

HW2—顺序表构建

实验结果：

```
请输入您想要的长度: 5
请输入5个元素:
1 2 3 4 5
1. 判空 2. 获取长度 3. 按位查找数据 4. 按值查找数据索引(第一个有相同值) 5. 在某位置插入值 6. 删除i位置的元素 7. 打印输出表
按0退出
请输入您想执行的操作(仅输入编号):
0
表不为空
请输入您想执行的操作(仅输入编号):
2
表的长度为: 5
请输入您想执行的操作(仅输入编号):
3
请输入您想查找第几位的数据(1开始):
3
您查找的数据是: 3
请输入您想执行的操作(仅输入编号):
4
请输入您想查找的数据:
3
您想查找的数据在第3位
请输入您想执行的操作(仅输入编号):
5
请输入您想在第几位插入数据(1开始):
4
请输入您想插入的数据:
10
插入成功
请输入您想执行的操作(仅输入编号):
6
请输入您想删除第几位的数据(1开始):
4
删除成功
请输入您想执行的操作(仅输入编号):
7
顺序表:
4 5 10 3 2
请输入您想执行的操作(仅输入编号):
0
3: Algorithm_HW_23Y\HW\64\Release\HW.exe (进程 36716) 已退出, 代码为 0.
按任意键防止非正常关闭控制台, 请启用“工具”->“选项”->“调试”->“调试停止时自动关闭控制台”.
按任意键关闭此窗口. . .
```

```
请输入您想要的长度: 1
请输入1个元素:
1
1. 判空 2. 获取长度 3. 按位查找数据 4. 按值查找数据索引(第一个有相同值) 5. 在某位置插入值 6. 删除i位置的元素 7. 打印输出表
按0退出
请输入您想执行的操作(仅输入编号):
2
请输入您想查找第几位的数据(1开始):
1
索引过大, 返回-1
您查找的数据是: -1
请输入您想执行的操作(仅输入编号):
3
请输入您想查找的数据:
5
未找到您查找的数据
请输入您想执行的操作(仅输入编号):
5
请输入您想在第几位插入数据(1开始):
2
请输入您想插入的数据:
10
位置过大, 无法插入
插入失败
请输入您想执行的操作(仅输入编号):
6
请输入您想删除第几位的数据(1开始):
2
位置过大, 无法删除
删除失败
请输入您想执行的操作(仅输入编号):
7
请输入您想删除第几位的数据(1开始):
2
删除成功
请输入您想执行的操作(仅输入编号):
0
表为空
请输入您想执行的操作(仅输入编号):
7
顺序表为空
```

实验代码：

HWs.h

```
//作业 2: 顺序表
template <typename T, int maxSize>
class HW2_SequenceList {
public:
    HW2_SequenceList();
    ~HW2_SequenceList();

private:
    T data[maxSize];
    int length;

public:
    void process();

private:
    void createlist();
    bool isEmpty();
    int getLength();
    T getValue(int i);
    int locate(T x);
    bool insert(T x, int);
    bool erase(int i);

    void print();
};
```

