



**#include**<stdio.h>

**#include**<stdlib.h>

**typedef** **struct** bstnode

{

**int** data;

**struct** bstnode \*left,\*right;

}bst;

bst \***find**(bst \*,**int**);

bst \***insert**(bst \*,**int**);

bst \***create**();

**void** **inorder**(bst \*t);

**void** **preorder**(bst \*t);

**void** **postorder**(bst \*t);

**int** **main**()

{

bst \*root=NULL,\*p;

**int** x,op;

**do**

{

**printf**("\n1)create\n2)search\n3)insert\n4)preorder\n5)inorder\n6)postorder\n7)quit");

**printf**("choose");

**\_flushall**();

**scanf**("%d",&op);

**switch**(op)

{

**case** 1:root=create();**break**;

**case** 2:**printf**("enter key to be searched");

**\_flushall**();

**scanf**("%d",&x);

p=find(root,x);

**if**(p==NULL)

**printf**("\n Not found");

**else**

**printf**("\n found");

**break**;

**case** 3:**printf**("\n enter data");

**scanf**("%d",&x);

root=insert(root,x);

**break**;

**case** 4:preorder(root);**break**;

**case** 5:inorder(root);**break**;

**case** 6:postorder(root);**break**;

}

}**while**(op!=7);

**return** 0;

}

bst \***create**()

{

**int** n,x,i;

bst \*root;

root=NULL;

**printf**("\n Enter no. of nodes");

**\_flushall**();

**scanf**("%d",&n);

**printf**("\n Enter tree values");

**\_flushall**();

**for**(i=0;i<n;i++)

{

**scanf**("%d",&x);

root=insert(root,x);

}**return**(root);

}

bst \***insert**(bst \*t,**int** x)

{

bst \*p,\*q,\*r;

r=(bst\*)**malloc**(**sizeof**(bst));

r->data=x;

r->left=NULL;

r->right=NULL;

**if**(t==NULL)

**return**(r);

p=t;

**while**(p!=NULL)

{

q=p;

**if**(x>p->data)

p=p->right;

**else**

**if**(x<p->data)

p=p->left;

**else**

{

**printf**("\n duplicate entry");

**return**(t);

}

}

**if**(x>q->data)

q->right=r;

**else**

q->left=r;

**return**(t);

}

**void** **postorder**(bst \*t)

{

**if**(t!=NULL)

{

postorder(t->left);

postorder(t->right);

**printf**("%d",t->data);

}

}

**void** **inorder**(bst \*t)

{

**if**(t!=NULL)

{

inorder(t->left);

**printf**("%d\t",t->data);

inorder(t->right);

}

}

**void** **preorder**(bst \*t)

{

**if**(t!=NULL)

{

**printf**("%d\t",t->data);

preorder(t->left);

preorder(t->right);

}

}

bst \***find**(bst \*root,**int** x)

{

**while**(root!=NULL)

{

**if**(x==root->data)

**return**(root);

**if**(x>root->data)

root=root->right;

**else**

root=root->left;

}

**return**(NULL);

}