**实验报告（一）**

姓 名： 孟岩

学 号： 111700426

专 业： 微电子科学与工程

年 级： 2017

班 级： 微电二班

指导老师： 潘林

实验时间： 2020.11.11

**数据结构与算法（C语言版）实验报告**

**实验一 线性表**

1. **实验内容**：

利用单链表实现下述应用

目的：线性表的建立、新增、插入、删除和查询

具体内容：学生成绩统计。问题描述如下：

（1）以“学生”为数据元素的线性表

（2）用typedef、struct写出其链式存储结构的类型定义，其中“学生”为复合型数据类型，包括：学号、姓名、性别、年龄、考试科目、考试成绩。

（3）统计考试科目的平均成绩，考试科目3门，分别是“高数”、“外语”和“物理”

（4）功能1：根据学号进行成绩录入（新增），删除

（5）功能2：根据学号进行成绩查询，成绩修改

（6）功能3：根据考试科目分别求总平均分、男女生平均分

（7）功能4：显示出各门成绩的最高分同学信息

**二、实验目的：**

**1.**线性表的应用，C语言和环境的熟悉

**2.**理解掌握线性表（顺序表、链表）的建立、新增、插入和删除的程序设计

**3.**线性表（顺序表、链表）的应用

三、**程序流程图**

1.定义链表，必要元素：学号，姓名，性别，年龄，考试科目，考试成绩

2.初始化链表，使用txt文件进行初始化

3.添加数据

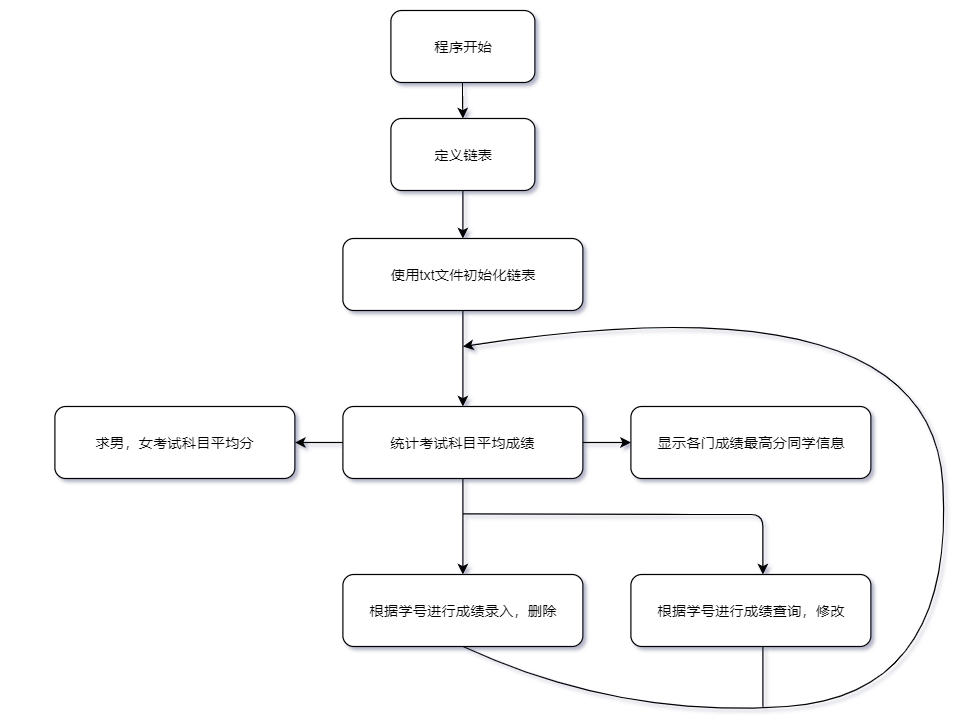
4.求平均值：高数，外语，物理

5.添加功能：根据学号，实现插入，删除

6.添加功能：根据学号，实现查询，成绩修改

7.添加功能：求各科总平均分，男女分别平均分

8.添加功能：求各科成绩最高分同学的信息并显示



四、**程序设计**

1、**数据类型定义：**

**使用class进行完全封装，将结构体与链表封装在class中**

class Student

{

    public:

    //定义链表的必要元素

        typedef struct LinkList\_Student

        {

            unsigned long student\_number;   //学号

            string name;                    //姓名

            string gender;                  //性别

            int age;                        //年龄

            int math\_grade;                 //高数分数

            int english\_grade;              //英语成绩

            int physics\_grade;              //物理成绩

            struct LinkList\_Student \*next;

        }LinkList\_Student,\*List;

    private:

        int size = 100;

        LinkList\_Student test\_Score = {0};

        string file\_addreess = "D:/code/data\_structure/LinkList/data.txt";//文件地址

public:

**void setSize(int setSize);**

**int getSize(void);**

**void split(const char \*s,vector<string>& strs, char delim);**

**void Init\_List();        //初始化链表,封闭操作**

**void Input\_List(void);   //将文件内容写入链表**

**void average(double arg[]);    //求平均成绩,输入数组存储**

**void highest\_grade(void);      //查找各科分数最高的同学**

**List seek\_student(unsigned long student\_number);    //查看输入学号的同学是否在链表中并返回地址**

**void query\_grade(void);        //根据学号查成绩**

**void amend\_grade(void);        //根据学号修改成绩**

**void delete\_student(void);     //根据学号删除**

**void add\_student(void);        //添加一条**

**Student(string s);**

**~Student();**

};

**2、主要代码**

//使用文件初始化链表

void Student::Input\_List(void)

{

    List p, L;

