DS LAB TEST 2

Anitej prasad

1BM19CS194

3-D

4.1.21

BINARY SEARCH TREE QUESTION (3)

```c
#include <stdio.h>
#include <stdlib.h>
struct node
{
        int data;
        struct node* left;
        struct node *right;
};
struct node *create()
{
        struct node *temp;
        printf("\n Enter data:");
        temp=(struct node*)malloc(sizeof(struct node));
        scanf("%d",&temp->data);
        temp->left=temp->right=NULL;
        return temp;
}
 void insert(struct node *root,struct node *temp)
{
```

```c
        if(temp->data<root->data)
        {
                if(root->left!=NULL)
                        insert(root->left,temp);
                else
                        root->left=temp;
        }


        if(temp->data>root->data)
        {
                if(root->right!=NULL)
                        insert(root->right,temp);
                else
                        root->right=temp;
        }
}

void max(struct node *root)
{
        while(root->right!=NULL)
        {
                root= root->right;
        }
        printf("Max is %d\n",root->data);
}
void display(struct node *r,int level)
{
    int i;
    if(root == NULL )
         return;
    else
```

```c
    {
            display(root->r, level+1);

            printf("\n");

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

                printf("    ");

            printf("%d", root->info);

            display(root->l, level+1);

    }


int main()
{
        int ch,count=1;

        struct node *tree;

        struct node *rt;

        do

        {
                printf("1.Create and insert node in BST\n2.display\n3.Find maximum element\n4.Exit\n");

                printf("enter choice\n");

                scanf("%d",&ch);

                switch(ch)

                {
                        case 1:

                        if(count==1)

                        {
                                rt=create();

                                count++;

                        }

                        else

                        {
                                tree=create();
```

```c
                insert(rt,tree);
        }
        break;

        case 2:
        display(rt);
        break;

        case 3:
        max(rt);
        break;



        case 4:
        break;

        default:
        printf("wrong choice!\n");
        break;
    }
}while(ch!=4);
return 0;
}
```