C++/项目

```
/* 版本：通讯录8.0
 * 更新时间：2021.1.5
 * 完善：显示的数据分行
 * */

#include <iostream>
#include <windows.h>
#include <cstdlib>
#include <string>
#include <ctime>

# pragma warning (disable:4819)

using namespace std;
//学生的通讯信息内容（学号、姓名、联系电话和学院)--相关知识点（链表）
typedef struct student {
    string stu_num;        //学号
    string stu_name;       //姓名
    string stu_tele;       //电话
    string stu_college;    //学院
    struct student* next;    //链表结构，指向下一个地址
}Student;

//使用全局变量（可避免头插入不成功）
Student* head = NULL;

void mycreat();     //创建通讯录
void myadd();       //添加通讯录
void mydelete();    //删除通讯录
void mydisplay();   //显示通讯录
void mysearch(int pattern, string stu_mes);       //查询通讯录信息
void sch_num(string stu_num);         //按学号查找通讯录
void sch_name(string stu_name);      //按姓名查找通讯录
void mysort();                        //暂时用学号来标准来进行排序
void mymodify(string stu_name);     //修改通讯录
void myprint();                       //显示通讯录主菜单

int main() {
   // system("chcp 936");
   // system("chcp 936");                                    //避免乱码(自己打开的是65001,别人的是
936)
    system("title 华师学生通讯系统    by:F C C");        //设置标题
    system("mode con: cols=32 lines=25");                 //设置窗口大小
```

```
    system("color 79");                                    //设置颜色和大小
    while (1) {
        myprint();
    }
}
//创建通讯录
void mycreat() {
    //★■□○〓⊕▲▼◆●


    head = new Student;             //分配一个通讯录空间给head
    cout << "============================" << endl;
    cout << "● 姓名是：";
    cin >> head->stu_name;
    cout << "● 学号是：";
    cin >> head->stu_num;
    cout << "● 电话是：";
    cin >> head->stu_tele;
    cout << "● 学院是：";
    cin >> head->stu_college;
    cout << "============================" << endl;
    head->next = NULL;              //将下一个节点指向空指针
    return;
}
//添加通讯录
void myadd() {
    if (head == NULL) {              //边界判断
        cout << "当期通讯录为空，无法进行添加！" << endl;
        return;
    }
    //头插法（相对更简单）（链表的换头）
    Student* current = new Student;
    cout << "● 姓名是：";
    cin >> current->stu_name;
    cout << "● 学号是：";
    cin >> current->stu_num;
    cout << "● 电话是：";
    cin >> current->stu_tele;
    cout << "● 学院是：";
    cin >> current->stu_college;
    cout << "============================" << endl;
    current->next = head;
    head = current;
}
//删除通讯录
void mydelete() {
```

```
    if (head == NULL) {              //边界判断
        cout << "〓 当期通讯录为空,无内容可以删除!" << endl;
        return;
    }
    Student* temp, * current;
    current = head;
    while (current != NULL) {
        temp = current->next;
        delete current;
        current = temp;
    }
    head = NULL;
    return;
}
//显示通讯录
void mydisplay() {
    if (head == NULL) {
        cout << "〓 当期通讯录为空" << endl;
        return;
    }
    Student* current = head;
    for (int i = 1; current != NULL; i++) {          //注意这里的结束条件是 current!=NULL
       // cout << "  【姓名—学号—学院—电话】" << endl;
        cout << "★" << endl;
        cout << "⊕名 " << current->stu_name << "\n⊕号 " << current->stu_num << "\n⊕院 " <<
current->stu_college << "\n⊕话 " << current->stu_tele << endl << endl;
        current = current->next;
    }
}
//根据学号查找通讯录
void sch_num(string stu_num) {
    Student* current = head;
    int isfind = 0;      //用来记录是否找到
    while (current != NULL) {
        if (stu_num == current->stu_num) {
            cout << "★" << endl;
            cout << "⊕名 " << current->stu_name << "\n⊕号 " << current->stu_num << "\n⊕院 "
<< current->stu_college << "\n⊕话 " << current->stu_tele << endl << endl;
            isfind = 1;          //已找到
            break;
        }
        current = current->next;
    }
    if (isfind == 0) {
        cout << "〓 未在通讯录中找到该学号! " << endl ;
    }
```

```
    return;
}
//根据姓名查找通讯录
void sch_name(string stu_name) {
    Student* current = head;
    int isfind = 0;     //用来记录是否找到
    while (current != NULL) {
        if (stu_name == current->stu_name) {
            cout << "★" << endl;
            cout << "⊕名 " << current->stu_name << "\n⊕号 " << current->stu_num << "\n⊕院 "
<< current->stu_college << "\n⊕话 " << current->stu_tele << endl << endl;
            isfind = 1;     //已找到
            break;
        }
        current = current->next;
    }
    if (isfind == 0) {
        cout << "〓 未在通讯录中找到该姓名！" << endl;
    }
    return;

}
//查找通讯录
//参数：模式——1代表根据学号查找，2代表根据姓名查找
void mysearch(int pattern, string stu_mes) {
    if (head == NULL) {
        cout << "〓 当前通讯录为空，无法进行查找！" << endl;
        return;
    }
    if (pattern == 1) {
        sch_num(stu_mes);
    }
    else if (pattern == 2) {
        sch_name(stu_mes);
    }
    else {
        cout << "〓 输入错误，请输入正确数字" << endl;
    }
    return;
}
//排序通讯录
void mysort() {
    if (head == NULL) {
        cout << "〓 当前通讯录为空，无法进行排序!" << endl;
        return;
    }
```

```
    Student* current1, * current2;
    //思想：冒泡排序，直接交换他们的值更加方便

    //注意在这里的条件是current->next!=NULL
    for (current1 = head; current1->next != NULL; current1 = current1->next) {
        for (current2 = head; current2->next != NULL; current2 = current2->next) {
            //注意下面的操作
            if (atoi(current2->stu_num.c_str()) > atoi(current2->next->stu_num.c_str())) {
                //交换字符串
                current2->stu_name.swap(current2->next->stu_name);
                current2->stu_tele.swap(current2->next->stu_tele);
                current2->stu_college.swap(current2->next->stu_college);
                current2->stu_num.swap(current2->next->stu_num);
            }
        }
    }
    cout << "★ 排序完毕，请自行查看……";
