**1: Implementation of recursive function for tree traversal and Fibonacci.**

**1. A. Inorder recursive**

#include<iostream>

using namespace std;

struct node {

int data;

struct node \*left;

struct node \*right;

};

struct node \*createNode(int val) {

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

temp->data = val;

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

return temp;

}

void inorder(struct node \*root) {

if (root != NULL) {

inorder(root->left);

cout<<root->data<<" ";

inorder(root->right);

}

}

struct node\* insertNode(struct node\* node, int val) {

if (node == NULL) return createNode(val);

if (val < node->data)

node->left = insertNode(node->left, val);

else if (val > node->data)

node->right = insertNode(node->right, val);

return node;

}

int main() {

struct node \*root = NULL;

root = insertNode(root, 4);

insertNode(root, 5);

insertNode(root, 2);

insertNode(root, 9);

insertNode(root, 1);

insertNode(root, 3);

cout<<"In-Order traversal of the Binary Search Tree is: ";

inorder(root);

return 0;

}

**Output**

In-Order traversal of the Binary Search Tree is: 1 2 3 4 5 9

**1.B. Preorder recursive**

#include<iostream>

using namespace std;

struct node {

int data;

struct node \*left;

struct node \*right;

};

struct node \*createNode(int val) {

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

temp->data = val;

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

return temp;

}

void preorder(struct node \*root) {

if (root != NULL) {

cout<<root->data<<" ";

preorder(root->left);

preorder(root->right);

}

}

struct node\* insertNode(struct node\* node, int val) {

if (node == NULL) return createNode(val);

if (val < node->data)

node->left = insertNode(node->left, val);

else if (val > node->data)

node->right = insertNode(node->right, val);

return node;

}

int main() {

struct node \*root = NULL;

root = insertNode(root, 4);

insertNode(root, 5);

insertNode(root, 2);

insertNode(root, 9);

insertNode(root, 1);

insertNode(root, 3);

cout<<"Pre-Order traversal of the Binary Search Tree is: ";

preorder(root);

return 0;

}

**Output**

Pre-Order traversal of the Binary Search Tree is: 4 2 1 3 5 9

**1.C. Postorder recursive**

#include<iostream>

using namespace std;

struct node {

int data;

struct node \*left;

struct node \*right;

};

struct node \*createNode(int val) {

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

temp->data = val;

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

return temp;

}

void postorder(struct node \*root) {

if (root != NULL) {

postorder(root->left);

postorder(root->right);

cout<<root->data<<" ";

}

}

struct node\* insertNode(struct node\* node, int val) {

if (node == NULL) return createNode(val);

if (val < node->data)

node->left = insertNode(node->left, val);

else if (val > node->data)

node->right = insertNode(node->right, val);

return node;

}

int main() {

struct node \*root = NULL;

root = insertNode(root, 4);

insertNode(root, 5);

insertNode(root, 2);

insertNode(root, 9);

insertNode(root, 1);

insertNode(root, 3);

cout<<"Post-Order traversal of the Binary Search Tree is: ";

postorder(root);

return 0;

}

**Output**

Post-Order traversal of the Binary Search Tree is: 1 3 2 9 5 4

**1.D. Fibonacci using recurvice function**

#include <iostream>

using namespace std;

int fib(int x) {

if((x==1)||(x==0)) {

return(x);

}else {

return(fib(x-1)+fib(x-2));

}

}

int main() {

int x , i=0;

cout << "Enter the number of terms of series : ";

cin >> x;

cout << "\nFibonnaci Series : ";

while(i < x) {

cout << " " << fib(i);

i++;

}

return 0;

}

**Output**

Enter the number of terms of series : 15

Fibonnaci Series : 0 1 1 2 3 5 8 13 21 34 55 89 144 233 377