# SSN College of Engineering, Kalavakkam

# Department of Computer Science and Engineering

# III Semester - CSE 'A ',’B’ & ‘C’

# UCS 1312 Data Structures Lab

# Academic Year: 2019-2020 Batch: 2018-2022

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**Name : Rahul Ram M**

**Class : CSE – B**

**Register Number : 185001121**

**Exercise:08**

**AIM:**

** To Create BST for the following entries Kumar, Anusha, Ram, Charan, Mohan, Karthika,**

**Chitra, Lakshmi, Abishek, Swetha, Tarun, Sanjana.**

** Display the tree elements in alphabetical order.**

** Find the grand parent of Lakshmi and Karthika.**

** Find the grand children of Charan.**

** Find the sibling of Swetha and Chitra**

** Delete Ram and display the resultant tree.**

**//main()**

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include"binarytree.h"

int main()

{

int ch=1;

char sname[25];

bst \*head,\*t;

printf("Press\n1.Insert name to the tree.\n2.Find the Grandparent.\n3.Find the Grandchild.\n4.Find the Siblings.\n5.Delete a name.\n6.Display the tree.\n7.Exit.\nEnter Choice:");

scanf("%d",&ch);

while(ch!=7)

{

switch(ch)

{

case 1:

{

head = createtree();

printf("Displaying tree in inorder transversal..\n");

display(head);

break;

}

case 2:

{

printf("Enter the name of the grandchild:");

scanf("%s",sname);

t=findGP(sname,head);

if(t!=NULL)

printf("%s is the GrandParent\n",t->name);

else

printf("%s has no GrandParent\n",sname);

break;

}

case 3:

{

printf("Enter the name of the grandparent:");

scanf("%s",sname);

t=findPos(sname,head);

if(t!=NULL)

{

printf("GrandChildrens : ");

findGC(t);

printf("\n");

}

else

{

printf("Name not Found!\n");

}

break;

}

case 4:

{

printf("Enter the name of the Sibling:");

scanf("%s",sname);

t=findSiblings(sname,head);

if(t!=NULL)

{

printf("\nSibling of %s\n",sname);

if(strcmp(sname,t->right->name)==0)

printf("%s\n",t->left->name);

else if(strcmp(sname,t->left->name)==0)

printf("%s\n",t->right->name);

else

printf("%s has no sibling\n",sname);

}

else

printf("%s is not found\n",sname);

break;

}

case 5:

{

printf("Enter the name to be deleted:");

scanf("%s",sname);

head=deleteNode(sname,head);

break;

}

case 6:

{

printf("Displaying tree in inorder transversal..\n");

display(head);

break;

}

default:printf("Invalid Input!\n");

}

printf("Press\n1.Insert name to the tree.\n2.Find the Grandparent.\n3.Find the Grandchild.\n4.Find the Siblings.\n5.Delete a name.\n6.Display the tree.\n7.Exit.\nEnter Choice:");

scanf("%d",&ch);

}

return 0;

}

**//binarytree.h**

typedef struct binarytree

{

char name[25];

struct binarytree \*right,\*left;

}bst;

bst\* createtree();

bst\* insert(char sname[],bst \*hd1);

void display(bst\* tree);

bst\* findGP(char name[],bst \*hd);

void findGC(bst\* hd);

bst\* findSiblings(char name[],bst \*hd);

bst\* deleteNode(char name[],bst \*hd);

bst\* findMin(bst \*hd);

bst\* findPos(char name[],bst\* hd);

//bst\* findPos2(char name[],bst\* hd);

bst\* createtree()

{

char tempname[25];

int ch=1;

bst \*hd = NULL;

while(ch==1)

{

printf("Enter the name :");

scanf("%s",tempname);

hd=insert(tempname,hd);

printf("Continue(1.Yes/2.No) :");

scanf("%d",&ch);

}

return hd;

}

bst\* insert(char sname[],bst \*hd1)