    return;
}
//修改通讯录
void mymodify(string stu_name) {
    Student* current = head;
    int isfind = 0;      //用来记录是否找到
    while (current != NULL) {
        if (stu_name == current->stu_name) {
            isfind = 1;
            break;
        }
        current = current->next;
    }
    if (isfind == 0) {
        cout << "〓 未在通讯录中找到该姓名！";
        return;
    }
    cout << "*****请输入对应修改的内容*******" << endl;
    cout << "● 学号是：";
    cin >> current->stu_num;
    cout << "● 电话是：";
    cin >> current->stu_tele;
    cout << "● 学院是：";
    cin >> current->stu_college;
    cout << "============================" << endl;
    return;
}
//主菜单
void myprint() {
```

```
    cin.clear();              //清空缓冲区，避免闪屏
    cout << "========学生通讯录系统========\n"
        "       1. 创建通讯录\n"
        "       2. 显示通讯录\n"
        "       3. 查询通讯录\n"
        "       4. 修改通讯录\n"
        "       5. 添加通讯录\n"
        "       6. 删除通讯录\n"
        "       7. 排序通讯录\n"
        "       8. 退出\n\n";
    int choose, num;
    cout << "■ 请选择:";
    cin >> choose;
    system("cls");            //清屏
    cout << "========学生通讯录系统========\n";
    string tmp;
    switch (choose) {
    case 1:
        cout << "● 正在创建通讯录，请输入需要录入的人数：";
        cin >> num;
        mycreat();
        for (int i = 2; i <= num; i++) {
            myadd();
        }
        cout << "★ 创建完毕，正在保存……";
        Sleep(2000);
        break;
    case 2:
        mydisplay();
        system("pause");
        break;
    case 3:
        cout << "● 请输入要查询的模式(1.学号查询 2.姓名查询):";
        int pattern;
        cin >> pattern;
        if (pattern == 1) {
            cout << "● 请输入学号:";
        }
        else if(pattern == 2 ) {
            cout << "● 请输入姓名:";
        }
        else {
            cout << "〓 没有该模式，请重新选择" << endl;
            return;
        }
        cin >> tmp;
```

```
        mysearch(pattern, tmp);
        system("pause");
        break;
    case 4:
        cout << "● 请输入要修改信息的学生姓名:";
        cin >> tmp;
        mymodify(tmp);
        cout << "★ 修改完毕，正在保存……";
        Sleep(2000);
        break;
    case 5:          //添加通讯录
        myadd();
        cout << "★ 添加成功，正在保存……";
        Sleep(2000);
        break;
    case 6:          //删除通讯录
        mydelete();
        cout << "◆ 正在删除……";
        Sleep(2000);
        break;
    case 7:          //排序通讯录
        mysort();
        system("pause");
        break;
    case 8:          //退出
        exit(0);
        break;
    default:
        cout << "〓 输入错误，请重新输入！" << endl;


        system("pause");
        break;
    }
    system("cls");           //退出前清屏
   /// cin.clear();             //清空缓冲区，避免闪屏
    return;
}
```

```
/* 版本：通讯录7.0
 * 时间：2020.12.31
 * */

#include <iostream>
#include <windows.h>
```

```
#include <cstdlib>
#include <string>
#include <ctime>

# pragma warning (disable:4819)

using namespace std;
//学生的通讯信息内容（学号、姓名、联系电话和学院)--相关知识点（链表）
typedef struct student {
    string stu_num;       //学号
    string stu_name;      //姓名
    string stu_tele;      //电话
    string stu_college;   //学院
    struct student* next;   //链表结构，指向下一个地址
}Student;

//使用全局变量（可避免头插入不成功）
Student* head = NULL;

void mycreat();     //创建通讯录
void myadd();       //添加通讯录
void mydelete();    //删除通讯录
void mydisplay();   //显示通讯录
void mysearch(int pattern, string stu_mes);       //查询通讯录信息
void sch_num(string stu_num);        //按学号查找通讯录
void sch_name(string stu_name);      //按姓名查找通讯录
void mysort();                       //暂时用学号来标准来进行排序
void mymodify(string stu_name);     //修改通讯录
void myprint();                      //显示通讯录主菜单

int main() {
    system("chcp 65001");
   // system("chcp 936");                                    //避免乱码(自己打开的是65001,别人的是
936)
    system("title 华师学生通讯系统    by:F C C");        //设置标题
    system("mode con: cols=32 lines=25");                 //设置窗口大小
    system("color 79");                                   //设置颜色和大小
    while (1) {
        myprint();
    }
}
//创建通讯录
void mycreat() {
    //★■□○〓⊕▲▼◆●
```

```
    head = new Student;             //分配一个通讯录空间给head
    cout << "============================" << endl;
    cout << "● 姓名是：";
    cin >> head->stu_name;
    cout << "● 学号是：";
    cin >> head->stu_num;
    cout << "● 电话是：";
    cin >> head->stu_tele;
    cout << "● 学院是：";
    cin >> head->stu_college;
    cout << "============================" << endl;
    head->next = NULL;              //将下一个节点指向空指针
    return;
}
//添加通讯录
void myadd() {
    if (head == NULL) {              //边界判断
        cout << "当期通讯录为空，无法进行添加！" << endl;
        return;
    }
    //头插法（相对更简单）（链表的换头）
    Student* current = new Student;
    cout << "● 姓名是：";
    cin >> current->stu_name;
    cout << "● 学号是：";
    cin >> current->stu_num;
    cout << "● 电话是：";
    cin >> current->stu_tele;
    cout << "● 学院是：";
    cin >> current->stu_college;
    cout << "============================" << endl;
    current->next = head;
    head = current;
}
//删除通讯录
void mydelete() {
    if (head == NULL) {              //边界判断
        cout << "〓 当期通讯录为空,无内容可以删除!" << endl;
        return;
    }
    Student* temp, * current;
    current = head;
    while (current != NULL) {
        temp = current->next;
        delete current;
        current = temp;
```

```
    }
    head = NULL;
    return;
}
//显示通讯录
void mydisplay() {
    if (head == NULL) {
        cout << "〓 当期通讯录为空" << endl;
        return;
    }
    Student* current = head;
    for (int i = 1; current != NULL; i++) {          //注意这里的结束条件是 current!=NULL
