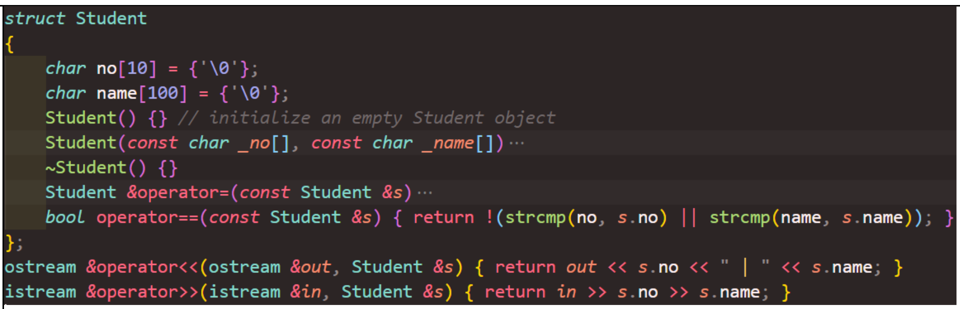
《数据结构》上机报告

2020 年 9 月 29 日

姓名： 林日中 学号： 1951112 班级： 10072602 得分：

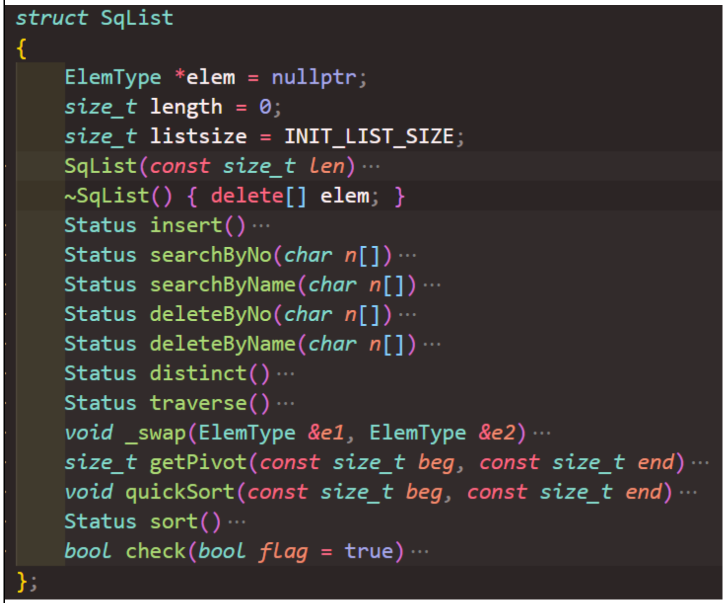
实验题目	顺序表实现学生管理系统	
问题描述	顺序表是指采用顺序存储结构的线性表，它利用内存中的一片连续存储区域存放表中的所有元素。可以根据需要对表中的所有数据进行访问，元素的插入和删除可以在表中的任何位置进行。 实验目的： 1、掌握线性表的定义及顺序表示； 2、掌握用顺序存储结构实现线性表的基本操作，如建立、查找、插入和删除以及去重等； 3、掌握顺序表的特点；	
基本要求	实验内容： （1）定义一个包含学生信息（学号，姓名）的顺序表，使其具有插入、删除、查找、遍历等功能； （2）对包含重复元素的顺序表，执行删除值为 e 的所有元素； （3）对包含重复元素的无序顺序表，完成去重功能；	
	已完成基本内容（序号）：	1，2，3
选做要求		
	已完成选做内容（序号）：	实现了按学号进行快速排序的功能
数据结构设计	本程序设计了两个结构体，分别是 struct Student 和 struct SqList，分别实现了元素和线性表的功能。 struct Student 有 2 个成员变量，分别是 char no[] 和 char name[]，用于存储学生的学号与姓名。struct SqList 有 3 个成员变量，分别是 ElemType *elem，size_t length 和 size_t listsize，用于存储指向动态元素数组的指针、线性表当前的长度和当前动态数组的长度（最大容量）。	

功能（函数）说明



```
struct Student
{
    char no[10] = {'\0'};
    char name[100] = {'\0'};
    Student() {} // initialize an empty Student object
    Student(const char _no[], const char _name[])…
    ~Student() {}
    Student &operator=(const Student &s)…
    bool operator==(const Student &s) { return !(strcmp(no, s.no) || strcmp(name, s.name)); }
};
ostream &operator<<(ostream &out, Student &s) { return out << s.no << " | " << s.name; }
istream &operator>>(istream &in, Student &s) { return in >> s.no >> s.name; }
```

　　struct Student 的成员函数实现了对应的构造、赋值、值比较等功能；对运算符<<和>>的重载函数分别实现了从 std::cout 和 std::cin 输出、输入元素对应信息。



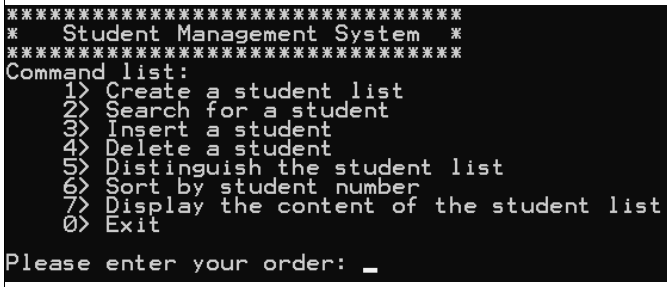
```
struct SqList
{
    ElemType *elem = nullptr;
    size_t length = 0;
    size_t listsize = INIT_LIST_SIZE;
    SqList(const size_t len)…
    ~SqList() { delete[] elem; }
    Status insert()…
    Status searchByNo(char n[])…
    Status searchByName(char n[])…
    Status deleteByNo(char n[])…
    Status deleteByName(char n[])…
    Status distinct()…
    Status traverse()…
    void _swap(ElemType &e1, ElemType &e2)…
    size_t getPivot(const size_t beg, const size_t end)…
    void quickSort(const size_t beg, const size_t end)…
    Status sort()…
    bool check(bool flag = true)…
};
```