{

if(hd1==NULL)

{

hd1=(bst \*)malloc(sizeof(bst));

strcpy(hd1->name,sname);

hd1->left=hd1->right=NULL;

}

else if(strcmp(hd1->name,sname)<0)

hd1->right=insert(sname,hd1->right);

else if(strcmp(hd1->name,sname)>0)

hd1->left=insert(sname,hd1->left);

else

printf("Name already used! Try Another..\n");

return hd1;

}

void display(bst\* tree)

{

if(tree!=NULL)

{

display(tree->left);

printf("%s\n",tree->name);

display(tree->right);

}

}

bst\* findGP(char aname[],bst \*hd)

{

if(hd==NULL)

return hd;

if(hd->left!=NULL)

{

if(hd->left->left!=NULL)

{

if(!strcmp(hd->left->left->name,aname))

return hd;

}

if(hd->left->right!=NULL)

{

if(!strcmp(hd->left->right->name,aname))

return hd;

}

}

if(hd->right!=NULL)

{

if(hd->right->left!=NULL)

{

if(!strcmp(hd->right->left->name,aname))

return hd;

}

if(hd->right->right!=NULL)

{

if(!strcmp(hd->right->right->name,aname))

return hd;

}

}

if(strcmp(aname,hd->name)<0)

hd=findGP(aname,hd->left);

else if(strcmp(aname,hd->name)>0)

hd=findGP(aname,hd->right);

}

void findGC(bst\* hd)

{

int s=0;

if(hd->left!=NULL && hd->left->left!=NULL)

{

s=1;

printf("%s, ",hd->left->left->name);

}

if(hd->left!=NULL && hd->left->right!=NULL)

{

s=1;

printf("%s, ",hd->left->right->name);

}

if(hd->right!=NULL && hd->right->left!=NULL)

{

s=1;

printf("%s, ",hd->right->left->name);

}

if(hd->right!=NULL && hd->right->right!=NULL)

{

s=1;

printf("%s, ",hd->right->right->name);

}

if(s==0)

{

printf("Not found!!!");

}

}

bst\* findSiblings(char name[],bst \*hd)

{

bst\* temp;

int f=0;

temp=(bst \*)malloc(sizeof(bst));

temp=findPos(name,hd);

if(temp!=NULL)

f=1;

if(f==1)

{

if(hd==NULL)

return NULL;

else if(strcmp(name,hd->left->name)<0)

return findSiblings(name,hd->left);

else if(strcmp(name,hd->right->name)>0)

return findSiblings(name,hd->right);

return hd;

}

else

printf("%s is not found\n",name);

}

bst\* deleteNode(char name[],bst \*hd)

{

bst \*tempnode;

if(hd==NULL)

{

printf("Name not found!\n");

}

else if(strcmp(name,hd->name)<0)

{

hd->left=deleteNode(name,hd->left);

}

else if(strcmp(name,hd->name)>0)

{

hd->right=deleteNode(name,hd->right);

}

else if(hd->left && hd->right)

{

tempnode=findMin(hd->right);

strcpy(hd->name,tempnode->name);

hd->right=deleteNode(hd->name,hd->right);

}

else

{

tempnode=hd;

if(hd->left==NULL)

{

hd=hd->right;

}

else if(hd->right==NULL)

{

hd=hd->left;

}

free(tempnode);

}

return hd;

}

bst\* findMin(bst \*hd)

{

if(hd==NULL)

return NULL;

else if(hd->left==NULL)

return hd;

else

return findMin(hd->left);

}

/\*bst\* findPos2(char name[],bst\* hd)

{

if(hd==NULL)

return NULL;

else if(hd->left!=NULL && strcmp(name,hd->left->name)==0)

return hd;

else if(hd->right!=NULL && strcmp(name,hd->right->name)==0)

return hd;

else if (strcmp(name,hd->name)<0)

return findPos(name,hd->left);

else if (strcmp(name,hd->name)>0)

return findPos(name,hd->right);

}\*/

bst\* findPos(char name[],bst\* hd)

{

if (hd==NULL)

return NULL;

else if (strcmp(name,hd->name)<0)

return findPos(name,hd->left);

else if (strcmp(name,hd->name)>0)

return findPos(name,hd->right);

else

return hd;

}

**//Input Output**

Press

1.Insert name to the tree.