        cout << "   【姓名—学号—学院—电话】" << endl;
        cout << i << "." << current->stu_name << " " << current->stu_num << " " << current-
>stu_college << " " << current->stu_tele << endl << endl;
        current = current->next;
    }
}
//根据学号查找通讯录
void sch_num(string stu_num) {
    Student* current = head;
    int isfind = 0;      //用来记录是否找到
    while (current != NULL) {
        if (stu_num == current->stu_num) {
            cout << "   【姓名—学号—学院—电话】" << endl;
            cout << "★ " << current->stu_name << " " << current->stu_num << " " << current-
>stu_college << " " << current->stu_tele << endl << endl;
            isfind = 1;          //已找到
            break;
        }
        current = current->next;
    }
    if (isfind == 0) {
        cout << "〓 未在通讯录中找到该学号！" << endl ;
    }
    return;
}
//根据姓名查找通讯录
void sch_name(string stu_name) {
    Student* current = head;
    int isfind = 0;      //用来记录是否找到
    while (current != NULL) {
        if (stu_name == current->stu_name) {
            cout <<  "★ " << current->stu_name << " " << current->stu_num << " " <<
current->stu_college << " " << current->stu_tele << endl;
            isfind = 1;      //已找到
```

```
            break;
        }
        current = current->next;
    }
    if (isfind == 0) {
        cout << "〓 未在通讯录中找到该姓名！" << endl;
    }
    return;

}
//查找通讯录
//参数：模式——1代表根据学号查找，2代表根据姓名查找
void mysearch(int pattern, string stu_mes) {
    if (head == NULL) {
        cout << "〓 当前通讯录为空，无法进行查找！" << endl;
        return;
    }
    if (pattern == 1) {
        sch_num(stu_mes);
    }
    else if (pattern == 2) {
        sch_name(stu_mes);
    }
    else {
        cout << "〓 输入错误，请输入正确数字" << endl;
    }
    return;
}
//排序通讯录
void mysort() {
    if (head == NULL) {
        cout << "〓 当前通讯录为空，无法进行排序!" << endl;
        return;
    }
    Student* current1, * current2;
    //思想：冒泡排序，直接交换他们的值更加方便

    //注意在这里的条件是current->next!=NULL
    for (current1 = head; current1->next != NULL; current1 = current1->next) {
        for (current2 = head; current2->next != NULL; current2 = current2->next) {
            //注意下面的操作
            if (atoi(current2->stu_num.c_str()) > atoi(current2->next->stu_num.c_str())) {
                //交换字符串
                current2->stu_name.swap(current2->next->stu_name);
                current2->stu_tele.swap(current2->next->stu_tele);
                current2->stu_college.swap(current2->next->stu_college);
```

```
                current2->stu_num.swap(current2->next->stu_num);
            }
        }
    }
    cout << "★ 排序完毕，请自行查看……";
    return;
}
//修改通讯录
void mymodify(string stu_name) {
    Student* current = head;
    int isfind = 0;      //用来记录是否找到
    while (current != NULL) {
        if (stu_name == current->stu_name) {
            isfind = 1;
            break;
        }
        current = current->next;
    }
    if (isfind == 0) {
        cout << "〓 未在通讯录中找到该姓名！";
        return;
    }
    cout << "*****请输入对应修改的内容*******" << endl;
    cout << "● 学号是：";
    cin >> current->stu_num;
    cout << "● 电话是：";
    cin >> current->stu_tele;
    cout << "● 学院是：";
    cin >> current->stu_college;
    cout << "==============================" << endl;
    return;
}
//主菜单
void myprint() {
    cin.clear();             //清空缓冲区，避免闪屏
    cout << "========学生通讯录系统========\n"
        "        1. 创建通讯录\n"
        "        2. 显示通讯录\n"
        "        3. 查询通讯录\n"
        "        4. 修改通讯录\n"
        "        5. 添加通讯录\n"
        "        6. 删除通讯录\n"
        "        7. 排序通讯录\n"
        "        8. 退出\n\n";
    int choose, num;
    cout << "■ 请选择:";
```

```
    cin >> choose;
    system("cls");          //清屏
    cout << "========学生通讯录系统========\n";
    string tmp;
    switch (choose) {
    case 1:
        cout << "● 正在创建通讯录，请输入需要录入的人数：";
        cin >> num;

        mycreat();
        for (int i = 2; i <= num; i++) {
            myadd();
        }
        cout << "★ 创建完毕，正在保存……";
        Sleep(2000);
        break;
    case 2:
        mydisplay();
        system("pause");
        break;
    case 3:
        cout << "● 请输入要查询的模式(1.学号查询 2.姓名查询):";
        int pattern;
        cin >> pattern;
        if (pattern == 1) {
            cout << "● 请输入学号:";
        }
        else {
            cout << "● 请输入姓名:";
        }
        cin >> tmp;
        mysearch(pattern, tmp);
        system("pause");
        break;
    case 4:
        cout << "● 请输入要修改信息的学生姓名:";
        cin >> tmp;
        mymodify(tmp);
        cout << "★ 修改完毕，正在保存……";
        Sleep(2000);
        break;
    case 5:          //添加通讯录
        myadd();
        cout << "★ 添加成功，正在保存……";
        Sleep(2000);
        break;
```

```
    case 6:          //删除通讯录
        mydelete();
        cout << "◆ 正在删除……";
        Sleep(2000);
        break;
    case 7:          //排序通讯录
        mysort();
        system("pause");
        break;
    case 8:          //退出
        exit(0);
        break;
    default:
        cout << "〓 输入错误，请重新输入！" << endl;
        system("pause");
        break;
    }
    system("cls");           //退出前清屏
   /// cin.clear();             //清空缓冲区，避免闪屏
    return;
}
```

```
/* 版本：通讯录6.0
 * 时间：2020.12.19
 * 目前仍可完善：1.交互的文字
 *               2.界面的美化
 * */

#include <iostream>
#include <cstring>
#include <windows.h>
#include <cstdlib>
#include <string.h>
#include <ctime>

using namespace std;
//学生的通讯信息内容（学号、姓名、联系电话和学院)--相关知识点（链表）
typedef struct student {
    string stu_num;        //学号
    string stu_name;       //姓名
    string stu_tele;       //电话