　　struct SqList 的成员函数实现了构造、插入、搜索、删除、去重、遍历、排序等操作，函数名很好地描述了各函数的对应功能。

开发环境

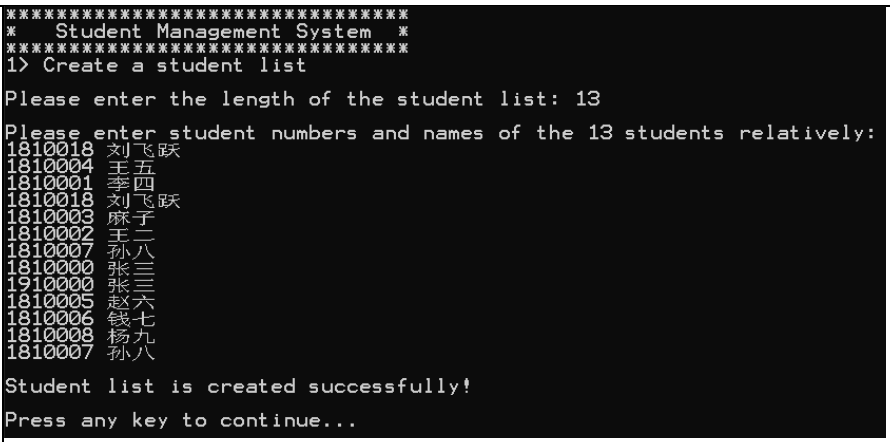
Windows 10, Visual Studio Code with g++, C++ language

调试分析



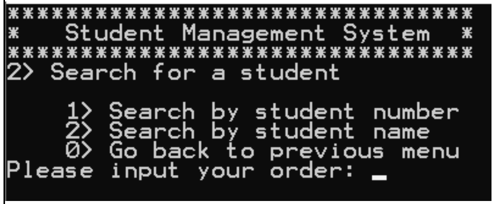
```
**********************************
*    Student Management System   *
**********************************
Command list:
    1> Create a student list
    2> Search for a student
    3> Insert a student
    4> Delete a student
    5> Distinguish the student list
    6> Sort by student number
    7> Display the content of the student list
    0> Exit

Please enter your order: _
```

此为主菜单。



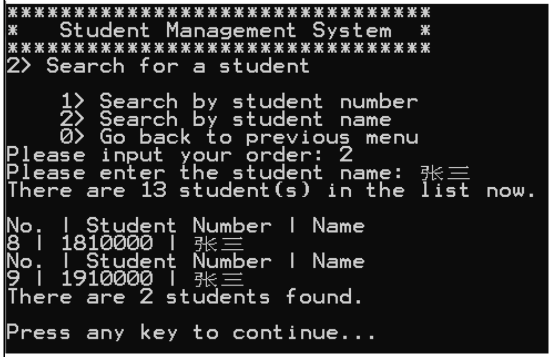
```
*********************************
*   Student Management System   *
*********************************
1> Create a student list

Please enter the length of the student list: 13

Please enter student numbers and names of the 13 students relatively:
1810018 刘飞跃
1810004 王五
1810001 李四
1810018 刘飞跃
1810003 麻子
1810002 王二
1810007 孙八
1810000 张三
1910000 张三
1810005 赵六
1810006 钱七
1810008 杨九
1810007 孙八

Student list is created successfully!

Press any key to continue...
```

本测试数据为乱序数据，含有 2 组完全相同的元素和 1 组同名但不同学号的元素。



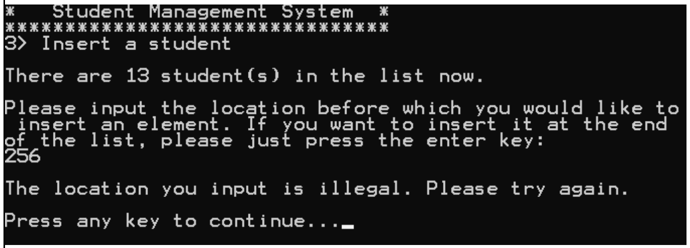
```
*********************************
*   Student Management System   *
*********************************
2> Search for a student

    1> Search by student number
    2> Search by student name
    0> Go back to previous menu
Please input your order: _
```

此为搜索菜单，用户可以选择以学号或名字搜索对应学生。



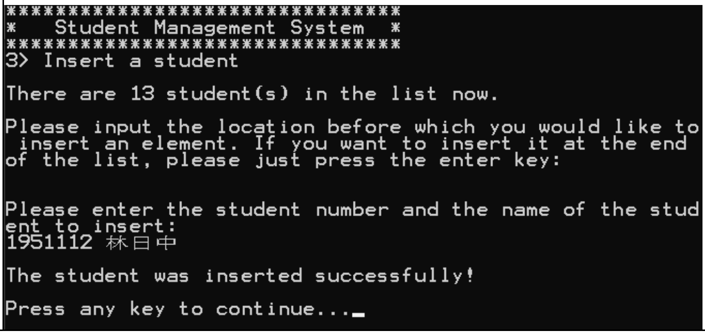
```
*********************************
*   Student Management System   *
*********************************
2> Search for a student

    1> Search by student number
    2> Search by student name
    0> Go back to previous menu
Please input your order: 2
Please enter the student name: 张三
There are 13 student(s) in the list now.

No. | Student Number | Name
8 | 1810000 | 张三
No. | Student Number | Name
9 | 1910000 | 张三
There are 2 students found.

Press any key to continue...
```

