Institute of Engineering & Management
Department of Computer Science & Engineering
Data Structure Laboratory for 2nd year 3rd semester 2017
Code: CS 392

Date: 13/9/17

ASSIGNMENT-5

Problem-1

Problem Statement: Implementation of Binary Search Tree

Algorithm:

```
Step-1: START
Step-2: Create a structure of integer num and bst pointer left, right, mother.
Step-3: Create function alloc(), insert(), del(), search(), display(), find(), create(),
        ftree().
Step-4: For value of j switch between
            case 1:  insert()
            case 2:  display()
            case 3:  search()
            case 4:  del()
            defualt: print "wrong input"
Step-5: inside insert(), scan for number of elements to be inserted
Step-6: scan for the elements
Step-7: pass them to create() function
Step-8: Inside display(),
            call display( temp->left )
            print "temp->num"
            call display( temp->right )
Step-9: inside search(), scan fr the element to search
Step-10: pass the element to the find(), and store the address in a pointer
Step-11: if address is not NULL then print "found"
            else print "not found"
Step-12: inside del(), scan for the element to be deleted
Step-13: pass the element to find(), and store the address in a pointer
Step-14: replace the node by the leftmost descendent node of the node to be
            deleted
Step-15: if the node is terminal then only delete the node
Step-16: STOP
```

Source code:

```
#include <stdio.h>
#include <stdlib.h>

typedef struct node
{
      int num;
      struct node *left, *right, *mother;
} bst;

bst *head=NULL, *flag=NULL;

bst *alloc()
```

```
{
      return (bst *)malloc(sizeof(bst));
}
void insert();
void create( int *, int );
void display( bst * );
void ftree( bst * );
void search();
bst *find( int, bst * );
void del();

int main()
{
      int i,  j;
      printf("Enter the following commands\n '1' to insert\n
            '2' to display\n '3' to search\n '4' to
                        delete\n");
      do
      {
            printf("Enter the command\n");
            scanf("%d", &j);
            switch(j)
            {
                  case 1:     insert(); break;
                  case 2:     display( head ); break;
                  case 3:     search(); break;
                  case 4:     del(); break;
                  default: printf("wrong input\n");
            }
            printf("enter 1 to continue\n");
            scanf("%d", &i);
      } while(i==1);
      ftree( head );
      return 0;
}

void ftree( bst *temp )
{
      if(temp!=NULL)
      {
            ftree( temp->left );
            ftree( temp->right );
            free( temp );
      }
}

void insert()
{
      int count, i;
      printf("Enter the no of elements\n");
      scanf("%d", &count);
      int elm[count];
      printf("enter the elements separated by spaces\n");
      for(i=0;i<count;i++)
      {
            scanf("%d", &elm[i]);
      }
      create( elm, count );
```

```
}

void create( int *elm, int count )
{
      bst *temp, *new;
      int i;
      temp=head;
      for(i=0;i<count;i++,temp=head)
      {
            new=alloc(); new->left=new->right=NULL;
            if(head==NULL)
            {
                  head=new;
                  head->mother=NULL;
                  head->num=elm[i];
                  continue;
            }
            while(1)
            {
                  if(temp->left!=NULL && temp->right!=NULL)
                  {
                        if( elm[i]<temp->num )
                              temp=temp->left;
                        else if( elm[i]>temp->num )
                              temp=temp->right;
                  }
                  else if( elm[i]<temp->num )
                  {
                        if(temp->left==NULL)
                        {
                              new->num=elm[i];
                              temp->left=new;
                              new->mother=temp;
                              break;
                        }
                        else temp=temp->left;
                  }
                  else if( elm[i]>temp->num )
                  {
                        if(temp->right==NULL)
                        {
                              new->num=elm[i];
                              temp->right=new;
                              new->mother=temp;
                              break;
                        }
                        else temp=temp->right;
                  }
            }
      }
}

void search()
{
      printf("Enter the element to search ");
      int elm;
      scanf(" %d", &elm); flag=NULL;
      bst *temp=find( elm, head );
```

```
	if( temp==NULL )
		printf("No such element found\n");
	else printf("%d is Found\n",temp->num);
}

bst *find(int elm, bst *temp)
{
	if(temp!=NULL && flag==NULL)
	{
		find(elm, temp->left);
		find(elm, temp->right);
		if(elm==temp->num)
			flag=temp;
	}
	return flag;
}

void del()
{
	printf("enter the element to delete ");
	int elm;
	scanf(" %d",&elm ); flag=NULL;
	bst *node=find( elm, head ), *temp;
	if(node==NULL)
	{
		printf("No such element found\n"); return;
	}
	if(node->left!=NULL)
	{
		temp=node->left;
		while(temp->left!=NULL)
			temp=temp->left;
		temp->mother->left=temp->right;
		if(temp->right!=NULL)
			temp->right->mother=temp->mother;
		node->num=temp->num;
		free(temp);
	}
	else{
		if(node->right==NULL)
		{
			if(node->mother->left==node)
				node->mother->left=NULL;
			else node->mother->right=NULL;
		}
		else{
			if(node->mother->left==node)
				node->mother->left=node->right;
			else node->mother->right=node->right;
			node->right->mother=node->mother;
		}
		free(node);
	}
}

void display( bst *temp )
{
	if( temp!=NULL )
```

```
        {
              display( temp->left );
              printf("%d, ",temp->num);
              display( temp->right );
        }
  }
```

Input/Output:

```
Enter the following commands
 '1' to insert
 '2' to display
 '3' to search
 '4' to delete
Enter the command
1
Enter the no of elements
5
enter the elements separated by spaces
6 7 3 8 1
enter 1 to continue
1
Enter the command
2
1, 3, 6, 7, 8, enter 1 to continue
1
Enter the command
3
Enter the element to search 3
3 is Found
enter 1 to continue
1
Enter the command
4
enter the element to delete 7
enter 1 to continue
1
Enter the command
2
1, 3, 6, 8, enter 1 to continue
0
```