

重 庆 大 学

学 生 实 验 报 告

实验课程名称 数据结构与算法

开课实验室 DS1501

学 院 软件学院 年级 2021 专业班 软件 X 班

学 生 姓 名 XXX 学 号 2021XXXX

开 课 时 间 2022 至 2023 学年第 1 学期

总 成 绩	
教师签名	XXX

《数据结构与算法》实验报告

开课实验室：DS1501

2022 年 10 月 20 日

学院	软件学院	年级、专业、班	2021 级软件工 程 X 班	姓名	XXX	成绩	
课程 名称	数据结构与算法	实验项目 名 称	2022-2023 学年第一学期数 据结构与算法上机练习 002	指导教师	XXX		
教 师 评 语	教师签名： 年 月 日						

一、实验目的

- 请复习教材第 5 章二叉树知识点，包括二叉树、二叉检索树、堆与优先队列的概念和实现方法。
- 实现教材第 5 章 115 页所述最大堆建堆的 Buildheap/Siftdown 算法。以教材 126 页习题 5.25 所提供的输入作为案例检验算法正确性。
- 完成教材 126 页习题 5.18 和 5.19。

二、使用仪器、材料

PC 微机；
Windows 操作系统，VS2022 编译环境；

三、实验步骤

- 1、实现教材第 5 章 115 页所述最大堆建堆的 Buildheap/Siftdown 算法。以教材 126 页习题 5.25 所提供的输入作为案例检验算法正确性；
- 2、习题 5.18 的解答；
- 3、习题 5.19 的解答；
- 4、完成实验报告并提交。

四、实验过程原始记录(数据、图表、计算等)

1. 实现教材第 5 章 115 页所述最大堆建堆的 Buildheap/Sift-down 算法。以教材 126 页习题 5.25 所提供的输入作为案例检验算法正确性;

```
1  #include<iostream>
2  #include <assert.h>
3  using namespace std;
4
5  template<typename E>class heap  <T> 提供 IntelliSense 的示例模板参数
6  {
7  private:
8      E* Heap;
9      int maxsize;
10     int n;
11
12     void sift-down(int pos)
13     {
14         while (!isLeaf(pos))
15         {
16             int j = leftchild(pos);
17             int rc = rightchild(pos);
18             if ((rc < n) && (Heap[rc] > Heap[j]))
19                 j = rc;
20             if (Heap[pos] > Heap[j])
21                 return;
22             swap(Heap, pos, j);
23             pos = j;
24         }
25     }
26
27 public:
28
29     inline void swap(int* Heap, int i, int j) {
30         int temp = Heap[i];
31         Heap[i] = Heap[j];
32         Heap[j] = temp;
33     };
34
```

```
35 heap(E* h, int num, int max)
36 {
37     Heap = h;
38     n = num;
39     maxsize = max;
40     buildHeap();
41 };
42
43 int size() const
44 {
45     return n;
46 };
47
48 bool isLeaf(int pos) const
49 {
50     return (pos >= n / 2) && (pos < n);
51 }
52
53 int leftchild(int pos) const
54 {
55     return 2 * pos + 1;
56 }
57
58 int rightchild(int pos) const
59 {
60     return 2 * pos + 2;
61 }
62
63 int parent(int pos) const
64 {
65     return (pos - 1) / 2;
66 }
67
```

```

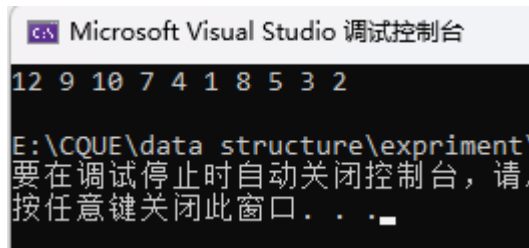
68 void buildHeap()
69 {
70     for (int i = n / 2 - 1; i >= 0; i--)
71     {
72         siftDown(i);
73     }
74 }
75
76 E removefirst()
77 {
78     swap(Heap, 0, --n);
79     if (n != 0) siftDown(0);
80     return Heap[n];
81 }
82
83 E remove(int pos)
84 {
85     if (pos == (n - 1)) n--;
86     else
87     {
88         swap(Heap, pos, --n);
89         while ((pos != 0) && (Heap[pos] > Heap[parent(pos)]))
90         {
91             swap(Heap, pos, parent(pos));
92             pos = parent(pos);
93         }
94         if (n != 0)
95             siftDown(pos);
96     };
97     return Heap[n];
98 };
99
100 void prt()
101 {
102     for (int i = 0; i < n; i++)
103     {
104         cout << Heap[i] << ' ';
105     }
106 };
107
108 };