    L = &(this->test\_Score);

    fstream file;

    string s\_data;

    stringstream s\_change;

    vector<string> vs\_data;      //初始化容器

    file.open(this->file\_addreess);

    if (!file) {

        cout << "未找到相关文件，无法打开！检查文件路径！" << endl;

        exit(ERROR);

    }

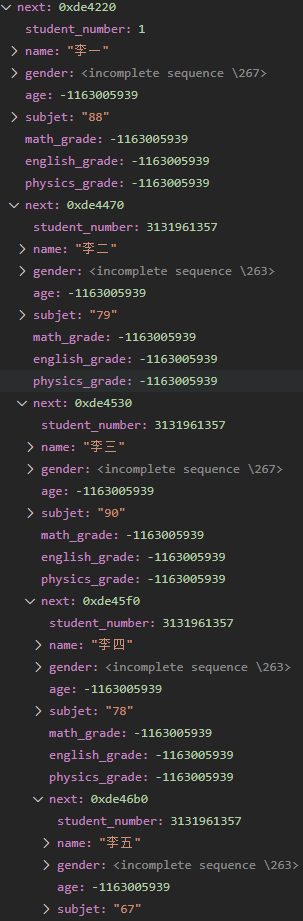
    getline(file,s\_data);   //跳过第一行

    while (file.good()) {

        getline(file,s\_data);

        this->split(s\_data.data(),vs\_data); //字符串

        p = new LinkList\_Student;

        s\_change<<vs\_data[0];  //将string转换成其它类型

        s\_change>>p->student\_number; //学号

        s\_change.clear();

        p->name = vs\_data[1];       //名字

        p->gender = vs\_data[2];     //性别

        s\_change<<vs\_data[3];       //年龄

        s\_change>>p->age;

        s\_change.clear();

        s\_change<<vs\_data[4];       //高数分数

        s\_change>>p->math\_grade;

  s\_change.clear();

        s\_change<<vs\_data[5];       //英语分数

        s\_change>>p->english\_grade;

 s\_change.clear();

        s\_change<<vs\_data[6];       //物理分数

        s\_change>>p->physics\_grade;

 s\_change.clear();

        p->next = NULL;

        vs\_data.clear();            //清空容器

}

//将字符串进行分割

void Student::split(const char \*s, vector<string>& strs, char delim = ' ')

{

    if(s == nullptr) {

        return;

    }

    const char \*head, \*tail;

    head = tail = s;

    while(\*head != '\0') {

        while(\*head != '\0' && \*head == delim) {

**图1 链表初始化部分内容**

            head++;

        }

        tail = head;

        while(\*tail != '\0' && \*tail != delim) {

            tail++;

        }

        if(head != tail) {

            strs.push\_back(string(head,tail));

            head = tail;

        } else {

            break;

        }

    }

}

//查询要查找的学号是否存在于链表，返回指针

Student::List Student::seek\_student(unsigned long student\_number)

{

    List p;

    p = this->test\_Score.next;

    while(p!=NULL && (p->student\_number != student\_number)){

        p = p->next;

    }

    if(p!=NULL)

        return p;

    else{

        cout<<"该学号不存在，请检查输入"<<endl;

        return NULL;

    }

}

//删除某个节点

void Student::delete\_student(void)

{

    List p,d;

    p = &(this->test\_Score);

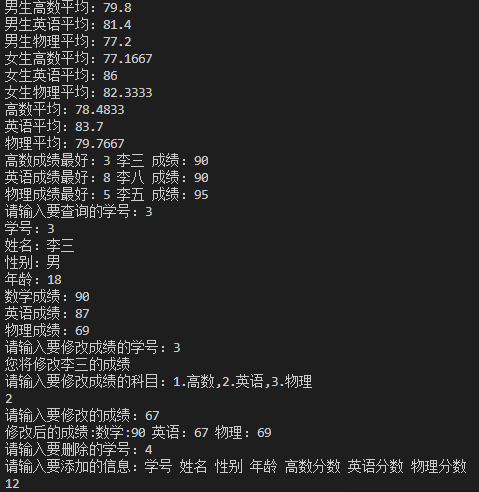
    unsigned int student\_number = 0;

    cout<<"请输入要删除的学号：";

    cin>>student\_number;

    while(p->next!=NULL && (p->next->student\_number != student\_number)){

        p = p->next;

    }

    if(p->next!=NULL){

        d = p->next;

        p->next = p->next->next;

        free(d);

    }

    else{

        cout<<"该学号不存在，请检查输入"<<endl;

    }

}

int main()

{

    Student s("D:/code/data\_structure/LinkList/data.txt");

    double average[9] = {0};//mm,me,mp,wm,we,wp,am,ae,ap

    s.Init\_List();

    s.Input\_List();

    s.average(average);

    s.highest\_grade();

    s.query\_grade();

s.amend\_grade();

s.delet\_student();

s.add\_student();

    getchar();

**图2 运行程序后的执行结果**

}

**五、一些心得**

1. 使用C++的过程中熟悉了类的封装与使用

2. 在编码过程中使用到了容器，并使用了数据流处理，使用了stringstream 进行string与其它数据类型之间进行转换

3.学会了对单链表进行熟练度增删改查操作

4.使用VSCode+mingw-w64进行编码配置，了解了较多C++编码的流程，掌握了shell的使用

5.使用Git+GitHub进行版本控制，掌握了Git的简单使用

6.由于shell输入无法支持中文，暂不知如何修改，添加信息时无法写入中文

[GitHub代码链接](https://github.com/MY-stone/data_structure)：https://github.com/MY-stone/data\_structure

[笔记链接：](http://note.youdao.com/noteshare?id=cb8aabd4352b5f422bc7ea1e3cf41a61)

http://note.youdao.com/noteshare?id=cb8aabd4352b5f422bc7ea1e3cf41a61