    string stu_college;    //学院
    struct student* next;    //链表结构，指向下一个地址
}Student;
```

```
//使用全局变量（可避免头插入不成功）
Student* head = NULL;

void mycreat();    //创建通讯录
void myadd();      //添加通讯录
void mydelete();   //删除通讯录
void mydisplay();  //显示通讯录
void mysearch(int pattern, string stu_mes);      //查询通讯录信息
void sch_num(string stu_num);       //按学号查找通讯录
void sch_name(string stu_name);     //按姓名查找通讯录
void mysort();                      //暂时用学号来标准来进行排序
void mymodify(string stu_name);    //修改通讯录
void myprint();                     //显示通讯录主菜单

int main() {
    system("title 华师学生通讯系统    by:F C C");       //设置标题
    system("mode con: cols=32 lines=25");               //设置窗口大小
    system("color 87");                                 //设置颜色和大小
    while (1) {
        myprint();
    }
}
//创建通讯录
void mycreat() {
    Student* current;
    head = new Student;            //分配一个通讯录空间给head
    cout << "============================" << endl;
    cout << "姓名是：";
    cin >> head->stu_name;
    cout << "学号是：";
    cin >> head->stu_num;
    cout << "电话是：";
    cin >> head->stu_tele;
    cout << "学院是：";
    cin >> head->stu_college;
    cout << "============================" << endl;
    head->next = NULL;             //将下一个节点指向空指针
    return;
}
//添加通讯录
void myadd() {
    if (head == NULL) {             //边界判断
        cout << "当期通讯录为空，无法进行添加！" << endl;
        return;
    }
```

```
    //头插法（相对更简单）（链表的换头）
    Student* current = new Student;
    cout << "姓名是：";
    cin >> current->stu_name;
    cout << "学号是：";
    cin >> current->stu_num;
    cout << "电话是：";
    cin >> current->stu_tele;
    cout << "学院是：";
    cin >> current->stu_college;
    cout << "============================" << endl;
    current->next = head;
    head = current;
}
//删除通讯录
void mydelete() {
    if (head == NULL) {              //边界判断
        cout << "当期通讯录为空,无内容可以删除1" << endl;
        return;
    }
    Student* temp, * current;
    current = head;
    while (current != NULL) {
        temp = current->next;
        delete current;
        current = temp;
    }
    return;
}
//显示通讯录
void mydisplay() {
    if (head == NULL) {
        cout << "当期通讯录为空" << endl;
        return;
    }
    Student* current = head;
    for (int i = 1; current != NULL; i++) {          //注意这里的结束条件是 current!=NULL
        cout << i << "." << current->stu_name << " " << current->stu_num << " " << current-
>stu_college << " " << current->stu_tele << endl;
        current = current->next;
    }
}
//根据学号查找通讯录
void sch_num(string stu_num) {
    Student* current = head;
    int isfind = 0;      //用来记录是否找到
```

```
    while (current != NULL) {
        if (stu_num == current->stu_num) {
            cout << current->stu_name << " " << current->stu_num << " " << current-
>stu_college << " " << current->stu_tele << endl;
            isfind = 1;          //已找到
            break;
        }
        current = current->next;
    }
    if (isfind == 0) {
        cout << "未在通讯录中找到该学号！" << endl;
    }
    return;
}
//根据姓名查找通讯录
void sch_name(string stu_name) {
    Student* current = head;
    int isfind = 0;      //用来记录是否找到
    while (current != NULL) {
        if (stu_name == current->stu_name) {
            cout << current->stu_name << " " << current->stu_num << " " << current-
>stu_college << " " << current->stu_tele << endl;
            isfind = 1;      //已找到
            break;
        }
        current = current->next;
    }
    if (isfind == 0) {
        cout << "未在通讯录中找到该姓名！" << endl;
    }
    return;

}
//查找通讯录
//参数：模式——1代表根据学号查找，2代表根据姓名查找
void mysearch(int pattern, string stu_mes) {
    if (head == NULL) {
        cout << "当前通讯录为空，无法进行查找！" << endl;
        return;
    }
    if (pattern == 1) {
        sch_num(stu_mes);
    }
    else if (pattern == 2) {
        sch_name(stu_mes);
    }
```

```
    else {
        cout << "输入错误，请输入正确数字" << endl;
    }
    return;
}
//排序通讯录
void mysort() {
    if (head == NULL) {
        cout << "当前通讯录为空，无法进行排序!" << endl;
        return;
    }
    Student* current1, * current2;
    //思想：冒泡排序，直接交换他们的值更加方便

    //注意在这里的条件是current->next!=NULL
    for (current1 = head; current1->next != NULL; current1 = current1->next) {
        for (current2 = head; current2->next != NULL; current2 = current2->next) {
            //注意下面的操作
            if (atoi(current2->stu_num.c_str()) > atoi(current2->next->stu_num.c_str())) {
                //交换字符串
                current2->stu_name.swap(current2->next->stu_name);
                current2->stu_tele.swap(current2->next->stu_tele);
                current2->stu_college.swap(current2->next->stu_college);
                current2->stu_num.swap(current2->next->stu_num);
            }
        }
    }
    return;
}
//修改通讯录
void mymodify(string stu_name) {
    Student* current = head;
    int isfind = 0;      //用来记录是否找到
    while (current != NULL) {
        if (stu_name == current->stu_name) {
            isfind = 1;
            break;
        }
        current = current->next;
    }
    if (isfind == 0) {
        cout << "未在通讯录中找到该姓名！";
        return;
    }
    cout << "*****请输入对应修改的内容*******" << endl;
    cout << "学号是：";
```

```
    cin >> current->stu_num;
    cout << "电话是：";
    cin >> current->stu_tele;
    cout << "学院是：";
    cin >> current->stu_college;
    cout << "============================" << endl;
    return;
}
//主菜单
void myprint() {
    cin.clear();             //清空缓冲区，避免闪屏
    cout << "========学生通讯录系统========\n"
        "       1. 创建通讯录\n"
