EED 1010 ALGORITHMS & PROGRAMMING

LAB#10

Name: Enes

Surname: Erten

Number: 2018502036

Date: 13.05.2020

Task 1 : Write function binaryTreeSearch that attempts to locate a specified value in a binary search tree. The function should take as arguments a pointer to the root node of the binary tree and a search key to be located. If the node containing the search key is found, the function should return a pointer to that node; otherwise, the function should return a NULL pointer.

https://github.com/EnesErten/chowtoprogram/blob/master/labtask12%201

click to see the code in github

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
//include libraries

//defining new variable type
//left and right pointers make
//as a tree the variable type
struct treenode{
   struct treenode *left;
   int data;
   struct treenode *right;
};
//changinig names by typedef
typedef struct treenode treenode;
typedef treenode *treenodeptr;
//functions prototype
void insertnode(treenodeptr *a,int num);
```

```c
treenodeptr binarysearchtree(treenodeptr a,int num);
void inorder(treenodeptr a);
//insert an integer to tree
void insertnode(treenodeptr *a,int num)
{
    if(*a==NULL)
    {//if first elemnt
        *a=(treenode *)malloc(sizeof(treenode));
//dynamic memory
        if(*a!=NULL)
        {
            (*a)->data=num;
            (*a)->right=NULL;
            (*a)->left=NULL;
        }//assign num and NULLs

        else
            printf("No memory available!\n");
    }

    else
    {//if is not first elemnt
        //check num is big or data
        if(num<(*a)->data)
            insertnode(&((*a)->left),num);
            //recursievly call look for the left side of tree
        else
        {
            if(num>(*a)->data)
                insertnode(&((*a)->right),num);
                //recursievly call look for the right side of tree
            else
                printf("dup\n");
        }//if same num is assign to tree
    }
}
//this function is searching
//a num is exist in tree or not
treenodeptr binarysearchtree(treenodeptr a,int num)
{
    if(a==NULL)
        return NULL;
//if not exist return NULL
    else if(a->data==num)
        return a;
//if exist return tree pointer(which keeps node &)
    else if(num<a->data)
        binarysearchtree(a->left,num);
//recursievly call searching on left side num<a->data
    else if(num>a->data)
        binarysearchtree(a->right,num);
}
//print tree to respect to the their
//ascending order
void inorder(treenodeptr a)
{
    if(a!=NULL)
    {
        inorder(a->left);//recursievly call
        printf("%3d",a->data);//display data
```

```
        inorder(a->right);//recursievly call
    }
}

main()
{
    int n,i,a,l;
    treenodeptr aptr=NULL,bptr;

    printf("Enter how many number you want to enter to tree:");
    scanf("%d",&n);
//take data from stdin to loop

    for(i=0;i<n;i++)
    {//for loop
        printf("Enter number:");
        scanf("%d",&a);
        insertnode(&aptr,a);
        //take data from stdin to assign
        //assign to the tree
    }

    printf("The values being inserted in the tree are:\n");
    inorder(aptr);
//print tree to respect ascending order
    printf("\nEnter a key value to be searched:");
    scanf("%d",&l);
//take data from stdin to search a key
    bptr=binarysearchtree(aptr,l);
//send key and pointer
    if(bptr==NULL)
        printf("%d is not found in the binary tree!\n",l);
//if pointer is equal to NULL means key is not exist in tree
    else
        printf("%d is found in the binary tree node address %p",bptr->data,&(bptr->data));
//if pointer different from NULL means key is exist in tree
}
```

The Outputs

```
C:\Users\Enes\OneDrive\Documents\labtask12.exe
Enter how many number you want to enter to tree:4
Enter number:1
Enter number:2
Enter number:3
Enter number:3
dup
The values being inserted in the tree are:
  1  2  3
Enter a key value to be searched:3
3 is found in the binary tree node address 006659BC
--------------------------------
Process exited after 9.766 seconds with return value 0
Press any key to continue . . .
```

```
C:\Users\Enes\OneDrive\Documents\labtask12.exe
Enter how many number you want to enter to tree:4
Enter number:11
Enter number:13
Enter number:17
Enter number:19
The values being inserted in the tree are:
 11 13 17 19
Enter a key value to be searched:17
17 is found in the binary tree node address 008759BC
--------------------------------
Process exited after 15.87 seconds with return value 0
Press any key to continue . . .
```

```
C:\Users\Enes\OneDrive\Documents\labtask12.exe
Enter how many number you want to enter to tree:3
Enter number:1
Enter number:1
dup
Enter number:2
The values being inserted in the tree are:
  1  2
Enter a key value to be searched:3
3 is not found in the binary tree!