```

以教材 126 页习题 5.25 所提供的输入作为案例：

```
110 int main()
111 {
112     int test[10]={10, 5, 12, 3, 2, 1, 8, 7, 9, 4};
113     heap<int> heap1(test, 10, 20);
114     heap1.prt();
115     cout << endl;
116     return 0;
117 }
```

结果：



```
Microsoft Visual Studio 调试控制台
12 9 10 7 4 1 8 5 3 2
E:\CQUE\data structure\expriment\
要在调试停止时自动关闭控制台，请
按任意键关闭此窗口。 . . .
```

2. 习题 5.18 的解答；

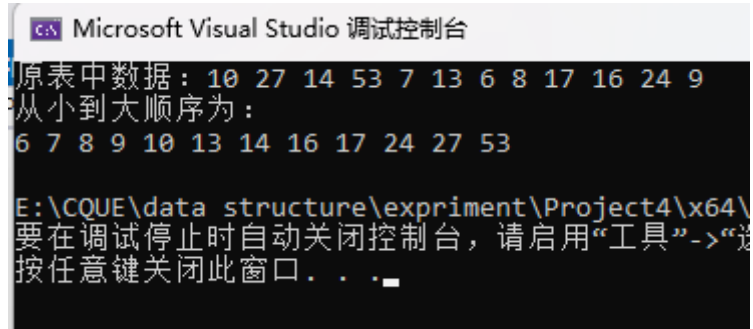
关键代码：

```
void reorder(BST T)
{
    if (NULL!=T)
    {
        reorder(T->lChild);
        cout<<T->key<<" ";
        reorder(T->rChild);
    }
}
```

主程序：

```
int main()
{
    int test[12] = { 10, 27, 14, 53, 7, 13, 6, 8, 17, 16, 24, 9 };
    int n = 12;
    cout << "原表中数据： ";
    for (int i = 0; i < n; i++)
    {
        cout << test[i] << " ";
    }
    cout << endl;
    BST T = NULL;
    createBST(T, test, n); //创建BST
    cout << "从小到大顺序为： " << endl;
    reorder(T);
    cout << endl;
    return 0;
}
```

结果:



```
Microsoft Visual Studio 调试控制台
原表中数据: 10 27 14 53 7 13 6 8 17 16 24 9
从小到大顺序为:
6 7 8 9 10 13 14 16 17 24 27 53

E:\CQUE\data structure\expriment\Project4\x64\
要在调试停止时自动关闭控制台, 请启用“工具”->“透
按任意键关闭此窗口. . .
```

3. 习题 5.19 的解答:

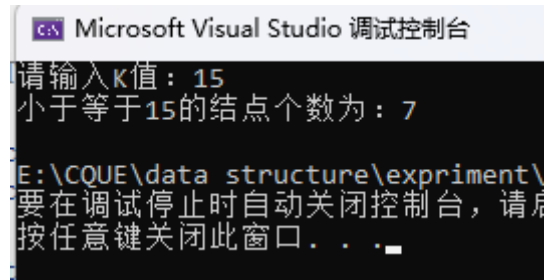
关键代码:

```
//递归函数smallcount找出小于等于K值的结点个数
int smallcount(BST T, int K)
{
    int count = 0;
    if (NULL != T)
    {
        if (T->key <= K)
        {
            count++;
            count = count + smallcount(T->rChild, K);
        }
        count = count + smallcount(T->lChild, K);
    }
    return count;
}
```

主程序:

```
int main()
{
    int test[12] = {10, 27, 14, 53, 7, 13, 6, 8, 17, 16, 24, 9};
    int n=12, K;
    cout<<"请输入K值: ";
    cin >> K;
    BST T = NULL;
    createBST(T, test, n); //创建BST
    int count;
    count=smallcount(T, K); //返回值小于等于K的结点个数
    cout<<"小于等于"<<K<<"的结点个数为: " << count << endl;
    return 0;
}
```

结果：



```
Microsoft Visual Studio 调试控制台
请输入K值：15
小于等于15的结点个数为：7
E:\CQUE\data structure\expriment\
要在调试停止时自动关闭控制台，请启
按任意键关闭此窗口..._
```

五、实验结果及分析

结果都已对应显示在原始数据记录中，结果都与预期的分析符合。