






# 01、顺序表删除最小值元素并用最后一个元素填补

24计算机考研成员一战成硕！

 题目描述：  
从顺序表中删除具有最小值的元素(假设唯一)并由函数返回被删元素的值。空出的位置由最后一个元素填补

 倒计时

## 1、知识点及难度



题解人：多动症男孩

难度：简单

知识点 & 注意点：

1. 当你要在顺序表的中间插入或删除一个元素时，你需要将插入点之后的元素向后移动（或删除点之后的元素向前移动）来保持顺序表的连续性。(本题c的代码通过修改表长，逻辑上实现了删除，题目没有特殊要求的话，只用在逻辑上删除就可以了)

时间复杂度：O(n)，仅遍历一遍顺序表，其中n为顺序表长度。

空间复杂度：O(1)，没有使用额外的辅助空间，仅常数级。

## 2、算法题

思路

1. 初始的时候最小值设置为顺序表中的第一个元素，并记录该元素的索引(下标)。
2. 从第二个元素开始遍历顺序表，如果比当前最小值更小的话，就更新当前最小值和其索引。

- 遍历完成后，将最小值保存下来，并将最后一个元素覆盖到被删除的位置上，实现空出的位置由最后一个元素填补。
- 最后返回被删除的值即可。

#### 基本实现-C++

```
1 #include <iostream>
2 #include <vector>
3 using namespace std;
4
5 int removeMinValue(vector<int>& sequence) {
6     if (sequence.empty()) {
7         cerr << "顺序表为空! " << endl;
8         return -1; // 返回-1表示错误
9     }
10
11     int minInd = 0;
12     int minVal = sequence[0];
13
14     for (int i = 1; i < sequence.size(); i++) {
15         // 比较获取最小值和其索引
16         if (sequence[i] < minVal) {
17             minVal = sequence[i];
18             minInd = i;
19         }
20     }
21
22     int deletedVal = sequence[minInd];
23     sequence[minInd] = sequence.back(); // 最后一个元素填补删除位
24     sequence.pop_back(); // 删除最后一个元素
25
26     return deletedVal; // 返回删除值
27 }
28
```

#### 基本实现C

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int removeMinValue(int* sequence, int* length) {
5     if (*length == 0) {
6         fprintf(stderr, "顺序表为空! \n");
7         return -1; // 返回-1表示错误
8     }
9
10    int minInd = 0;
11    int minVal = sequence[0];
12
13    for (int i = 1; i < *length; i++) {
14        if (sequence[i] < minVal) {
15            minVal = sequence[i];
16            minInd = i;
17        }
18    }
19
20    int deletedVal = sequence[minInd];
21    sequence[minInd] = sequence[*length - 1]; // 最后一个元素填补删除位
```

```
22     *length -= 1;          // 修改表长。
23     return deletedVal; // 返回删除值
24 }
25
```

3、一些宝子的题解

```
#include <stdio.h>
#include <stdlib.h>
#define MaxSize 10
// 定义顺序表
typedef struct
{
    int *data;
    int length;
} SeqList;
// 初始化顺序表
void InitList(SeqList &L)
{
    L.data = (int *)malloc(sizeof(int)*MaxSize);
    L.length = 0;
    printf("请输入元素: \n");
    for(int i=0; i<L.length; i++)
    {
        scanf("%d", &L.data[i]);
    }
}
// 删除
int deleteElem(SeqList &L)
{
    int p=0;
    int min=L.data[0];
    for(int j=1; j<L.length; j++)
    {
        if(L.data[j]<min)
        {
            min=L.data[j]; // 将被删除元素的值赋给min
            p=j;
        }
    }
    L.data[p]=L.data[L.length-1]; // 将最后一个元素移到p位置
    L.length--;
    return min;
}
int main(int argc, char *argv[])
{
    SeqList L;
    InitList(L);
    int Value = deleteElem(L);
    printf("删除的元素: %d\n", Value);
    printf("打印删除元素后的顺序表:");
    for (int i = 0; i < L.length; i++) {
        printf(" %d", L.data[i]);
    }
    return 0;
}
```

7月12日 03. 删除顺序表中最小值元素, 并用最后一个元素填补.

思路: 顺序遍历, 每次对比元素大小, 并记录下当前最小元素位置. 最后由末位元素填补并令表长减一.

```
int DeleMin (SeqList &L){
    int min = L.data[0]; // 首先默认第一个元素为最小值
    int m=0;
    for (int i=0; i<L.length; i++){
        if (L.data[i]<min){
            m=i;
            min = L.data[i]; // 更新最小元素
        } // if
    } // for
    L.data[m] = L.data[L.length-1]; // 由末位元素填补
    L.length--; // 表长减一
    return min; // 返回最小值
}
```

之所以从0开始遍历是考虑表长为1极端情况.

思路: 顺序遍历, 找出最小值并保存, 其中, 记录的第一个元素之后, 将最小值保存等待返回, 将最后一个元素填充到最小值位置. 因此, 需保存最小值 + 最小值位置.

②. 代码: // 顺序遍历.

```
define MaxSize 100;
typedef struct {
    int data[MaxSize];
    int length;
} SeqList;
// 删除函数
Status Dele_Min (SeqList &L) {
    int i; // 记录最小值位置
    int min; // 记录最小值
    min = L.data[0]; // 一开始, 将第一个元素作为最小值
    for (i=1; i<L.length; i++) { // 开始遍历顺序表
        if (L.data[i]<min) { // 如果发现比当前最小值还小, 则更新最小值
            min = L.data[i];
            i = i;
        }
    }
    L.data[i] = L.data[L.length-1]; // 替换最小值
    return min; // 返回最小值
}
```

长度忘记-1

4、总结



总结栏

- 1. 今天大家的问题基本都是关于代码上的一些书写标准, 对于题目的算法思想, 问题不大!

蓝蓝B站首页: [蓝蓝希望你上岸呀B站首页](#)

蓝蓝公众号: [算法训练营9分计划](#)

蓝蓝知识星球介绍: [关于知识星球的权益](#)

如何在星球打卡记录:

- 计算机考研数据结构算法专项day[1/60]:
- 学习内容: 最好能发出自己写的图片
- 遇到的问题: 如果无就不用写了
- 小结: 这部分一周写一次即可。