课堂代码作业

1. 二叉树求和

【题目】:

自己实现如图 1 的二叉树求和程序，看该求和函数有什么问题。

图 1 作业 1 的题目

【分析】:

通过对函数进行分析，该递归语句“Sumbt(bt->lchild)”，当进行到最后一个左孩子时，该节点没有右孩子，所以执行 Sumbt(bt->rchild)会出现问题，因为没有定义当节点为 NULL 的情况，导致程序不能运行。

【解决】:

在 Sumbt 函数内部添加节点为 NULL 的情况，具体程序如下：

```
int Sumbt(BTNode *bt)
{
    if(bt == NULL){ //添加节点为空的情况
        return 0;
    }else{

        if(bt->lchild== NULL && bt->rchild== NULL)
        {
            return bt->data;
        }else{
            return Sumbt(bt->lchild) + Sumbt(bt->rchild)+bt->data;
        }
        }
}
```

主程序定义一个 7 个节点的二叉搜索树，如图 2，运行求和结果如图 3。

```c
int main()
{

    int arr[7]={6,3,8,2,5,1,7};
    Tree tree;
    tree.root = NULL;

    int i;
    for(i=0;i<7;i++){
        insert(&tree,arr[i]); //把值插入到节点中以构造二叉树
    }
    int sum = 0;
    sum = Sumbt(tree.root); //求二叉树的和
    printf("二叉树的和为：\n%d",sum);

     return 0;
```

图 2 main 函数

【运行结果】:

```
PS D:\VS_code_content\C>  & 'c:\Use
Engine-Out-2va5ytyk.xti' '--stderr=
二叉树的和为：
32
```

图 3 运行结果

可以看到，二叉树{6,3,8,2,5,1,7}的求和结果为 32，正确。本题完整代码(4 个)见文末附录或源程序压缩包 code.z 文件。

2. 折半查找

【题目】:

书写折半查找程序并测试。

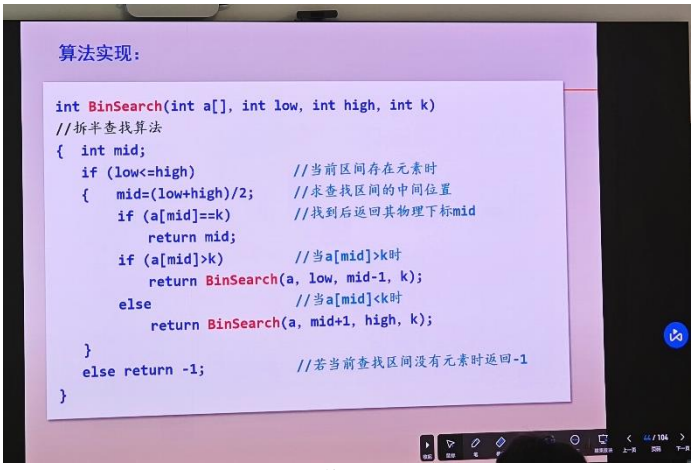

图 4 作业 2 的题目

【分析】:

通过对函数进行分析，BinSearch 函数实现对顺序序列中某一个值的折半查找，如图 5 所示思想。

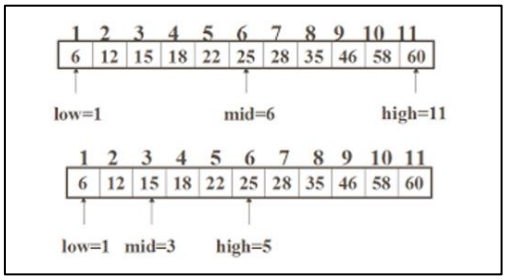

图 5 折半查找思想

【编写】:

参照图 4 代码，编写如下：

```
int BinSearch(int a[], int low, int high, int k) //折半查找算法
{

int mid;
if(low<=high) //当前区间有元素
{
mid = (low + high) / 2;

if(a[mid] == k)

return mid;

if(a[mid]  > k)

return BinSearch(a,low, mid - 1, k);

else

return BinSearch(a,mid + 1, high, k);

}
else {

//  printf("没有查找到该值\n");
return -1;

}          //若当前查找区间没有元素时返回-1
}
```

【运行结果】:

