Name:Siddhi Vinod Pande Class: SYBCA
Roll no:-66 Batch:B2
Practical 13: Implementation of Binary Search Tree

```
#include<iostream.h>
#include<conio.h>

class node
{
   int info;
   node *left,*right;
   friend class BST;
}*root;

class BST
{
   node *pp,*p,*n,*t,*ptr;
   public:
      BST()
      {
         root=NULL;
         ptr=NULL;
         pp=NULL;
      }
      void create();
      void search();
      void deletion();
      void display(node *,int);
};

void BST::create()
{
   char ch;
   do
   {
      n=new node;
      cout<<"\n Enter The Element:";
      cin>>n->info;
      n->left=NULL;
      n->right=NULL;
      if(root==NULL)
         root=n;
      else
      {
         t=root;
         while(t!=NULL)
         {
            p=t;
            if(n->info<t->info)
```

```
                t=t->left;
            else
                t=t->right;
        }
        if(n->info<p->info)
            p->left=n;
        else
            p->right=n;
    }
    cout<<"\n Are You Want To Add Another(Y/y):";
    cin>>ch;
  } while(ch=='Y'||ch=='y');
}

void BST::search()
{
  int item,c=0;
  cout<<"\nEnter The Item To Be Search:";
  cin>>item;
  p=root;
  while(p!=NULL&&item!=p->info)
  {
    if(item<p->info)
    {
      p=p->left;
      c++;
    }
    else
    {
      p=p->right;
      c++;
    }
  }
  cout<<"\n Item Is Found At Level:" <<c+1;
  if(p==NULL)
  cout<<"\nItem Is Not Found.";
}

void BST::deletion()
{
  int item;
  cout<<"\n Enter The Item For Deletion:";
  cin>>item;
  p=root;
  while(p!=NULL&&p->info!=item)
  {
    pp=p;
    if(item<p->info)
    p=p->left;
    else
```

```
        p=p->right;
    }
    if(p==NULL)
    {
        cout<<"\n Item Is Not Found In The List.";
        return;
    }
    cout<<"\n Deleted Item Is:"<<p->info;
    if(p->left!=NULL&&p->right!=NULL)
    {
        node *s=p->left;
        node *ps=p;
        while(s->right!=NULL)
        {
            ps=s;
            s=s->right;
        }
        p->info=s->info;
        p=s;
        pp=ps;
    }
    node *c;
    if(p->left!=NULL)
        c=p->left;
    else
        c=p->right;
    if(p==root)
        root=c;
    else
    {
        if(p==pp->left)
            pp->left=c;
        else
            pp->right=c;
    }
    delete p;
}
void BST::display(node *ptr, int level)
{
     if (ptr != NULL)
     {
          display(ptr->right,level+1);
          cout<<"\n";
          if (ptr == root)
               cout<<"Root->:  ";
          else
          {
               for ( int i=0;i<level;i++)
                    cout<<"       ";
          }
```

```
        cout<<ptr->info;
        display(ptr->left, level+1);
    }
}

void main()
{
   int  ch;
   BST b;
   do
   {
      clrscr();
      cout<<"\n********** MENU*************";
      cout<<"\n1.Create\n2.Search\n3.Deletion\n4.Display \n5.Exit";
      cout<<"\nEnter Your Choice:";
      cin>>ch;
      switch(ch)
      {
         case 1:b.create();break;
         case 2:b.search();break;
         case 3:b.deletion();break;
         case 4:b.display(root,1);break;
         case 5:break;
         default: cout<<"\n Wrong Choice:";
      }
      getch();
   }while(ch!=5);
   getch();
}
```

OUTPUT:-

```
********** MENU*************
1.Create
2.Search
3.Deletion
4.Display
5.Exit
Enter Your Choice: 1
Enter The Element: 50
Are You Want To Add Another(Y/y): y
Enter The Element: 30
Are You Want To Add Another(Y/y): y
Enter The Element: 70
Are You Want To Add Another(Y/y): y
Enter The Element: 20
Are You Want To Add Another(Y/y): y
Enter The Element: 40
Are You Want To Add Another(Y/y): y
```

```
Enter The Element: 60
Are You Want To Add Another(Y/y): y
Enter The Element: 80
Are You Want To Add Another(Y/y): n

********** MENU*************
Enter Your Choice: 4
            80
        70
            60
Root->:  50
            40
        30
            20

********** MENU*************
Enter Your Choice: 2
Enter The Item To Be Search: 60
Item Is Found At Level: 3

********** MENU*************
Enter Your Choice: 3
Enter The Item For Deletion: 30
Deleted Item Is: 30

********** MENU*************
Enter Your Choice: 4
            80
        70
            60
Root->:  50
            40
        20

********** MENU*************
Enter Your Choice: 5
```

