NAME: Jagtap Rohit Badrinath

CLASS: SE COMPUTER

DIV: A

BATCH: B3

ASSIGNMENT NO:4

CODE:-

```
#include<iostream>
#include<stdlib.h>
using namespace std;
class bstnode
{
    public:
    int data;
    bstnode *left,*right;
    bstnode(int x)
    {
        data=x;
        left=right=NULL;
    }
};
class bst
{
    bstnode*root;
    public:
    bst()
    {
        root=NULL;
    }
        bstnode*create();
        void insert(int x);
        bstnode*find(int x);
        bstnode*find_min(bstnode*root);
        int longest_path(bstnode*T);
        void display(bstnode*t);
        bstnode*swapper(bstnode*t);
};
bstnode*bst::create()
{
```

```
int x,i,n;
root=NULL;
cout<<"enter total number of nodes :";
cin>>n;
cout<<"enter tree value :";
for(i=0;i<n;i++)
{
        cin>>x;
        insert(x);
    }
    return(root);
}
void bst::insert(int x)
{
    bstnode *p,*q,*r;
    r=new bstnode (x);
    if(root==NULL)
    {
        root=r;
    return;
}
p=root;
while(p!=NULL)
{
    q=p;
    if(x>p->data)
        p=p->right;
    else
        p=p->left;
}
if(x>q->data)
        q->right=r;
    else
        q->left=r;
}
bstnode*bst::find(int x)
{
   while(root!=NULL)
     {
```

```
        if(x==root->data)
        return (root);
        if(x>root->data)
        root=root->right;
        else
        root=root->left;
}
return NULL;
}
bstnode *bst::find_min(bstnode*root)
{
        while(root->left!=NULL)
        {
                return(root->left);
        }
        return(root);
}
int bst::longest_path(bstnode*T)
{
int hl,hr;
if(T==NULL)
return(0);
if(T->left==NULL && T->right==NULL)
return(0);
hl=longest_path(T->left);
hr=longest_path(T->right);
if(hl>hr)
{
        return(hl+1);
}
else
        {
                return(hr+1);
        }
}
void bst::display(bstnode *t)
{
        if(t!=NULL)
        {
```

```
            display(t->left);
            cout<<"\t"<<t->data;
            display(t->right);
        }
}
bstnode*swapper(bstnode*t)
{
        bstnode*c;
        if(t!=NULL)
        {
                c=t->left;
                t->left=swapper(t->right);
                t->right=swapper(c);
        }
        return(t);
}
int main()
{
        int ch,x,i;
        bst b;
        bstnode*p,*q,*root;
        do
        {
                cout<<"\n1.create \n2.find
\n3.find_min\n4.longest_path\n5.display\n6.swap";
                cout<<"\nenter u r choice : ";
                cin>>ch;
                switch(ch)
                {
                        case 1:
                        root=b.create();
                        break;
                        case 2:
                        cout<<"enter node to be searched ";
                        cin>>x;
                        p=b.find(x);
                        if(p==NULL)
                        cout<<"\nnode not found ";
                        else
```

```
cout<<"node found"<<p->data;
break;
case 3:
q=b.find_min(root);
cout<<"minimum value in tree "<<q->data;
break;
case 4:
i=b.longest_path(root);
        cout<<" longest path in tree  "<<i+1;
        break;
        case 5:
        b.display(root);
        break;
        case 6:
        swapper(root);
        break;
    }
}
while(ch!=7);
    return 0;
}
```

OUTPUT:-

```
1.create
2.find
3.find_min
4.longest_path
5.display
6.swap
enter u r choice : 1
enter total number of nodes :5
enter tree value :34
58
36
74
91

1.create
2.find
```

3.find_min
4.longest_path
5.display
6.swap
enter u r choice : 2
enter node to be searched 91
node found91

1.create
2.find
3.find_min
4.longest_path
5.display
6.swap
enter u r choice : 3
minimum value in tree 34

1.create
2.find
3.find_min
4.longest_path
5.display
6.swap
enter u r choice : 4
 longest path in tree 4

1.create
2.find
3.find_min
4.longest_path
5.display
6.swap
enter u r choice : 5
      34   36   58   74   91
1.create
2.find
3.find_min
4.longest_path
5.display

6.swap
enter u r choice : 6

1.create
2.find
3.find_min
4.longest_path
5.display
6.swap
enter u r choice : 5
91 74 58 36 34