08

Binary Search Tree and Operations

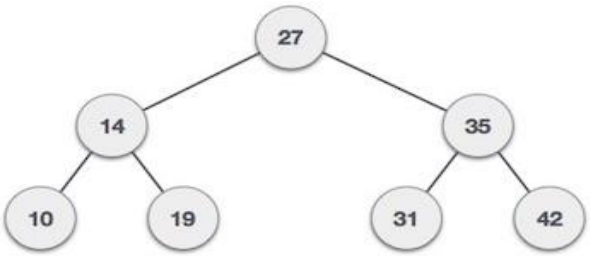
DATA STRUCTURE LAB

SESSION - 08

Binary Search Tree (BST):

→ Left child < Node

→ Right child > Node



Basic Operations:

Insert:

To insert a data element first search from the root node. If the data is less than the key value then search an empty location in the left subtree and insert the data. Otherwise, search empty location in the right subtree and insert the data.

Insert function Implementation:

```
void insert(tree **t, int val)
{
        tree *temp=NULL;
        if(!(*t))
        {
                        temp=(tree*)malloc(sizeof(tree));
                        temp->left=temp->right=NULL;
                        temp->data=val;
                        *t=temp;
                        return;
        }
        if(val<(*t)->data)
```

```
        {
                        insert(&(*t)->left,val);
        }
        else if(val>(*t)->data)
        {
                        insert(&(*t)->right,val);
        }
}
```

Search:

Whenever searching an element start from the root node. Then if the data is less than the key value, then search in the left subtree. Else if the data is greater than the key value search in the right subtree.

Search Implementation:

```
void search(tree *t,int data)
{
        if(t!=NULL)
        {
                        if(t->data==data)
                        {
                                printf("\nYes\n");
                        }
                        else if(t->data > data)
                        {
                                search(t->left,data);
                        }
                        else if(t->data < data)
                        {
                                search(t->right, data);
                        }
        }
        else
```

```
        {
                        printf("\nNo\n");
        }
}
```

#EXERCISE:

1. Using the following data make a binary search tree. Then convert it into max heap & min heap.

10,16,4,25,54,12,8,3,20,32