以名字搜索可以显示出所有同名学生的信息。



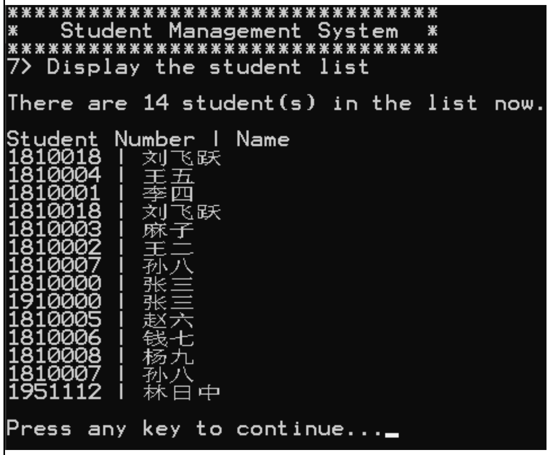
```
*   Student Management System   *
*********************************
3> Insert a student

There are 13 student(s) in the list now.

Please input the location before which you would like to
 insert an element. If you want to insert it at the end
of the list, please just press the enter key:
256

The location you input is illegal. Please try again.

Press any key to continue..._
```

若输入的 `loc` 大于 `length + 1`，则程序输出相关错误信息，并返回上一层菜单。



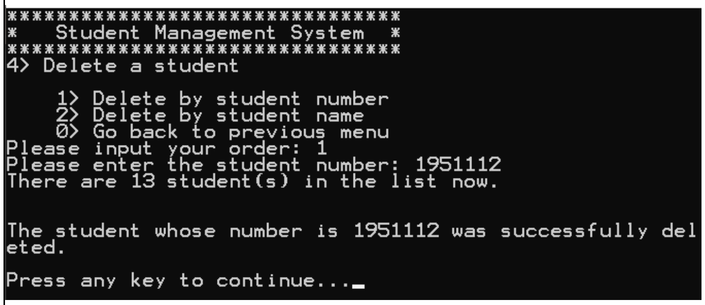
```
*********************************
*   Student Management System   *
*********************************
3> Insert a student

There are 13 student(s) in the list now.

Please input the location before which you would like to
 insert an element. If you want to insert it at the end
of the list, please just press the enter key:

Please enter the student number and the name of the stud
ent to insert:
1951112 林日中

The student was inserted successfully!

Press any key to continue..._
```

将新增元素插到线性表末尾。



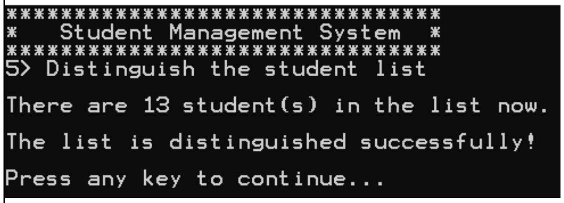
```
***********************************
*    Student Management System    *
***********************************
7> Display the student list

There are 14 student(s) in the list now.

Student Number | Name
1810018 | 刘飞跃
1810004 | 王五
1810001 | 李四
1810018 | 刘飞跃
1810003 | 麻子
1810002 | 王二
1810007 | 孙八
1810000 | 张三
1910000 | 张三
1810005 | 赵六
1810006 | 钱七
1810008 | 杨九
1810007 | 孙八
1951112 | 林日中

Press any key to continue...
```

展示所有学生的信息，可以观察到新元素成功插入线性表末尾。



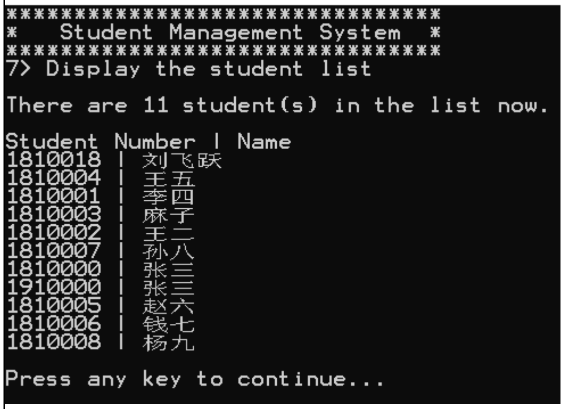
```
***********************************
*    Student Management System    *
***********************************
4> Delete a student

    1> Delete by student number
    2> Delete by student name
    0> Go back to previous menu
Please input your order: 1
Please enter the student number: 1951112
There are 13 student(s) in the list now.


The student whose number is 1951112 was successfully del
eted.

Press any key to continue...
```

删除新增元素。



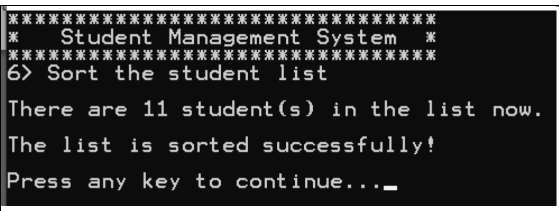
```
***********************************
*    Student Management System    *
***********************************
5> Distinguish the student list

There are 13 student(s) in the list now.

The list is distinguished successfully!

Press any key to continue...
```

