LP #10 : WAP a) To construct a binary search tree b) To traverse the tree using all the methods (inorder, preorder, postorder) c) To display the element in the tree.

```
#include <stdio.h>
#include <stdlib.h>
struct node {
int info;
struct node *rlink;
struct node *llink;
};
typedef struct node *NODE;
NODE getnode() {
NODE x;
x = (NODE) malloc (sizeof(struct node));
if (x == NULL) {
printf ("mem full\n");
exit(0);
}
return x;
}
void freenode (NODE x) {
free(x);
}
```

```
NODE insert (NODE root, int item) {
NODE temp, cur, prev;
temp = getnode() ;
temp -> rlink = NULL;
temp -> llink = NULL ;
temp -> info = item ;
if (root == NULL)
 return temp ;
prev=NULL;
cur = root;
while (cur != NULL) {
 prev = cur;
 cur = (item < cur->info) ? cur -> llink : cur-> rlink ;
 }

if (item < prev->info)
 prev->llink = temp;
else prev -> rlink =temp;
 return root;
 }


void display (NODE root, int i) {
 int j;
 if (root != NULL) {
 display (root -> rlink , i+1);
```

```
for (j=0 ; j<i ; j++)
 printf(" ");
 printf("%d \n", root->info);
  display (root->llink , i+1); }
}


void preorder(NODE root) {
if (root!=NULL ) {
 printf("%d\n", root->info);
 preorder (root->llink);
preorder (root ->rlink); }
 }


void postorder (NODE root) {
if (root ! =NULL ) {
 postorder(root->llink);
 postorder (root ->rlink);
 printf("%d\n", root->info);
 }
}


void inorder (NODE root) {
if(root ! =NULL) {
 inorder (root->llink );
```

```
printf ("%d\n", root→info );
inorder (root→rlink); }
}

int main () {
int item, choice ;
NODE root = NULL;
do
{
printf ("\n1. insert \n2. display \n 3 . preorder \n 4. postorder \n 5. inorder
\n 6. exit \n");
printf ("enter the choice \n");
scanf ("%d", &choice);
switch (choice)
{
case 1 : printf ("enter the item\n");
         scanf ("%d", &item);
         root = insert (root, item);
         break;
case 2 : display (root, 0);
         break;
case 3 : preorder (root);
        break;
case 4 : inorder (root);
break;
case 6 : break;
defaut : exit(0); break;
}
} while(choice ! =6);
 }
```