        "       2. 显示通讯录\n"
        "       3. 查询通讯录\n"
        "       4. 修改通讯录\n"
        "       5. 添加通讯录\n"
        "       6. 删除通讯录\n"
        "       7. 排序通讯录\n"
        "       8. 退出\n";
    int choose, num, t;
    cin >> choose;
    system("cls");           //清屏
    cout << "========学生通讯录系统========\n";
    string tmp;

    switch (choose) {
    case 1:
        cout << "正在创建通讯录，请输入需要录入的人数：";
        cin >> num;

        mycreat();
        for (int i = 2; i <= num; i++) {
            myadd();
        }
        cout << "创建完毕，正在保存……";
        Sleep(2000);
        break;
    case 2:
        mydisplay();
        system("pause");
        break;
    case 3:
        cout << "请输入要查询的模式(1.学号查询 2.姓名查询):";
        int pattern;
        cin >> pattern;
```

```
        if (pattern == 1) {
            cout << "请输入学号:";
        }
        else {
            cout << "请输入姓名:";
        }
        cin >> tmp;
        mysearch(pattern, tmp);
        system("pause");
        break;
    case 4:
        cout << "请输入要修改信息的学生姓名:";
        cin >> tmp;
        mymodify(tmp);
        cout << "修改完毕，正在保存……";
        Sleep(2000);
        break;
    case 5:          //添加通讯录
        myadd();
        cout << "添加成功，正在保存……";
        Sleep(2000);
        break;

    case 6:          //删除通讯录
        mydelete();
        cout << "正在删除……";
        Sleep(2000);
        break;
    case 7:          //排序通讯录
        mysort();
        cout << "排序完毕，请自行查看……";
        system("pause");
        break;
    case 8:          //退出
        exit(0);
        break;
        //这里有错，输入111111会报错，不断闪烁
    default:
        cout << "输入错误，请重新输入！" << endl;
        system("pause");
        break;
    }
    system("cls");           //退出前清屏
    return;
}
```

```
/* 版本：通讯录5.0
 * 时间：12.15
 * 完善：1.基本实现通讯录的功能
 *       2.美化界面
 *       3.逻辑语法完善
 * */

#include <iostream>
#include <cstring>
#include <windows.h>
#include <cstdlib>
#include <string.h>
#include <ctime>

using namespace std;
//学生的通讯信息内容（学号、姓名、联系电话和学院）
typedef struct student{
    string stu_num;        //学号
    string stu_name;       //姓名
    string stu_tele;       //电话
    string stu_college;    //学院
    struct student* next;    //链表结构，指向下一个地址
}Student;


//使用全局变量可避免头插入不成功
Student *head = NULL;

void mycreat();     //创建通讯录
void myadd();       //添加通讯录
void mydelete();    //删除通讯录
void mydisplay();   //显示通讯录
void mysearch(int pattern,string stu_mes);       //查询通讯录信息
void sch_num(string stu_num);        //按学号查找通讯录
void sch_name(string stu_name);      //按姓名查找通讯录
void mysort();                       //暂时用学号来标准来进行排序
void mymodify(string stu_name);     //修改通讯录
void myprint();                      //显示通讯录主菜单

int main() {
    system("title 华师学生通讯系统     by:F C C");         //设置标题
    system("mode con: cols=32 lines=25");                 //设置窗口大小
    system("color 87");                                   //设置颜色和大小
```

```
    while(1){
        myprint();
    }

}
void mycreat(){
    Student *current;
    head = new Student;
    cout<<"============================="<<endl;
    cout<<"姓名是: ";
    cin>>head->stu_name;
    cout<<"学号是: ";
    cin>>head->stu_num;
    cout<<"电话是: ";
    cin>>head->stu_tele;
    cout<<"学院是: ";
    cin>>head->stu_college;
    cout<<"============================="<<endl;
    head->next=NULL;         //将下一个节点指向空指针
    return;
}
void myadd(){
   if( head == NULL ){
        cout<<"当期通讯录为空，无法进行添加！"<<endl;
        return;
    }
    //头插法（相对更简单）
    Student *current = new Student;
    cout<<"姓名是: ";
    cin>>current->stu_name;
    cout<<"学号是: ";
    cin>>current->stu_num;
    cout<<"电话是: ";
    cin>>current->stu_tele;
    cout<<"学院是: ";
    cin>>current->stu_college;
    cout<<"============================="<<endl;
    current->next = head;
    head = current;
}
void mydelete(){
    if( head == NULL ){
        cout<<"当期通讯录为空,无内容可以删除1"<<endl;
        return;
    }
    Student* temp,*current;
```

```
    current=head;
    while ( current != NULL ){
        temp = current->next;
        delete current;
        current = temp;
    }
    return;
}
void mydisplay(){
    if( head == NULL ){
        cout<<"当期通讯录为空"<<endl;
        return;
    }
    Student* current = head;
    for(int i=1;current!=NULL;i++){          //注意这里的结束条件是 current!=NULL
        cout<<i<<"."<<current->stu_name<<" "<<current->stu_num<<" "<<current->stu_college<<"
"<<current->stu_tele<<endl;
        current=current->next;
    }
}
void sch_num(string stu_num){
    Student* current = head;
    int isfind = 0;      //用来记录是否找到
    while(current!=NULL) {
        if ( stu_num == current->stu_num ){
            cout << current->stu_name << " " << current->stu_num << " " << current-
>stu_college << " " << current->stu_tele<< endl;
            isfind = 1;
            break;
        }
        current = current->next;
    }
    if( isfind == 0 ){
        cout<<"未在通讯录中找到该学号！"<<endl;
    }
    return;
}
void sch_name(string stu_name){
    Student* current = head;
    int isfind = 0;      //用来记录是否找到
    while(current!=NULL) {
        if ( stu_name == current->stu_name ){
            cout << current->stu_name << " " << current->stu_num << " " << current-
>stu_college << " " << current->stu_tele<< endl;
            isfind = 1;
            break;
```

```
        }
        current = current->next;
    }
    if( isfind == 0 ){
        cout<<"未在通讯录中找到该姓名！"<<endl;
    }
    return;

}
//参数：模式——1代表根据学号查找，2代表根据姓名查找
void mysearch(int pattern,string stu_mes){
    if( head == NULL ){
        cout<<"当前通讯录为空，无法进行查找！"<<endl;