HW2.cpp

```
#include "Hws.h"
#include <iostream>
using namespace std;

//初始化
template<typename T, int maxSize>
HW2_SequenceList<T,maxSize>::HW2_SequenceList(){
    length = 0;
}
template<typename T, int maxSize>
HW2_SequenceList<T, maxSize>::~~HW2_SequenceList() {
    delete[] data;
    length = 0;
}

//程序入口
template<typename T, int maxSize>
void HW2_SequenceList<T, maxSize>::process() {
    length = 0;
    createList();

    cout << "1.判空 2.获取长度 3.按位查找数据 4.按值查找数据索引(第一个有相同值) 5.在某位置插入值 6.删除 i 位置的值 7.打印输出表" << endl;
    cout << "按 0 退出" << endl;

    while (true) {
        cout<< endl << "请输入您想执行的操作(仅输入编号): " << endl;

        int option;
        cin >> option;
        if (option == 0) {
            exit(0);
        }

        int index; T x;

        switch (option){
        case 1:
            if (isEmpty()) cout << "表为空" << endl;
            else          cout << "表不为空" << endl;
            break;
```

```
case 2:
    cout << "表的长度为: " << getLength() << endl;
    break;

case 3:
    cout << "请输入您想查找第几位的数据(1 开始): " << endl;
    cin >> index;
    cout << "您查找的数据是: " << getValue(index) << endl;
    break;

case 4:
    cout << "请输入您想查找的数据: " << endl;
    cin >> x;
    index = locate(x);
    if (index == -1) cout << "未找到您要找的数据" << endl;
    else cout << "您想查找的数据在第" << locate(x) << "位" << endl;
    break;

case 5:
    cout << "请输入您想在第几位插入数据(1 开始): " << endl;
    cin >> index;
    cout << "请输入您想插入的数据: " << endl;
    cin >> x;
    if (insert(x, index)) cout << "插入成功" << endl;
    else cout << "插入失败" << endl;
    break;

case 6:
    cout << "请输入您想删除第几位的数据(1 开始): " << endl;
    cin >> index;
    if (erase(index)) cout << "删除成功" << endl;
    else cout << "删除失败" << endl;
    break;

case 7:
    print();

default:
    break;
}
}
```

```
//创建顺序表
template<typename T, int maxSize>
void HW2_SequenceList<T, maxSize>::createList() {
    cout << "请输入想要的长度: ";
    cin >> length;
    cout << "请输入" << length << "个元素: " << endl;
    for (int i = 0; i < length; i++) {
        cin >> data[i];
    }
}
```

```
//判断顺序表是否为空
template<typename T, int maxSize>
bool HW2_SequenceList<T, maxSize>::isEmpty() {
    return length == 0;
}
```

```
//获取顺序表长度
template<typename T, int maxSize>
int HW2_SequenceList<T, maxSize>::getLength() {
    return length;
}
```

```
//按位查找顺序表
template<typename T, int maxSize>
T HW2_SequenceList<T, maxSize>::getValue(int i) {
    if (i < 1) {
        cout << "索引过小, 返回-1" << endl;
        return -1;
    }
    else if (i > length) {
        cout << "索引过大, 返回-1" << endl;
        return -1;
    }
    else {
        return data[i - 1];
    }
}
```

```

//按值查找顺序表
template<typename T, int maxSize>
int HW2_SequenceList<T, maxSize>::locate(T x) {
    for (int i = 0; i < length; i++) {
        if (data[i] == x) {
            return i + 1;
        }
    }
    return -1;
}

//在 i 位置插入值
template<typename T, int maxSize>
bool HW2_SequenceList<T, maxSize>::insert(T x, int i) {
    if (length == maxSize) {
        cout << "顺序表已满，无法插入" << endl;
        return false;
    }
    if (i < 1) {
        cout << "位置过小，无法插入" << endl;
        return false;
    }
    else if (i > length + 1) {
        cout << "位置过大，无法插入" << endl;
        return false;
    }

    for (int j = length; j >= i; j--) {
        data[j] = data[j - 1];
    }
    data[i - 1] = x;
    length++;

    return true;
}

```

```

//删除顺序表 i 位置的值
template<typename T, int maxSize>
bool HW2_SequenceList<T, maxSize>::erase(int i) {
    if (i < 1) {
        cout << "位置过小，无法删除" << endl;
        return false;
    }
    else if (i > length) {
        cout << "位置过大，无法删除" << endl;
        return false;
    }

    for (int j = i - 1; j < length - 1; j++) {
        data[j] = data[j + 1];
    }
    length--;

    return true;
}

```

```

//打印输出顺序表
template<typename T, int maxSize>
void HW2_SequenceList<T, maxSize>::print() {
    if (length == 0) {
        cout << "顺序表为空" << endl;
    }
    else {
        cout << "顺序表: " << endl;
        for (int i = 0; i < length; i++) {
            cout << data[i] << ", ";
        }
        cout << endl;
    }
}

```

```

int main(int argc, const char* argv) {
    //作业 2: 顺序表
    const int maxSize = 100;
    HW2_SequenceList<int, maxSize> seqlist;
    seqlist.process();
}

```