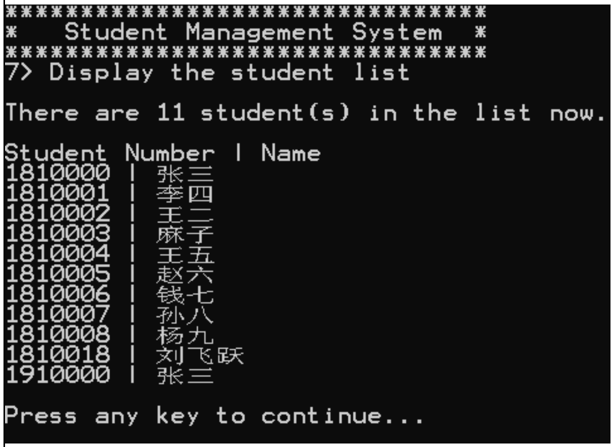
去重。

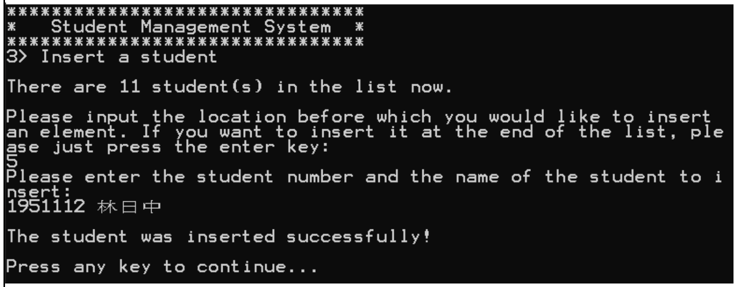


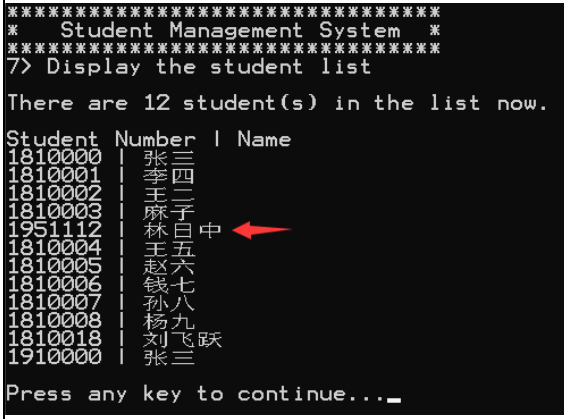
```
***********************************
*    Student Management System    *
***********************************
7> Display the student list

There are 11 student(s) in the list now.

Student Number | Name
1810018 | 刘飞跃
1810004 | 王五
1810001 | 李四
1810003 | 麻子
1810002 | 王二
1810007 | 孙八
1810000 | 张三
1910000 | 张三
1810005 | 赵六
1810006 | 钱七
1810008 | 杨九

Press any key to continue...
```

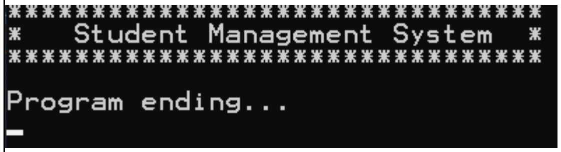
成功去重。观察到学号姓名完全相同的学生被成功去重，而同名学生依然被保留。

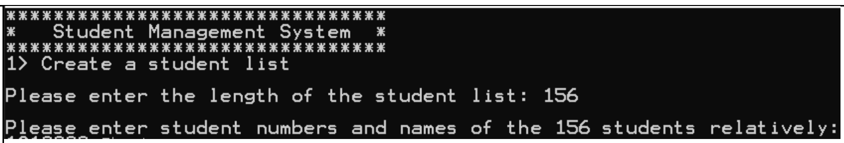

```
**********************************
*    Student Management System   *
**********************************
6> Sort the student list

There are 11 student(s) in the list now.

The list is sorted successfully!

Press any key to continue...
```
进行排序操作。


```
**********************************
*    Student Management System   *
**********************************
7> Display the student list

There are 11 student(s) in the list now.

Student Number | Name
1810000 | 张三
1810001 | 李四
1810002 | 王二
1810003 | 麻子
1810004 | 王五
1810005 | 赵六
1810006 | 钱七
1810007 | 孙八
1810008 | 杨九
1810018 | 刘飞跃
1910000 | 张三

Press any key to continue...
```
可以观察到成功按照学号数值大小从小到大排列。


```
**********************************
*    Student Management System   *
**********************************
3> Insert a student

There are 11 student(s) in the list now.

Please input the location before which you would like to insert
an element. If you want to insert it at the end of the list, ple
ase just press the enter key:
5
Please enter the student number and the name of the student to i
nsert:
1951112 林日中

The student was inserted successfully!

Press any key to continue...
```
在第 5 个元素之前插入新元素。

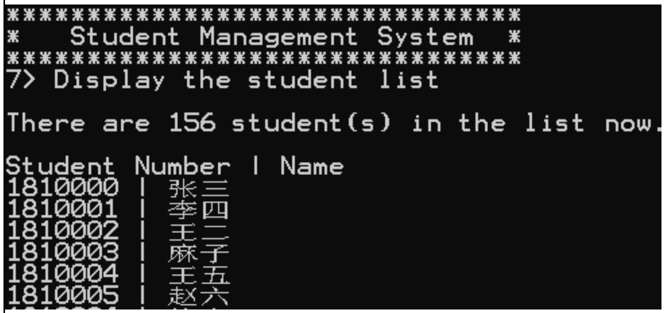

```
**********************************
*    Student Management System   *
**********************************
7> Display the student list

There are 12 student(s) in the list now.

Student Number | Name
1810000 | 张三
1810001 | 李四
1810002 | 王二
1810003 | 麻子
1951112 | 林日中
1810004 | 王五
1810005 | 赵六
1810006 | 钱七
1810007 | 孙八
1810008 | 杨九
1810018 | 刘飞跃
1910000 | 张三

Press any key to continue...
```
观察到新元素成功插入到预期位置（第 5 个）。


```
**********************************
*    Student Management System   *
**********************************

Program ending...
```
成功退出程序，无内存错误掷出。



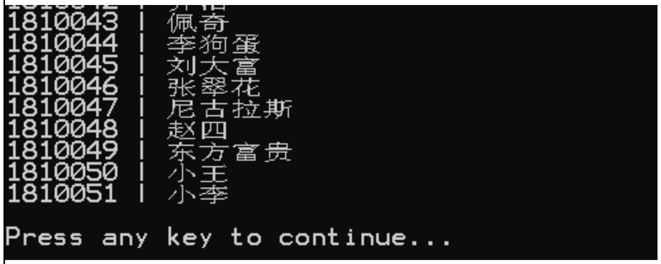
```
**********************************
*   Student Management System   *
**********************************
1> Create a student list

Please enter the length of the student list: 156

Please enter student numbers and names of the 156 students relatively:
```