        return;
    }
    if( pattern == 1 ){
        sch_num(stu_mes);
    }else if( pattern == 2 ){
        sch_name(stu_mes);
    }else{
        cout<<"输入错误，请输入正确数字"<<endl;
    }
    return;
}
void mysort(){
    if( head == NULL ){
        cout<<"当前通讯录为空，无法进行排序！"<<endl;
        return;
    }
    Student *current1,*current2;
    //思想：冒泡排序，直接交换他们的值更加方便

    //注意在这里的条件是current->next!=NULL
    for (current1=head;current1->next!=NULL;current1=current1->next){
        for (current2=head;current2->next!=NULL;current2=current2->next) {
            if( atoi(current2->stu_num.c_str()) > atoi(current2->next->stu_num.c_str()) ){
                //交换字符串
                current2->stu_name.swap(current2->next->stu_name);
                current2->stu_tele.swap(current2->next->stu_tele);
                current2->stu_college.swap(current2->next->stu_college);
                current2->stu_num.swap(current2->next->stu_num);
            }
        }

    }
    return;
```

```
}
void mymodify(string stu_name){
    Student* current = head;
    int isfind = 0;     //用来记录是否找到
    while(current!=NULL) {
        if ( stu_name == current->stu_name ){
            isfind = 1;
            break;
        }
        current = current->next;
    }
    if( isfind == 0 ){
        cout<<"未在通讯录中找到该姓名！";
        return;
    }
    cout<<"*****请输入对应修改的内容*******"<<endl;
    cout<<"学号是：";
    cin>>current->stu_num;
    cout<<"电话是：";
    cin>>current->stu_tele;
    cout<<"学院是：";
    cin>>current->stu_college;
    cout<<"=============================="<<endl;
    return;
}
void myprint(){
    cin.clear();             //清空缓冲区，避免闪屏
    cout<<"========学生通讯录系统========\n"
          "       1. 创建通讯录\n"
          "       2. 显示通讯录\n"
          "       3. 查询通讯录\n"
          "       4. 修改通讯录\n"
          "       5. 添加通讯录\n"
          "       6. 删除通讯录\n"
          "       7. 排序通讯录\n"
          "       8. 退出\n";
    int choose,num,t;
    cin>>choose;
    system("cls");
    cout<<"========学生通讯录系统========\n";
    string tmp;

    switch(choose){
        case 1:
            cout<<"正在创建通讯录，请输入需要录入的人数：";
            cin>>num;
```

```
            mycreat();
            for(int i=2;i<=num;i++){
                myadd();
            }
            cout<<"创建完毕，正在保存……";
            Sleep(2000);
            break;
        case 2:
            mydisplay();
            system("pause");
            break;
        case 3:
            cout<<"请输入要查询的模式(1.学号查询 2.姓名查询):";
            int pattern;
            cin>>pattern;
            if( pattern == 1 ){
                cout<<"请输入学号:";
            }else{
                cout<<"请输入姓名:";
            }
            cin>>tmp;
            mysearch(pattern,tmp);
            system("pause");
            break;
        case 4:
            cout<<"请输入要修改信息的学生姓名:";
            cin>>tmp;
            mymodify(tmp);
            cout<<"修改完毕，正在保存……";
            Sleep(2000);
            break;
        case 5:          //添加通讯录
            myadd();
            cout<<"添加成功，正在保存……";
            Sleep(2000);
            break;

        case 6:          //删除通讯录
            mydelete();
            cout<<"正在删除……";
            Sleep(2000);
            break;
        case 7:          //排序通讯录
            mysort();
            cout<<"排序完毕，请自行查看……";
```

```
            system("pause");
            break;
        case 8:          //退出
            exit(0);
            break;
            //这里有错，输入111111会报错，不断闪烁
        default:
            cout<<"输入错误，请重新输入！"<<endl;
            system("pause");
            break;
    }
    system("cls");           //退出前清屏
    return;
}
```

```
/* 版本：通讯录4.0
 * 时间：2020.12.15
 * 完善：1.基本实现通讯录的功能
 * */

#include <iostream>
#include <cstring>
#include <windows.h>
#include <cstdlib>
#include <string.h>
#include <ctime>

using namespace std;
//学生的通讯信息内容（学号、姓名、联系电话和学院）
typedef struct student{
    string stu_num;        //学号
    string stu_name;       //姓名
    string stu_tele;       //电话
    string stu_college;    //学院
    struct student* next;    //链表结构，指向下一个地址
}Student;


//使用全局变量可避免头插入不成功
Student *head = NULL;

void mycreat();     //创建通讯录
void myadd();       //添加通讯录
void mydelete();    //删除通讯录
```

```
void mydisplay();  //显示通讯录
void mysearch(int pattern,string stu_mes);       //查询通讯录信息
void sch_num(string stu_num);      //按学号查找通讯录
void sch_name(string stu_name);      //按姓名查找通讯录
void mysort();      //暂时用学号来标准来进行排序
void mymodify(string stu_name);     //修改通讯录
void myprint();      //显示通讯录主菜单

int main() {
    /*
    mycreat();
    myadd();
    myadd();
    mydisplay();
 //  mymodify("czh");
 //  sch_num("2020");
 //  sch_name("czh");
    mysort();
    mydisplay();
    mydelete();*/

    system("title 华师学生通讯系统     by:F C C");         //设置标题
    system("mode con: cols=32 lines=25");
    while(1){
        myprint();
     //   mydisplay();
    }

}
void mycreat(){
    Student *current;
    head = new Student;
    cout<<"============================="<<endl;
    cout<<"姓名是: ";
    cin>>head->stu_name;
    cout<<"学号是: ";
    cin>>head->stu_num;
    cout<<"电话是: ";
    cin>>head->stu_tele;
    cout<<"学院是: ";
    cin>>head->stu_college;
    cout<<"============================="<<endl;
    head->next=NULL;         //将下一个节点指向空指针
    return;
}
void myadd(){
```

```
    //头插法（相对更简单）
    Student *current = new Student;
    cout<<"姓名是：";
    cin>>current->stu_name;
    cout<<"学号是：";
    cin>>current->stu_num;
    cout<<"电话是：";
    cin>>current->stu_tele;
    cout<<"学院是：";
