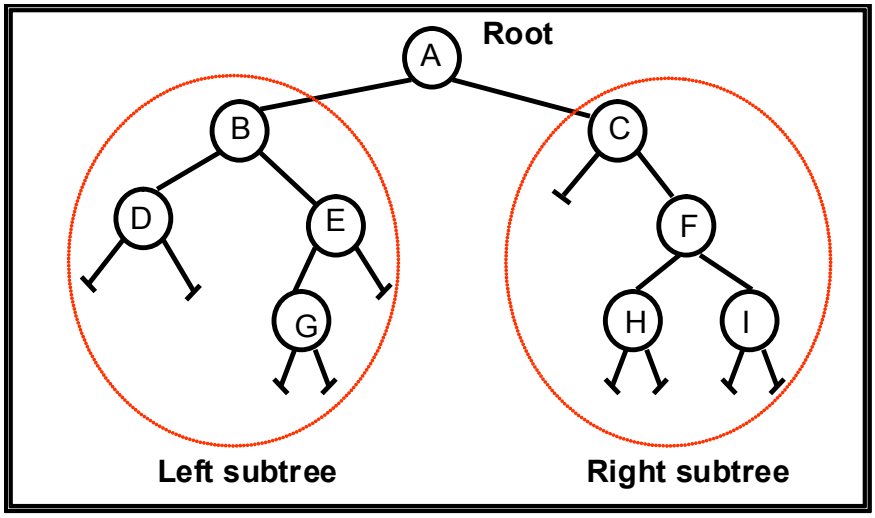

Figure 6-1: Binary Tree

Binary Tree Operations

If **p** is a pointer to a node **nd** of a binary tree, the following functions can be described as:

info(p)	returns the contents of *nd*
left(p)	returns a pointer to the left son of *nd*
right(p)	returns a pointer to the right son of *nd*
father(p)	returns a pointer to the father of *nd*
brother(p)	returns a pointer to the brother of *nd*
isleft(p)	returns true if *nd* is a left son of other node
isright(p)	returns true if *nd* is a right son of other node
maketree(x)	creates a new binary tree consisting of a single node with information field *x* and returns a pointer to a node.
s*etleft(p,x)*	creates a new left son with information field *x*.
s*etright(p,x)*	creates a new right son with information field *x*.

Traversal Methods

Another common application of Binary Tree is to **traverse** a binary tree.

- Preorder
  The sequence of operations is the following:
  1. Visit the root.
  2. Traverse the left subtree in preorder.
  3. Traverse the right subtree in preorder.

- Inorder
  The sequence of operations is the following:
  1. Traverse the left subtree in inorder.
  2. Visit the root.
  3. Traverse the right subtree in inorder.

- Postorder
  The sequence of operations is the following:
  1. Traverse the left subtree in postorder.
  2. Traverse the right subtree in postorder
  3. Visit the root.

III. Test and Debug

1. Write the following functions that will implement the Binary Tree operations:

Binary Tree Operations
info(p), left(p), right(p), father(p), brother(p), isleft(p), isright(p), maketree(x), setleft(p,x), setright(p,x)

2. Compile and test each operation using the following commands:

Commands	Operation
> m x	maketree (x)
> s L x	setleft(p,x)
> s R x	setright(p,x)
> d p/q	info(p); display info pointed to by p or q
> l q	left(p); returns pointer to q
> r q	right(p); returns pointer to q
> f q	father(p); returns pointer to q
> b q	brother(p); returns pointer to q
>? L	isleft(q)
>? R	isright(q)

Do the following sample runs:

Commands	Results/Outputs
> m a	p -> a
> s L b	Left node b
> s R c	Right node c
> d p	A
> l q	q -> left node
> d q	b
> r q	q -> right node
> d q	c
> f q	q -> father node
> d q	a
> b q	q -> brother node
> d q	no brother
> l q	q -> left node
> ? R	FALSE
> ? L	TRUE

IV. Supplementary Exercises

Consider the following Tree operations,

Additional Binary Tree Operations	
iscomplete(p, d)	returns true if the binary tree is a complete binary tree given the depth(d) of the tree.
root()	returns the pointer of the root of the binary tree

Develop functions for these operations and add them in *trees.c* file. Using the following commands, prepare a test plan that will evaluate the operations.

Commands	Operation
>? c d	iscomplete(p,d)
>rt	root()

V. Machine Problem

1. Write a program that will create the given binary tree. Display the contents of the created binary tree using the given format.

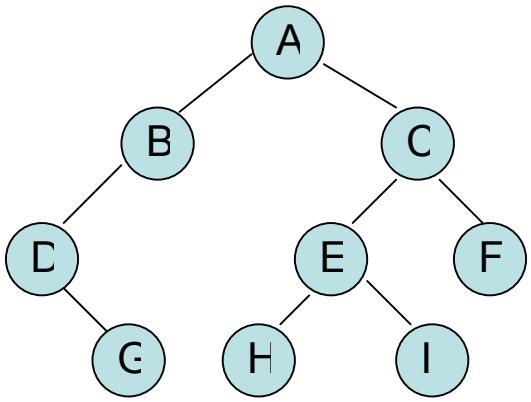

Figure 6-2: Binary Tree

DISPLAY FORMAT:

```
Level 0: A
Level 1: B C
Level 2: D- EF
Level 3: -G -- HI --
```

Note: *The instructor may change the machine problem.*

Binary Trees

DATA & RESULTS SHEET

(Tentative Laboratory Report)

Name: _____

Schedule:_____ Section:_____

Date: _____ Grade: _____

Test & Debug:

Step No.	Answers/Results
1	Attach source codes
2	

Commands	Results/Outputs
> m a	p -> a
> s L b	Left node b
> s R c	Right node c
> d p	A
> l q	q -> left node
> d q	b
> r q	q -> right node
> d q	c
> f q	q -> father node
> d q	a
> b q	q -> brother node
> d q	no brother
> rt	q -> root node
> l q	q -> left node
> ? R	FALSE
> ? L	TRUE
> ? c 1	TRUE

Supplementary Exercises:

Requirements	Completed?		Remarks
	YES	NO	
1. Programs			
2. Test Plan			

Machine Problem:

Topic	Completed?		Remarks
	YES	NO	

Note: Check the column YES if completed otherwise check the column NO.(for instructors use only)

INSTRUCTOR'S SIGNATURE: _____