超 INIT_LIST_SIZE（值为 100）测试，观察到成功插入。



```
**********************************
*   Student Management System   *
**********************************
7> Display the student list

There are 156 student(s) in the list now.

Student Number | Name
1810000 | 张三
1810001 | 李四
1810002 | 王二
1810003 | 麻子
1810004 | 王五
1810005 | 赵六
```

……



```
1810043 | 佩奇
1810044 | 李狗蛋
1810045 | 刘大富
1810046 | 张翠花
1810047 | 尼古拉斯
1810048 | 赵四
1810049 | 东方富贵
1810050 | 小王
1810051 | 小李

Press any key to continue...
```

成功展示出所有元素的信息。

综上，本程序基本完成了题目的功能要求（线性表的基本操作），能够较好地处理非法数据的输入并反馈相关错误提示，程序界面友好，具有比较恰当的人性化提醒，并且具有一定的鲁棒性。

心得体会

一、　实验总结

在本次上机实验中，我复习了理论课上学到的线性表的定义和作用，将课本上线性表的顺序存储结构和与其相关的建立、遍历、查找、插入、删除和去重等函数样例封装成了 struct SqList 结构体，并添加了排序和展示的操作。

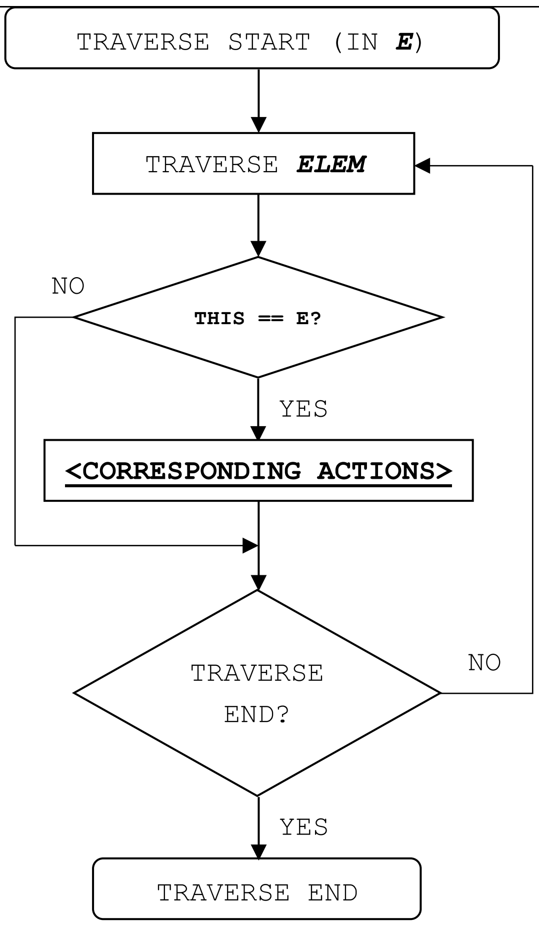
顺序表的优点是，结构简单；存储效率高，是紧凑结构；是一个随机存储结构（直接存取结构）。顺序表的缺点是，在顺序表中进行插入和删除操作时，需要移动数据元素，算法效率较低，在顺序表中元素个数较多时尤为明显；对长度变化较大的线性表，或者要预先分配较大空间或者要经常扩充线性表，给操作带来不方便。这些缺点是数组的静态特性造成的。