    cin>>current->stu_college;
    cout<<"============================="<<endl;
    current->next = head;
    head = current;
}
void mydelete(){
    Student* temp,*current;
    current=head;
    while ( current != NULL ){
        temp = current->next;
        delete current;
        current = temp;
    }
    return;
}
void mydisplay(){
   // cout<<"========学生通讯录系统========\n";
    if( head == NULL ){
        cout<<"当期通讯录为空";
        return;
    }
    Student* current = head;
    for(int i=1;current!=NULL;i++){          //注意这里的结束条件是 current!=NULL
        cout<<i<<"."<<current->stu_name<<" "<<current->stu_num<<" "<<current->stu_college<<"
"<<current->stu_tele<<endl;
        current=current->next;
    }
}
void sch_num(string stu_num){
    Student* current = head;
    int isfind = 0;     //用来记录是否找到
    while(current!=NULL) {
        if ( stu_num == current->stu_num ){
            cout << current->stu_name << " " << current->stu_num << " " << current-
>stu_college << " " << current->stu_tele<< endl;
            isfind = 1;
            break;
```

```
        }
        current = current->next;
    }
    if( isfind == 0 ){
        cout<<"未在通讯录中找到该学号！";
    }
    return;
}
void sch_name(string stu_name){
    Student* current = head;
    int isfind = 0;     //用来记录是否找到
    while(current!=NULL) {
        if ( stu_name == current->stu_name ){
            cout << current->stu_name << " " << current->stu_num << " " << current-
>stu_college << " " << current->stu_tele<< endl;
            isfind = 1;
            break;
        }
        current = current->next;
    }
    if( isfind == 0 ){
        cout<<"未在通讯录中找到该姓名！";
    }
    return;

}
//参数：模式——1代表根据学号查找，2代表根据姓名查找
void mysearch(int pattern,string stu_mes){
    if( head == NULL ){
        cout<<"当前通讯录为空，无法进行查找！\n";
        return;
    }



    if( pattern == 1 ){
        sch_num(stu_mes);
    }else if( pattern == 2 ){
        sch_name(stu_mes);
    }else{
        cout<<"数字输入错误，请输入正确数字"<<endl;
    }
    return;
}
void mysort(){
    if( head == NULL ){
```

```
        cout<<"当前通讯录为空，无法进行排序！\n";
        return;
    }

    Student *current1,*current2;
    //思想：冒泡排序，直接交换他们的值更加方便

    //注意在这里的条件是current->next!=NULL
    for (current1=head;current1->next!=NULL;current1=current1->next){
        for (current2=head;current2->next!=NULL;current2=current2->next) {
            if( atoi(current2->stu_num.c_str()) > atoi(current2->next->stu_num.c_str()) ){
                current2->stu_name.swap(current2->next->stu_name);
                current2->stu_tele.swap(current2->next->stu_tele);
                current2->stu_college.swap(current2->next->stu_college);
                current2->stu_num.swap(current2->next->stu_num);
            }
        }

    }
    return;

}
void mymodify(string stu_name){
    Student* current = head;
    int isfind = 0;     //用来记录是否找到
    while(current!=NULL) {
        if ( stu_name == current->stu_name ){
            // cout << current->stu_name << " " << current->stu_num << " " << current-
>stu_college << " " << current->stu_tele<< endl;
            isfind = 1;
            break;
        }
        current = current->next;
    }
    if( isfind == 0 ){
        cout<<"未在通讯录中找到该姓名！";
        return;
    }
    cout<<"*****请输入对应修改的内容*******"<<endl;
    cout<<"学号是：";
    cin>>current->stu_num;
    cout<<"电话是：";
    cin>>current->stu_tele;
    cout<<"学院是：";
    cin>>current->stu_college;
    cout<<"============================="<<endl;
```

```
    return;
}
void myprint(){
    cout<<"========学生通讯录系统========\n"
          "       1. 创建通讯录\n"
          "       2. 显示通讯录\n"
          "       3. 查询通讯录\n"
          "       4. 修改通讯录\n"
          "       5. 添加通讯录\n"
          "       6. 删除通讯录\n"
          "       7. 排序通讯录\n"
          "       8. 退出\n";
    int choose,num,t;
    cin>>choose;
    system("cls");
    cout<<"========学生通讯录系统========\n";
    string tmp;

    switch(choose){
        case 1:
            cout<<"正在创建通讯录，请输入需要录入的人数：";
            cin>>num;

            mycreat();
            for(int i=2;i<=num;i++){
                myadd();
            }
            cout<<"创建完毕，正在保存……";
            Sleep(2000);
            break;
        case 2:
            mydisplay();
            system("pause");
            break;
        case 3:
            cout<<"请输入要查询的模式(1.学号查询 2.姓名查询):";
            int pattern;
            cin>>pattern;
            if( pattern == 1 ){
                cout<<"请输入学号:";
            }else{
                cout<<"请输入姓名:";
            }
```

```
            cin>>tmp;
            mysearch(pattern,tmp);
            system("pause");
            break;
        case 4:
            cout<<"请输入要修改信息的学生姓名:";
            cin>>tmp;
            mymodify(tmp);
            cout<<"修改完毕，正在保存……";
            Sleep(2000);
            break;
        case 5:          //添加通讯录
            myadd();
            cout<<"添加成功，正在保存……";
            Sleep(2000);
            break;

        case 6:          //删除通讯录
            mydelete();
            cout<<"正在删除……";
            Sleep(2000);
            break;
        case 7:          //排序通讯录
            mysort();
            cout<<"排序完毕，请自行保存……";
            Sleep(2000);
            break;
        case 8:          //退出
            exit(0);
            break;
        default:
            cout<<"输入错误，请重新输入！"<<endl;
            break;
    }

    system("cls");          //退出前清一下屏
    return;
}
```

```
/* 版本：通讯录3.0
 * 时间：2020.12.11
 * 功能：实现创建，添加，删除，显示通讯录
 * 完善：1.完善了查询，排序，修改功能