2.Find the Grandparent.

3.Find the Grandchild.

4.Find the Siblings.

5.Delete a name.

6.Display the tree.

7.Exit.

Enter Choice:1

Enter the name :Kumar

Continue(1.Yes/2.No) :1

Enter the name :Anusha

Continue(1.Yes/2.No) :1

Enter the name :Ram

Continue(1.Yes/2.No) :1

Enter the name :Charan

Continue(1.Yes/2.No) :1

Enter the name :Mohan

Continue(1.Yes/2.No) :1

Enter the name :Karthika

Continue(1.Yes/2.No) :1

Enter the name :Chitra

Continue(1.Yes/2.No) :1

Enter the name :Lakshmi

Continue(1.Yes/2.No) :1

Enter the name :Abishek

Continue(1.Yes/2.No) :1

Enter the name :Swetha

Continue(1.Yes/2.No) :1

Enter the name :Tarun

Continue(1.Yes/2.No) :1

Enter the name :Sanjana

Continue(1.Yes/2.No) :2

Displaying tree in inorder transversal..

Abishek

Anusha

Charan

Chitra

Karthika

Kumar

Lakshmi

Mohan

Ram

Sanjana

Swetha

Tarun

Press

1.Insert name to the tree.

2.Find the Grandparent.

3.Find the Grandchild.

4.Find the Siblings.

5.Delete a name.

6.Display the tree.

7.Exit.

Enter Choice:2

Enter the name of the grandchild:Lakshmi

Ram is the GrandParent

Press

1.Insert name to the tree.

2.Find the Grandparent.

3.Find the Grandchild.

4.Find the Siblings.

5.Delete a name.

6.Display the tree.

7.Exit.

Enter Choice:2

Enter the name of the grandchild:Karthika

Anusha is the GrandParent

Press

1.Insert name to the tree.

2.Find the Grandparent.

3.Find the Grandchild.

4.Find the Siblings.

5.Delete a name.

6.Display the tree.

7.Exit.

Enter Choice:3

Enter the name of the grandparent:Charan

GrandChildrens : Chitra,

Press

1.Insert name to the tree.

2.Find the Grandparent.

3.Find the Grandchild.

4.Find the Siblings.

5.Delete a name.

6.Display the tree.

7.Exit.

Enter Choice:4

Enter the name of the Sibling:Swetha

Sibling of Swetha

Mohan

Press

1.Insert name to the tree.

2.Find the Grandparent.

3.Find the Grandchild.

4.Find the Siblings.

5.Delete a name.

6.Display the tree.

7.Exit.

Enter Choice:4

Enter the name of the Sibling:Chitra

Sibling of Chitra

Chitra has no sibling

Press

1.Insert name to the tree.

2.Find the Grandparent.

3.Find the Grandchild.

4.Find the Siblings.

5.Delete a name.

6.Display the tree.

7.Exit.

Enter Choice:5

Enter the name to be deleted:Ram

Press

1.Insert name to the tree.

2.Find the Grandparent.

3.Find the Grandchild.

4.Find the Siblings.

5.Delete a name.

6.Display the tree.

7.Exit.

Enter Choice:6

Displaying tree in inorder transversal..

Abishek

Anusha

Charan

Chitra

Karthika

Kumar

Lakshmi

Mohan

Sanjana

Swetha

Tarun

Press

1.Insert name to the tree.

2.Find the Grandparent.

3.Find the Grandchild.

4.Find the Siblings.

5.Delete a name.

6.Display the tree.

7.Exit.

Enter Choice:7