二、　性能分析

（1）　顺序表的遍历相关操作（建立、查找、插入、删除等操作）

时间复杂度为 O(n)。

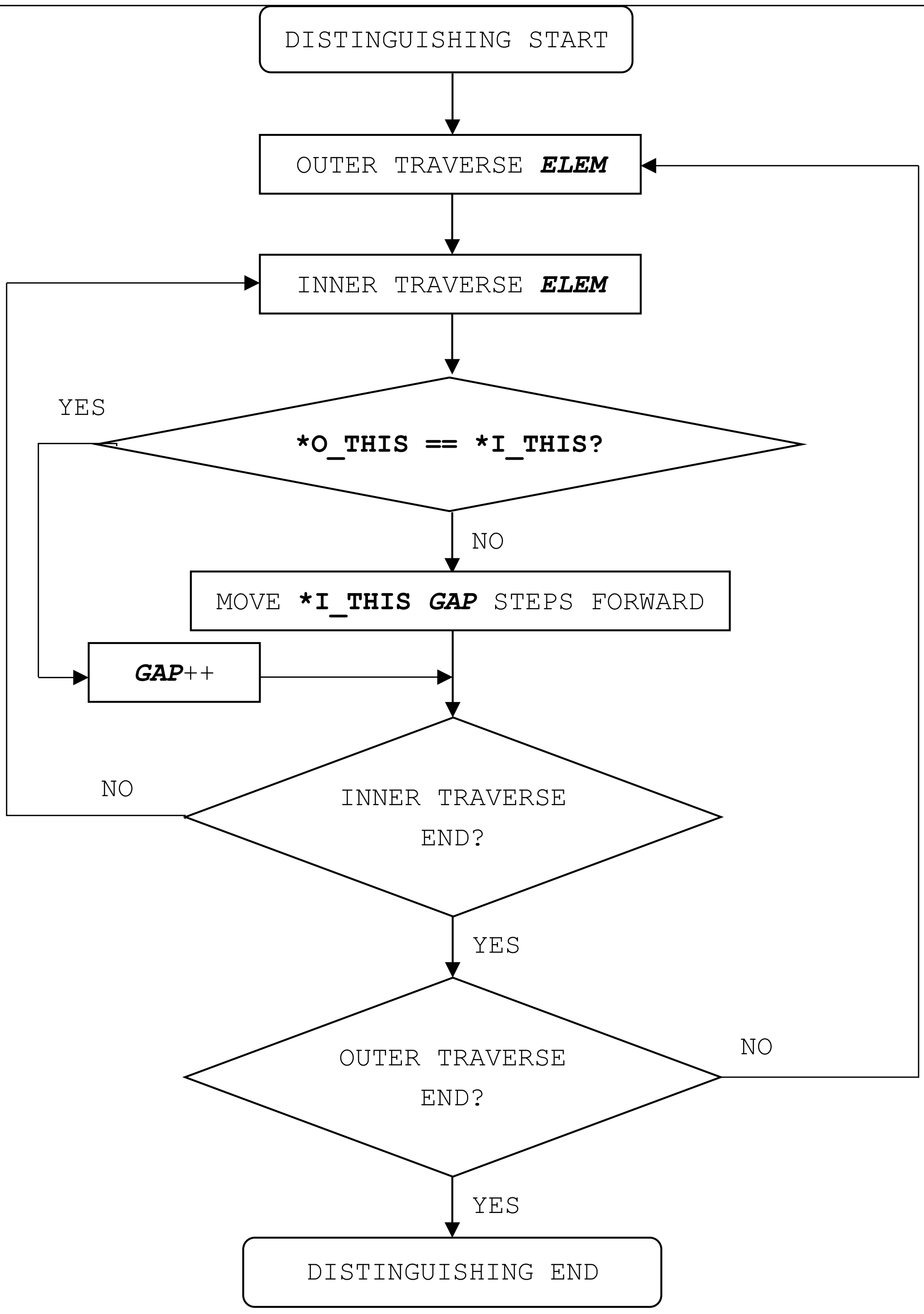
流程图如下：

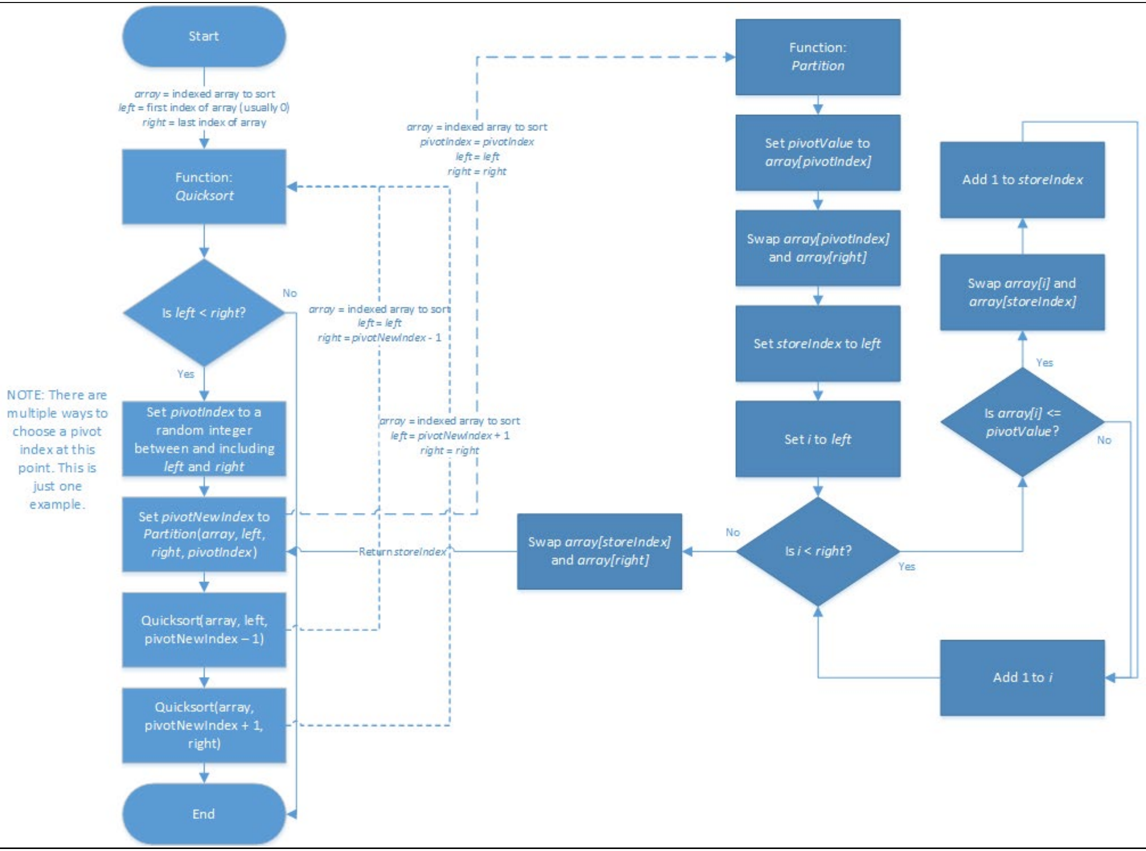


（2） 顺序表的去重

时间复杂度为 $O(n^2)$。

流程图如下：

	 DISTINGUISHING START OUTER TRAVERSE ELEM INNER TRAVERSE ELEM YES *O_THIS == *I_THIS? NO MOVE *I_THIS GAP STEPS FORWARD GAP++ NO INNER TRAVERSE END? YES OUTER TRAVERSE END? NO YES DISTINGUISHING END （3） 顺序表的快速排序 平均时间复杂度为O(NlogN)。 流程图如下：


Start
array = indexed array to sort
left = first index of array (usually 0)
right = last index of array
Function: Quicksort
Is left < right?
No
Yes
NOTE: There are multiple ways to choose a pivot index at this point. This is just one example.
Set pivotIndex to a random integer between and including left and right
Set pivotNewIndex to Partition(array, left, right, pivotIndex)
Quicksort(array, left, pivotNewIndex – 1)
Quicksort(array, pivotNewIndex + 1, right)
End
array = indexed array to sort
pivotIndex = pivotIndex
left = left
right = right
array = indexed array to sort
left = left
right = pivotNewIndex - 1
array = indexed array to sort
left = pivotNewIndex + 1
right = right
Return storeIndex
Function: Partition
Set pivotValue to array[pivotIndex]
Swap array[pivotIndex] and array[right]
Set storeIndex to left
Set i to left
Is i < right?
No
Yes
Swap array[storeIndex] and array[right]
Is array[i] <= pivotValue?
Yes
No
Swap array[i] and array[storeIndex]
Add 1 to storeIndex
Add 1 to i