 *       2.将char换成string方便交换
```

```
 *       3.显示函数
 * */

#include <iostream>
#include <cstring>
#include <cstdlib>
#include <cstdlib>
using namespace std;
//学生的通讯信息内容（学号、姓名、联系电话和学院)
typedef struct student{
    string stu_num;        //学号
    string stu_name;       //姓名
    string stu_tele;       //电话
    string stu_college;    //学院
    struct student* next;    //链表结构，指向下一个地址
}Student;


//使用全局变量可避免头插入不成功
Student *head = NULL;

void mycreat();     //创建通讯录
void myadd();       //添加通讯录
void mydelete();    //删除通讯录
void mydisplay();   //显示通讯录
void mysearch(int pattern);       //查询通讯录信息
void sch_num(string stu_num);      //按学号查找通讯录
void sch_name(string stu_name);      //按姓名查找通讯录
void mysort();      //暂时用学号来标准来进行排序
void mymodify(string stu_name);     //修改通讯录
void myprint();      //显示通讯录主菜单

int main() {
    /*
    mycreat();
    myadd();
    myadd();
    mydisplay();
 //  mymodify("czh");
 //  sch_num("2020");
 //  sch_name("czh");
    mysort();
    mydisplay();
    mydelete();*/

    myprint();
```

```
}
void mycreat(){
    Student *current;
    head = new Student;
    cout<<"姓名是: ";
    cin>>head->stu_name;
    cout<<"学号是: ";
    cin>>head->stu_num;
    cout<<"电话是: ";
    cin>>head->stu_tele;
    cout<<"学院是: ";
    cin>>head->stu_college;
    cout<<"============================="<<endl;
    head->next=NULL;         //将下一个节点指向空指针
    return;
}
void myadd(){
    //头插法（相对更简单）
    Student *current = new Student;
    cout<<"姓名是: ";
    cin>>current->stu_name;
    cout<<"学号是: ";
    cin>>current->stu_num;
    cout<<"电话是: ";
    cin>>current->stu_tele;
    cout<<"学院是: ";
    cin>>current->stu_college;
    cout<<"============================="<<endl;
    current->next = head;
    head = current;
}
void mydelete(){
    Student* temp,*current;
    current=head;
    while ( current != NULL ){
        temp = current->next;
        delete current;
        current = temp;
    }
    return;
}
void mydisplay(){
    if( head == NULL ){
        cout<<"当期通讯录为空";
        return;
    }
```

```
    Student* current = head;
    for(int i=1;current!=NULL;i++){          //注意这里的结束条件是 current!=NULL
        cout<<i<<"."<<current->stu_name<<" "<<current->stu_num<<" "<<current->stu_college<<"
"<<current->stu_tele<<endl;
        current=current->next;
    }
}
void sch_num(string stu_num){
    Student* current = head;
    int isfind = 0;      //用来记录是否找到
    while(current!=NULL) {
        if ( stu_num == current->stu_num ){
            cout << current->stu_name << " " << current->stu_num << " " << current-
>stu_college << " " << current->stu_tele<< endl;
            isfind = 1;
            break;
        }
        current = current->next;
    }
    if( isfind == 0 ){
        cout<<"未在通讯录中找到该学号！";
    }
    return;
}
void sch_name(string stu_name){
    Student* current = head;
    int isfind = 0;      //用来记录是否找到
    while(current!=NULL) {
        if ( stu_name == current->stu_name ){
            cout << current->stu_name << " " << current->stu_num << " " << current-
>stu_college << " " << current->stu_tele<< endl;
            isfind = 1;
            break;
        }
        current = current->next;
    }
    if( isfind == 0 ){
        cout<<"未在通讯录中找到该姓名！";
    }
    return;

}
//参数：模式——1代表根据学号查找，2代表根据姓名查找
void mysearch(int pattern,string stu_mes){
    if( pattern == 1 ){
        sch_num(stu_mes);
```

```
    }else if( pattern == 2 ){
        sch_name(stu_mes);
    }
    return;
}
void mysort(){
    Student *current1,*current2;
    //思想：冒泡排序，直接交换他们的值更加方便

    //注意在这里的条件是current->next!=NULL
    for (current1=head;current1->next!=NULL;current1=current1->next){
        for (current2=head;current2->next!=NULL;current2=current2->next) {
            if( atoi(current2->stu_num.c_str()) > atoi(current2->next->stu_num.c_str()) ){
                current2->stu_name.swap(current2->next->stu_name);
                current2->stu_tele.swap(current2->next->stu_tele);
                current2->stu_college.swap(current2->next->stu_college);
                current2->stu_num.swap(current2->next->stu_num);
            }
        }

    }
    return;

}
void mymodify(string stu_name){
    Student* current = head;
    int isfind = 0;     //用来记录是否找到
    while(current!=NULL) {
        if ( stu_name == current->stu_name ){
            // cout << current->stu_name << " " << current->stu_num << " " << current-
>stu_college << " " << current->stu_tele<< endl;
            isfind = 1;
            break;
        }
        current = current->next;
    }
    if( isfind == 0 ){
        cout<<"未在通讯录中找到该姓名！";
        return;
    }
    cout<<"*****请输入对应修改的内容*******"<<endl;
    cout<<"学号是：";
    cin>>current->stu_num;
    cout<<"电话是：";
    cin>>current->stu_tele;
    cout<<"学院是：";
```

```
    cin>>current->stu_college;
    cout<<"============================="<<endl;


    return;
}
void myprint(){
    cout<<"--------学生通讯录系统--------\n"
          "       1. 创建通讯录\n"
          "       2. 显示通讯录\n"
          "       3. 查询通讯录\n"
          "       4. 修改通讯录\n"
          "       5. 添加通讯录\n"
          "       6. 删除通讯录\n"
          "       7. 排序通讯录\n"
          "       8. 退出\n";
    int choose;
    cin>>choose;

    switch(choose){
        case 1:
            system("clear");

            mycreat();break;
        case 2:
            mydisplay();break;
        case 3:
            ;break;
        case 4:break;
        case 5:break;
        case 6:break;
        case 7:break;
        case 8:/*exit();*/break;
    }
    return;
}
```