1. Write a C program to print preorder , inorder and postorder traversal on binary tree?

Program:-

|  |
| --- |
| // C program for different tree traversals  #include <stdio.h>  #include <stdlib.h>    /\* A binary tree node has data, pointer to left child     and a pointer to right child \*/  struct node  {       int data;       struct node\* left;       struct node\* right;  };    /\* Helper function that allocates a new node with the     given data and NULL left and right pointers. \*/  struct node\* newNode(int data)  {       struct node\* node = (struct node\*)                                    malloc(sizeof(struct node));       node->data = data;       node->left = NULL;       node->right = NULL;         return(node);  }    /\* Given a binary tree, print its nodes according to the    "bottom-up" postorder traversal. \*/  void printPostorder(struct node\* node)  {       if (node == NULL)          return;         // first recur on left subtree       printPostorder(node->left);         // then recur on right subtree       printPostorder(node->right);         // now deal with the node       printf("%d ", node->data);  }    /\* Given a binary tree, print its nodes in inorder\*/  void printInorder(struct node\* node)  {       if (node == NULL)            return;         /\* first recur on left child \*/       printInorder(node->left);         /\* then print the data of node \*/       printf("%d ", node->data);         /\* now recur on right child \*/       printInorder(node->right);  }    /\* Given a binary tree, print its nodes in preorder\*/  void printPreorder(struct node\* node)  {       if (node == NULL)            return;         /\* first print data of node \*/       printf("%d ", node->data);         /\* then recur on left sutree \*/       printPreorder(node->left);         /\* now recur on right subtree \*/       printPreorder(node->right);  }    /\* Driver program to test above functions\*/  int main()  {       struct node \*root  = newNode(1);       root->left             = newNode(2);       root->right           = newNode(3);       root->left->left     = newNode(4);       root->left->right   = newNode(5);         printf("\nPreorder traversal of binary tree is \n");       printPreorder(root);         printf("\nInorder traversal of binary tree is \n");       printInorder(root);         printf("\nPostorder traversal of binary tree is \n");       printPostorder(root);         getchar();       return 0;  }  Output:-  Preorder traversal of binary tree is  1 2 3 4 5 3  Inorder traversal of binary tree is  4 2 5 1 3  Postorder traversal of binary tree is  4 5 2 3 1 |

2.Write a C program to create (or insert) or traversal on Binary Search Tree?

Program:-

//C program to demonstrate insert operation in binary search tree//

#include<stdio.h>

#include<stdlib.h>

struct node

{

int key;

struct node \*left, \*right;

};

struct node \*newNode(int item)

{

struct node \*temp = (struct node \*)malloc(sizeof(struct node));

temp->key = item;

temp->left = temp->right = NULL;

return temp;

}

void inorder(struct node \*root)

{

if (root != NULL)

{

inorder(root->left);

printf("%d \n", root->key);

inorder(root->right);

}

}

struct node\* insert(struct node\* node, int key)

{

/\* If the tree is empty, return a new node \*/

if (node == NULL) return newNode(key);

/\* Otherwise, recur down the tree \*/

if (key < node->key)

node->left = insert(node->left, key);

else if (key > node->key)

node->right = insert(node->right, key);

/\* return the (unchanged) node pointer \*/

return node;

}

int main()

{

struct node \*root = NULL;

root = insert(root, 5);

insert(root, 3);

insert(root, 2);

insert(root, 4);

insert(root, 7);

insert(root, 6);

insert(root, 8);

// print inoder traversal of the BST

inorder(root);

return 0;

}

Output:

2

3

4

5

6

7

8

3.Write a C program for linear search?

Program:-

#include<stdio.h>

int main ( )

{

int a[20],i,x,n;

printf(“How many elements ?”);

scanf(“%d”, &n);

printf(“Enter array elements : n”);

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

scanf(“%d” , &a[i]);

printf(“nEnter elements to search :”);

scanf(“%d” , &x) ;

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

if (a[i]==x)

break;

if (i<n)

printf(“Elements found at index %d” , i);

else

printf(“Element not found”);

return 0;

}

Output:-

Enter array elements:

21

22

34

15

61

87

19

20

52

48

Enter element to search:15

Element found at index 4

4.Write a C program for binary search?

Program:-

#include

Include main( )

{

int c, first, last, middle, n, search, array[100];

printf(“enter the number of elements:\n”);

scanf(“%d” ,&n);

printf(“Enter %d integers :\n”, n);

for (c=0; c<n; c++)

scanf(“%d”, &array[c]);

printf(“Enter the value to find: \n”);

scanf(“%d”, &search);

first=0;

last=n-1

middle=(first+last)/2;

while(first<=last){

if (array[middle]<search)

first=middle+1;

else if (array[middle]==search){

printf(“%d is present at index %d. \n”, search, middle+1);

break;

}

Else

Last=middle-1;

Middle=(first+last)/2;

}

}

Output:-

Enter number of elements:

5

Enter 5 integers:

1  
9  
22  
24  
46

**Enter the value to find:**

24

24 is present at index 4.