源码

```
#ifdef __GNUC__
#include <bits/stdc++.h>
#endif

#ifdef _MSC_VER
#define _CRT_SECURE_NO_WARNINGS
#include <iostream>
#include <cstdlib>
#include <cstring>
#include <cmath>
#endif

#include <conio.h>
#include <windows.h>

using namespace std;

#define TRUE 1
#define FALSE 0
#define OK 1
#define NOTFOUND 0
#define INFEASIBLE -1
typedef int Status;
typedef int Boolean;
```

```cpp

struct Student
{
    char no[10] = {'\0'};
    char name[100] = {'\0'};
    Student() {} // initialize an empty Student object
    Student(const char _no[], const char _name[])
    {
        strcpy(no, _no);
        strcpy(name, _name);
    }
    ~Student() {}
    Student &operator=(const Student &s)
    {
        strcpy(no, s.no);
        strcpy(name, s.name);
        return *this;
    }
    bool operator==(const Student &s) { return !(strcmp(no, s.no) || strcmp(name, s.name)); }
};
ostream &operator<<(ostream &out, Student &s) { return out << s.no << " | " << s.name; }
istream &operator>>(istream &in, Student &s) { return in >> s.no >> s.name; }

typedef Student ElemType;
const size_t INIT_LIST_SIZE = 100;
const size_t LIST_SIZE_INCREMENT = 20;

struct SqList
{
    ElemType *elem = nullptr;
    size_t length = 0;
    size_t listsize = INIT_LIST_SIZE;
    SqList(const size_t len)
    {
        while (len > listsize)
        {
            listsize += LIST_SIZE_INCREMENT;
        }

        length = len;
        elem = new ElemType[1 + listsize]();

        for (size_t i = 1; i <= length; i++)
        {
            cin >> elem[i];
        }
    }
    ~SqList()
    {
        if (elem)
        {
            delete[] elem;
        }
    }
    Status insert()
    {
        if (!check())
        {
            return ERROR;
        }
        size_t loc = (size_t)-1;
        char loc_s[20] = {'\0'};
        cout << "Please input the location before which you would like to insert an element. If you want to insert it at the end of the list, please just press the enter key: " << endl;
        if (cin.peek() == ' ' || cin.peek() == '\n')
        {
            cin.get();
        }
        cin.getline(loc_s, 1024);
        if (!(loc = atoi(loc_s)))
        {
            loc = length + 1;
        }
        if (loc > length + 1)
        {
            return ERROR;
        }
        if (length == listsize)
        {
            ElemType *_elem = new ElemType[1 + (listsize += LIST_SIZE_INCREMENT)]();
            copy(elem, elem + length + 1, _elem);
            delete[] elem;
            elem = _elem;
        }

        for (size_t i = length; i >= loc; i--)
        {
            elem[i + 1] = elem[i];
        }
        cout << "Please enter the student number and the name of the student to insert: " << endl;
        cin >> elem[loc];
        length++;

        return OK;
    }
    Status searchByNo(char n[])
    {
        if (!check())
        {
            return ERROR;
        }
        Status found = 0;
        for (size_t i = 1; i <= length; i++)
        {
            if (!strcmp(elem[i].no, n))
            {
                cout << i << " " << elem[i] << endl;
                found++;
            }
        }
        return found;
    }
    Status searchByName(char n[])
    {
        if (!check())
        {
            return ERROR;
        }
        Status found = 0;
        for (size_t i = 1; i <= length; i++)
        {
            if (!strcmp(elem[i].name, n))
            {
                cout << "No. | Student Number | Name" << endl;
                cout << i << " | " << elem[i] << endl;
                found++;
            }
        }
        return found;
    }
    Status deleteByNo(char n[])
    {
        if (!check(false))
        {
            return ERROR;
        }
        int count = 0;
        for (size_t i = 1; i <= length; i++)
        {
            if (!strcmp(elem[i].no, n))
            {
                count++;
            }
            else
            {
                elem[i - count] = elem[i];
            }
        }
        length -= count;
        check();
        return count;
    }
    Status deleteByName(char n[])
    {
        if (!check(false))
        {
            return ERROR;
        }
        int count = 0;
        for (size_t i = 1; i <= length; i++)
        {
            if (!strcmp(elem[i].name, n))
            {
                count++;
            }
            else
            {
                elem[i - count] = elem[i];
            }
        }
        length -= count;
```

```
        check();
        return count;
    }
    Status distinct()
    {
        if (!check())
        {
            return ERROR;
        }
        for (size_t i = 1; i <= length; i++)
        {
            ElemType &only = elem[i];
            size_t count = 0;
            for (size_t j = i + 1; j <= length; j++)
            {
                if (elem[j] == only)
                {
                    count++;
                }
                else
                {
                    elem[j - count] = elem[j];
                }
            }
            length -= count;
        }
        return OK;
    }
    Status traverse()
    {
        if (!check())
        {
            return ERROR;
        }
        cout << "Student Number | Name" << endl;
        for (size_t i = 1; i <= length; i++)
        {
            cout << elem[i] << endl;
        }
        return OK;
    }
    void _swap(ElemType &e1, ElemType &e2)
    {
        std::swap(e1.name, e2.name);
        std::swap(e1.no, e2.no);
    }
    size_t getPivot(const size_t beg, const size_t end)
    {
        size_t i = beg;
        size_t j = beg + 1;
        for (; j != end; j++)
        { // !there's a segmentation fault
            if (atoi(elem[j].no) < atoi(elem[beg].no))
            {
                _swap(elem[++i], elem[j]);
            }
        }
        _swap(elem[beg], elem[i]);

        return i;
    }
    void quickSort(const size_t beg, const size_t end)
    {
        if (beg != end)
        {
            size_t pivot = getPivot(beg, end);
            quickSort(beg, pivot);
            quickSort(pivot + 1, end);
        }
    }
    Status sort()
    {
        if (!check())
        {
            return ERROR;
        }
        quickSort(1, length + 1);
        return OK;
    }
    bool check(bool flag = true)
    {
        if (flag)
        {
            cout << "There are " << length << " student(s) in the list now." << endl
                << endl;
        }
        return (bool)length;
    }
};

inline void header()
{
    system("cls");
    cout << "**********************************" << endl;
    cout << "*   Student Management System   *" << endl;
    cout << "**********************************" << endl;
    return;
}

/*******************************************
 * <_MENU CONTENT_>
 * Command list:
 *     1> Create a student list
 *     2> Insert a student
 *     3> Delete a student
 *     4> Distinguish the student list
 *     5> Sort by student number
