**Practical No: 07**

**Write the program for the following:**

1. **write a program to create the tree and display the elements.**

Code:

#include<iostream>

#include<stdlib.h>

using namespace std;

struct node{

int data;

struct node \*left;

struct node \*right;

};

struct node \*cn(int n)

{

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

nn->data=n;

nn->left=NULL;

nn->right=NULL;

return nn;

}

void pt(struct node \*root)

{

if(root==NULL)

{

return;

}

cout<<root->data<<' ';

pt(root->left);

pt(root->right);

}

int main()

{

struct node \*root=cn(17);

root->left=cn(4);

root->right=cn(23);

root->left->left=cn(20);

cout<<"Tree using Link list: ";

pt(root);

cout<<endl;

}

Output:

**Tree using Link list: 17 4 20 23**

1. **Write a program to construct the binary tree.**

Code:

#include<iostream>

#include<stdlib.h>

using namespace std;

struct node{

int data;

struct node \*left;

struct node \*right;

};

struct node \*cn(int n)

{

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

nn->data=n;

nn->left=NULL;

nn->right=NULL;

return nn;

}

void pt(struct node \*root)

{

if(root==NULL)

{

return;

}

cout<<root->data<<' ';

pt(root->left);

pt(root->right);

}

int main()

{

struct node \*root=cn(17);

root->left=cn(4);

root->right=cn(23);

root->left->left=cn(20);

root->left->right=cn(19);

root->right->right=cn(19);

root->right->left=cn(9);

cout<<"Binary Tree using Link list: ";

pt(root);

cout<<endl;

}

Output:

**Binary Tree using Link list: 17 4 20 19 23 9 19**

1. **Write a program for inorder, postorder and preorder traversal of tree**

Code:

#include<iostream>

#include<stdlib.h>

using namespace std;

struct node{

int data;

struct node \*left;

struct node \*right;

};

struct node \*cn(int n)

{

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

nn->data=n;

nn->left=NULL;

nn->right=NULL;

return nn;

}

void pt(struct node \*root)

{

if(root==NULL)

{

return;

}

cout<<root->data<<' ';

pt(root->left);

pt(root->right);

}

void io(struct node \*root)

{

if(root==NULL)

{

return;

}

io(root->left);

cout<<root->data<<' ';

io(root->right);

}

void po(struct node \*root)

{

if(root==NULL)

{

return;

}

io(root->left);

io(root->right);

cout<<root->data<<' ';

}

void preo(struct node \*root)

{

if(root==NULL)

{

return;

}

cout<<root->data<<' ';

preo(root->left);

preo(root->right);

}

int main()

{

struct node \*root=cn(17);

root->left=cn(4);

root->right=cn(23);

root->left->left=cn(20);

root->left->right=cn(19);

root->right->right=cn(19);

root->right->left=cn(9);

cout<<"Binary Tree: ";

pt(root);

cout<<endl; cout<<"Inorder traversal: ";

io(root);

cout<<endl;

cout<<"Postorder traversal: ";

po(root);

cout<<endl;

cout<<"Preorder traversal: ";

preo(root);

}

Output:

Binary Tree: 17 4 20 19 23 9 19

Inorder traversal: 20 4 19 17 9 23 19

Postorder traversal: 20 4 19 9 23 19 17

Preorder traversal: 17 4 20 19 23 9 19