Name:Siddhi Vinod Pande Class: SYBCA
Roll no:-66 Batch:B2

Practical 14: Implementation of In-order, Pre-order and Post-order Traversals

```
#include<iostream.h>
#include<conio.h>

class node {
public:
   int info;
   node *left, *right;
} *root;

class BST {
public:
   BST() { root = NULL; }
   void create();
   void inorder(node *ptr);
   void preorder(node *ptr);
   void postorder(node *ptr);
};

void BST::create() {
   char ch;
   node *n, *t, *p;
   do {
      n = new node;
      cout << "\n Enter The Element: ";
      cin >> n->info;
      n->left = NULL;
      n->right = NULL;

      if (root == NULL)
         root = n;
      else {
         t = root;
         while (t != NULL) {
            p = t;
            if (n->info < t->info)
               t = t->left;
            else
               t = t->right;
         }
         if (n->info < p->info)
            p->left = n;
         else
            p->right = n;
      }
      cout << "\n Do you want to add another node (Y/N): ";
```

```
        cin >> ch;
    } while (ch == 'Y' || ch == 'y');
}

void BST::inorder(node *ptr) {
    if (ptr != NULL) {
        inorder(ptr->left);
        cout << ptr->info << "  ";
        inorder(ptr->right);
    }
}

void BST::preorder(node *ptr) {
    if (ptr != NULL) {
        cout << ptr->info << "  ";
        preorder(ptr->left);
        preorder(ptr->right);
    }
}

void BST::postorder(node *ptr) {
    if (ptr != NULL) {
        postorder(ptr->left);
        postorder(ptr->right);
        cout << ptr->info << "  ";
    }
}

void main() {
    clrscr();
    BST b;
    int ch;
    do {
        cout << "\n********** MENU **********";
        cout << "\n1.Create BST";
        cout << "\n2.In-order Traversal";
        cout << "\n3.Pre-order Traversal";
        cout << "\n4.Post-order Traversal";
        cout << "\n5.Exit";
        cout << "\nEnter Your Choice: ";
        cin >> ch;
        switch (ch) {
        case 1: b.create(); break;
        case 2: cout << "\nIn-order Traversal: "; b.inorder(root); break;
        case 3: cout << "\nPre-order Traversal: "; b.preorder(root); break;
        case 4: cout << "\nPost-order Traversal: "; b.postorder(root); break;
        }
    } while (ch != 5);
    getch();
}
```

OUTPUT:-

********** MENU **********
1.Create BST
2.In-order Traversal
3.Pre-order Traversal
4.Post-order Traversal
5.Exit
Enter Your Choice: 1
Enter The Element: 50
Do you want to add another node (Y/N): y
Enter The Element: 30
Do you want to add another node (Y/N): y
Enter The Element: 70
Do you want to add another node (Y/N): y
Enter The Element: 20
Do you want to add another node (Y/N): n

Enter Your Choice: 2
In-order Traversal: 20 30 50 70

Enter Your Choice: 3
Pre-order Traversal: 50 30 20 70

Enter Your Choice: 4
Post-order Traversal: 20 30 70 50

Enter Your Choice: 5

Name:Siddhi Vinod Pande Class: SYBCA
Roll no:-66 Batch:B2

Practical 15: Implementation of Graph Representation in Memory

```
#include<iostream.h>
#include<conio.h>

class Graph {
   int adj[10][10], n;
   char nodes[10];
public:
   void create();
   void display();
};

void Graph::create() {
   cout << "\nEnter number of nodes: ";
   cin >> n;
   cout << "\nEnter node names: ";
   for (int i = 0; i < n; i++)
      cin >> nodes[i];

   cout << "\nEnter adjacency matrix:\n";
   for (int i = 0; i < n; i++) {
      for (int j = 0; j < n; j++) {
         cin >> adj[i][j];
      }
   }
}

void Graph::display() {
   cout << "\nAdjacency Matrix Representation:\n";
   for (int i = 0; i < n; i++) {
      for (int j = 0; j < n; j++) {
         cout << adj[i][j] << " ";
      }
      cout << endl;
   }
}

void main() {
   clrscr();
   Graph g;
   g.create();
   g.display();
   getch();
}
```

OUTPUT:-

```
Enter number of nodes: 4
Enter node names: A B C D
Enter adjacency matrix:
0 1 1 0
1 0 0 1
1 0 0 1
0 1 1 0

Adjacency Matrix Representation:
0 1 1 0
1 0 0 1
1 0 0 1
0 1 1 0
```