 *     6> Display the content of the student list
 *     0> Exit
 * <_END_>
 *****************************************/
inline int menu()
{
    cout << "Command list: " << endl;
    cout << "    1> Create a student list" << endl;
    cout << "    2> Search for a student" << endl;
    cout << "    3> Insert a student" << endl;
    cout << "    4> Delete a student" << endl;
    cout << "    5> Distinguish the student list" << endl;
    cout << "    6> Sort by student number" << endl;
    cout << "    7> Display the content of the student list" << endl;
    cout << "    0> Exit" << endl;

    cout << endl
        << "Please enter your order: ";
    return _getche() - '0';
}

inline void wait_for_press()
{
    cout << endl
```

```cpp
        << "Press any key to continue...";
    (void)_getch();
    return;
}

int main()
{
    SqList *l = nullptr;
    while (true)
    {
        system("cls");
        header();
        int order = menu();
        Status num = 0;
        size_t len = 0;
        char str[100] = {'\0'};
        header();

        switch (order)
        {
        case 0: // exit
            if (l)
            {
                delete l;
            }
            cout << endl
                 << "Program ending..." << endl;
            Sleep(1000);
            exit(0);
            break;
        case 1: // create list
            cout << "1> Create a student list" << endl
                 << endl;
            cout << "Please enter the length of the student list: ";
            cin >> len;
            cout << endl
                 << "Please enter student numbers and names of the " << len << " students relatively: " << endl;
            if (l)
            {
                delete l;
                l = nullptr;
            }
            if ((l = new SqList(len)))
            {
                cout << endl
                    << "Student list is created successfully!" << endl;
            }
            else
            {
                cerr << endl
                    << "Something went wrong. Please try again. " << endl;
            }
            break;
        default:
            if (!l)
            {
                cerr << "The SqList does not exist. Please try again. " << endl;
                break;
            }
            switch (order)
            {
            case 2:
                cout << "2> Search for a student" << endl
                    << endl;
                cout << "    1> Search by student number" << endl
                    << "    2> Search by student name" << endl
                    << "    0> Go back to previous menu" << endl
                    << "Please input your order: ";
                switch (_getche() - '0')
                {
                case 0: // return
                    break;
                case 1: // number
                    cout << endl
                        << "Please enter the student number: ";
                    cin >> str;
                    num = l->searchByNo(str);
                    cout
                        << "There are " << num << " students found." << endl;
                    break;
                case 2: // name
                    cout << endl
                        << "Please enter the student name: ";
                    cin >> str;
                    num = l->searchByName(str);
                    cout
                        << "There are " << num << " students found." << endl;
                    break;
                default: // illegal
                    cerr << "Wrong input. Please try again." << endl;
                    break;
                }

                break;
            case 3: // insert
                cout << "3> Insert a student" << endl
                    << endl;
                if (l->insert())
                {
                    cout << endl
                        << "The student was inserted successfully! " << endl;
                }
                else
                {
                    cerr << endl
                        << "The location you input is illegal. Please try again. " << endl;
                }
                break;
            case 4: // delete
                cout << "4> Delete a student" << endl
                    << endl;
                cout << "    1> Delete by student number" << endl
                    << "    2> Delete by student name" << endl
                    << "    0> Go back to previous menu" << endl
                    << "Please input your order: ";
                switch (_getche() - '0')
                {
                case 0: // return
                    break;
                case 1: // number
                    cout << endl
                        << "Please enter the student number: ";
                    cin >> str;
                    if (l->deleteByNo(str))
                    {
                        cout << endl
                             << "The student whose number is " << str << " was successfully deleted. " << endl;
                    }
                    else
```

```
                    {
                        cerr << endl
                            << "The student whose number is " << str << " was not found. " << endl;
                    }
                    break;
                case 2: // name
                    cout << endl
                         << "Please enter the student name: ";
                    cin >> str;
                    if (l->deleteByName(str))
                    {
                        cout << endl
                            << "The student named " << str << " was successfully deleted. " << endl;
                    }
                    else
                    {
                        cerr << endl
                            << "The student named " << str << " was not found. " << endl;
                    }
                    break;
                default: // illegal
                    cerr << "Wrong input. Please try again." << endl;
                    break;
                }
                break;
            case 5: // distiguish
                cout << "5> Distinguish the student list" << endl
                     << endl;
                l->distinct();
                cout << "The list is distinguished successfully! " << endl;
                break;
            case 6: // sort
                cout << "6> Sort the student list" << endl
                     << endl;
                l->sort();
                cout << "The list is sorted successfully! " << endl;
                break;
            case 7: // display
                cout << "7> Display the student list" << endl
                     << endl;
                if (l)
```

```cpp
                {
                    l->traverse();
                }
                break;
            default: // illegal
                cerr << "Your input is not legal. Please try again. " << endl;
                break;
            }
        }
        wait_for_press();
    }
    return 0;
}
```