测试奇数长度(图 6-1)和偶数(图 6-2)长度数据链，进行三个值的查找，前两个值在数据链中，第三个值则不存在其中。运行结果如图 6-1、图 6-2 所示。

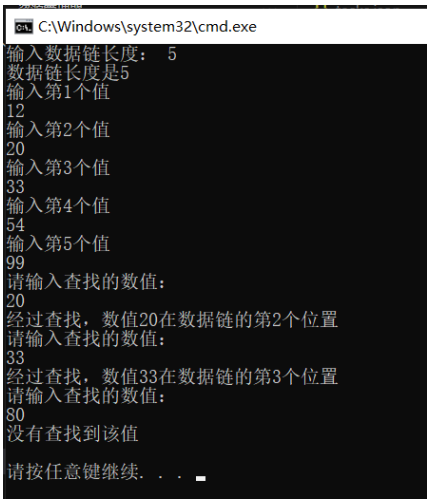

图 6-1 奇数长度(5)的结果

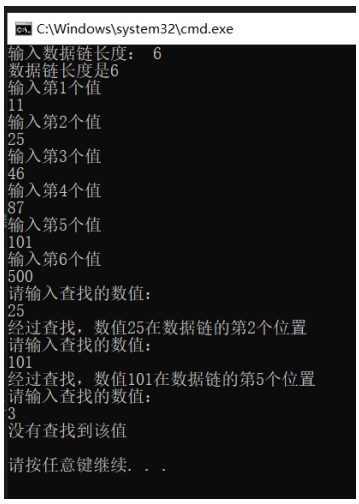

图 6-2 偶数长度(6)的结果

可以看到，两次结果查找均正确。本题完整代码(2 个)见文末附录或源程序压缩包 code.z 文件。

附录

（1）二叉树程序：

程序 1：BinaryTree.h	运行环境：VS code

```
%% 算法设计课报告(刘静老师)
% Time     : 2024-03-11
% Author   : 张炜
% Number   :23011210855

// 该程序仅仅定义一个二叉树
typedef struct BNode
{
    /* data */
    int data;
    struct BNode *lchild,*rchild ;
}BTNode;   //二叉链节点类型
```

程序 2：Tree.h	运行环境：VS code

```
// 使用定义的二叉树结构体，定义它的根节点，方便用根节点查找这棵树
#include"BinaryTree.h"

typedef struct Tree
{
    /* data */
    BTNode* root;
}Tree;
```

程序 3：Fun.h	运行环境：VS code

```
// 定义需要使用的函数
#include "Tree.h"
#include <stdlib.h>
#include<stdio.h>

void insert(Tree* tree, int value){   //构建 BST 二叉搜索树
    BTNode* node = (BTNode* )malloc(sizeof(BTNode));
    node->data   = value;
    node->lchild = NULL;
    node->rchild = NULL;
```

```
    if(tree->root == NULL ){
        tree->root = node;
    }
    else{
        BTNode* temp = tree->root;
        while ((temp != NULL))
        {
            /* code */
            if(value < temp->data){
                if(temp->lchild == NULL){
                    temp->lchild = node;
                    return;
                }else{
                    temp = temp->lchild;
                }
            }
            else{
                if(temp->rchild == NULL){
                    temp->rchild = node;
                    return;
                }else{
                    temp = temp->rchild;
                }
            }
        }

    }
}

//返回数的节点之和
int Sumbt(BTNode *bt)

{
    if(bt == NULL){
        return 0;
    }else{

        if(bt->lchild== NULL && bt->rchild== NULL)
        {
            return bt->data;
        }else{
            return Sumbt(bt->lchild) + Sumbt(bt->rchild)+bt->data;
        }
```

```
        }
}
```

程序 4：run.c	运行环境：VS code

```
// main 函数执行
#include<stdio.h>
// #include"BinaryTree.h"
#include"Fun.h"

int Sumbt(BTNode *bt);
int main()
{

   int arr[7]={6,3,8,2,5,1,7};
   Tree tree;
   tree.root = NULL;

   int i;
   for(i=0;i<7;i++){
        insert(&tree,arr[i]); //把值插入到节点中以构造二叉树
   }
   int sum = 0;
   sum = Sumbt(tree.root); //求二叉树的和
   printf("二叉树的和为: \n%d",sum);

    return 0;
}
```

（2）折半查找程序：

程序 1：myFun.h	运行环境：VS code

```
%% 算法设计课报告(刘静老师)
% Time     : 2024-03-24
% Author   : 张炜
% Number   :23011210855

// 折半查找函数
int BinSearch(int a[], int low, int high, int k) //折半查找算法
{
    int mid;
```

```
    if(low<=high) //当前区间有元素
    {
        mid = (low + high) / 2;

        if(a[mid] == k)
            return mid;
        if(a[mid]  > k)
            return BinSearch(a,low, mid - 1, k);
        else
            return BinSearch(a,mid + 1, high, k);
    }
    else {
      //  printf("没有查找到该值\n");
        return -1;
             }          //若当前查找区间没有元素时返回-1
}
```

程序 2：run.c	运行环境：VS code

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

// main 函数执行
int main(){

    int input;  printf("输入数据链长度:  ");
    scanf("%d",&input);
    int a[input];
    printf("数据链长度是%d\n",input);

     for (int i = 0; i < input; i++){//循环输入数据链的值

        printf("输入第%d 个值\n", i+1);
        scanf("%d", &a[i]);

    }
     for (int i = 0; i < 3; i++) //循环三次数据链查找
    {
        int k1 ;
        printf("请输入查找的数值: \n");
        scanf("%d", &k1);
        if (BinSearch(a,0,input,k1) != -1)
```

```
        printf("经过查找，数值%d 在数据链的第%d 个位置
\n",k1,BinSearch(a,0,input,k1) + 1);

        else

        printf("没有查找到该值\n");
    }

    return 0;
   // int arry[]={0,1,2,3};
}
```