--------------------------------
Process exited after 6.924 seconds with return value 0
Press any key to continue . . .
```

```
C:\Users\Enes\OneDrive\Documents\labtask12.exe
Enter how many number you want to enter to tree:3
Enter number:23
Enter number:29
Enter number:31
The values being inserted in the tree are:
 23 29 31
Enter a key value to be searched:32
32 is not found in the binary tree!

--------------------------------
Process exited after 19.24 seconds with return value 0
Press any key to continue . . .
```

Task 2 : Write a program which calls the function in part (1) to search a key value in a binary search tree. Your program will first create a binary search tree with 15 nodes.The data values of the nodes are random integers from the range [1, 20]. (NOTE: if the same integer is created randomly, it will not be inserted, therefore nodes in the tree will be less than 15.) The data data values will be displayed on the screen. Then the user will enter the key value. Your program will call binaryTreeSearch() function. If the key is not found, it will display a message. If the key is found, it will display another message with key value and the address of the node. Sample program outputs are given below:

to see the code in the github.

https://github.com/EnesErten/chowtoprogram/blob/master/labtask2

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
//include libraries

//defining new variable type
//left and right pointers make
//as a tree the variable type
struct treenode{
   struct treenode *left;
   int data;
   struct treenode *right;
};
//changinig names by typedef
typedef struct treenode treenode;
typedef treenode *treenodeptr;
//functions prototype
void insertnode(treenodeptr *a,int num);
treenodeptr binarysearchtree(treenodeptr a,int num);

//insert an integer to tree
void insertnode(treenodeptr *a,int num)
{
   if(*a==NULL)
   {//if first elemnt
      *a=(treenode *)malloc(sizeof(treenode));
//dynamic memory
      if(*a!=NULL)
      {
         (*a)->data=num;
         (*a)->right=NULL;
         (*a)->left=NULL;
      }//assign num and NULLs

      else
         printf("No memory available!\n");
   }

   else
```

```
    {//if is not first elemnt
        //check num is big or data
        if(num<(*a)->data)
            insertnode(&((*a)->left),num);
            //recursievly call look for the left side of tree
        else
        {
            if(num>(*a)->data)
                insertnode(&((*a)->right),num);
                //recursievly call look for the right side of tree
            else
                printf("dup");
        }//if same num is assign to tree
    }
}
//this function is searching
//a num is exist in tree or not
treenodeptr binarysearchtree(treenodeptr a,int num)
{
    if(a==NULL)
        return NULL;
//if not exist return NULL
    else if(a->data==num)
        return a;
//if exist return tree pointer(which keeps node &)
    else if(num<a->data)
        binarysearchtree(a->left,num);
//recursievly call searching on left side num<a->data
    else if(num>a->data)
        binarysearchtree(a->right,num);
}
//print tree to respect to the their
//ascending order

main()
{
    int n,i,a,l;
    treenodeptr aptr=NULL,bptr;

    srand(time(NULL));

    printf("The values being inserted in the tree are:\n ");

    for(i=0;i<15;i++)
    {//for loop
        a=rand()%20+1;
        printf("%d",a);
        insertnode(&aptr,a);
        printf("  ");
        //assign to the tree
    }

    printf("\nEnter a key value to be searched:");
    scanf("%d",&l);
//take data from stdin to search a key
    bptr=binarysearchtree(aptr,l);
//send key and pointer
    if(bptr==NULL)
        printf("%d is not found in the binary tree!\n",l);
//if pointer is equal to NULL means key is not exist in tree
```

```
  else
      printf("%d is found in the binary tree node address %p",bptr->data,&(bptr->data));
//if pointer different from NULL means key is exist in tree
}
```

The Outputs