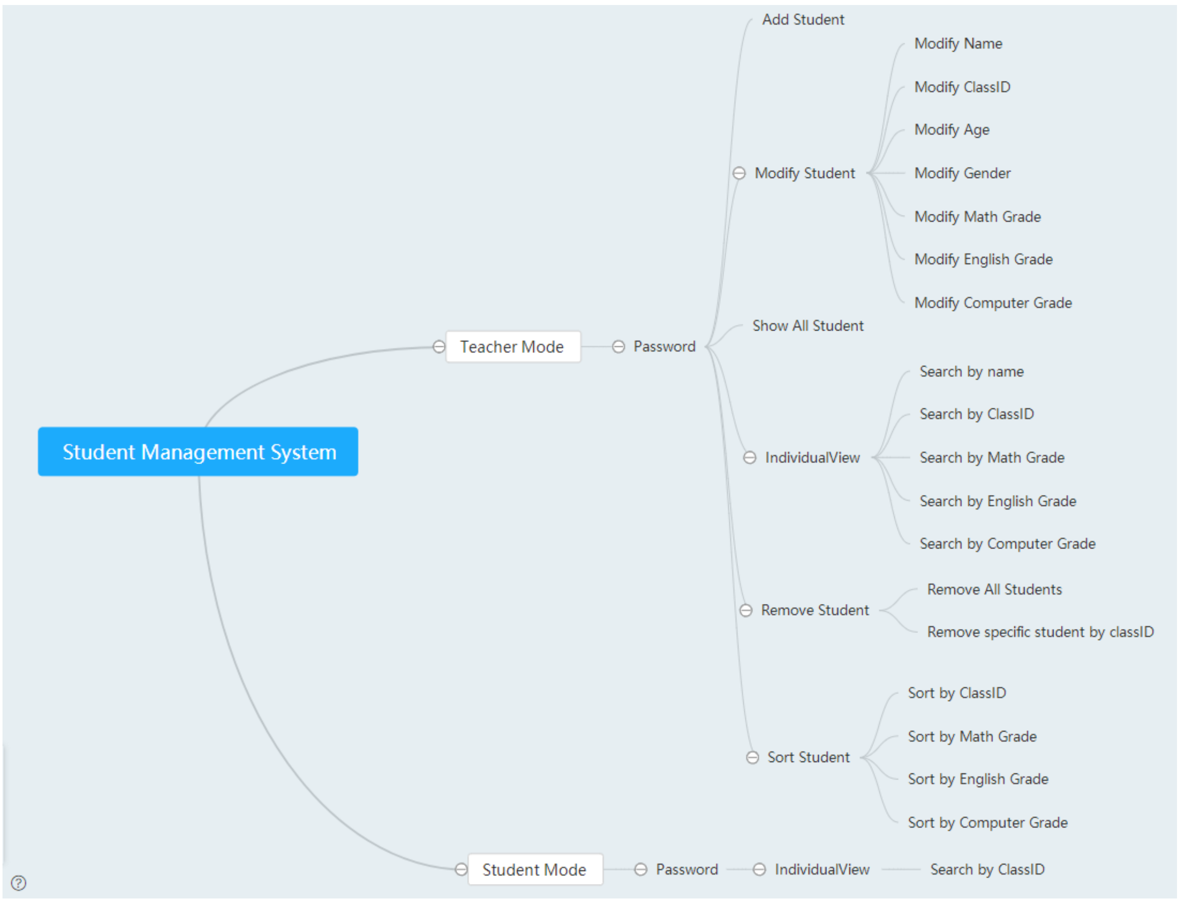
VG101 C++ final report
Student Management System

1. Introduction

Student management system is a system that can manage the students' information, including addition, modification, show, individual view, removement and sort. In the system, the user need to follow the tips to manage the students' information. The system has two mode: **Student Mode** and **Teacher Mode**. In the Student Mode, the user can only view the information of student by classID. In the Teacher Mode, the user can use all functions.

In the system, class is used to store the information of students and the functions that can manage the information. During the process of programming I encounter many problems because I am not so familiar with the grammar the C++. I tried to test again an again to solve them or search the solutions on the internet. As for the testing of the program, I will run the program step by step to check the function one by one.

Here is the structure of the system.



2. Methodology

2.1 Data structure

2.1.1 Essential classes and member function/variables

class is chosen but not structure because class is more powerful. In C++, there are many functions related to classes.

The system includes 4 class:

```
class Grade

class Student

class StudentFunction

class TeacherFunction
```

Student and Grade are used to store the information of the students. Studentfunction and Teacherfunction include the functions that can be used in Student Mode and Teacher Mode.

2.1.1.1 Class "Grade"

It includes three member variables:

```
char math[20]
char computer[20]
char english[20]
```

This class is used to store the grade of the student. This is taken as an independent class to show grade as a whole.

As for the type of data, char is chosen but not int because it's easy to check whether the input is number(error test, I will mention in the function part).

2.1.1.2 Class "Student"

It includes 5 member variables:

```
char name[20]
char classID[20]
char gender[20]
char age[20]
Grade g;
```

This class is used to store the basic information of the student.

The length of the member variables is 20. They are 20 because it will have more space for them to store information. The length of them are the same because it's easy to program other functions related to them (I don't need to worry about the problem of setting length).

2.1.1.3 Class "Student Function"

It includes 2 member functions:

```
void IndividualView();//to find specific information
friend void searchID();
```

IndividualView is used to view the information of students. SearchID means you can only search by classID in student mode.

searchID is set as an ordinary function to use it easily. And it is set as the friend function of this class to show their relationship.

2.1.1.4 Class "Teacher Function"

It includes 11 member functions:

```
void AddStudent(); //to add new information
void ModifyStudent(); //to modify information
void ShowAllStudent(); //to show all information
void IndividualView(); //to find specific information
friend void searchname();
friend void searchID();
friend void searchmath();
friend void searchenglish();
friend void searchcomputer();
void RemoveStudent(); //to remove information
void SortStudent(); //to sort information
```

The **void functions** are the basic function of the system. The **friend void functions** are the subfunction of IndividualView, which means the user can search the information of the students by name, classID, math grade, English grade and computer grade.

For the same reason as class student function, the search functions are placed in it show that they are a whole.

2.1.2 The flow of execution of each function.

2.1.2.1 Basic show function

These functions are used to show specific information.

`void error()`, when the input is error, prompt error.

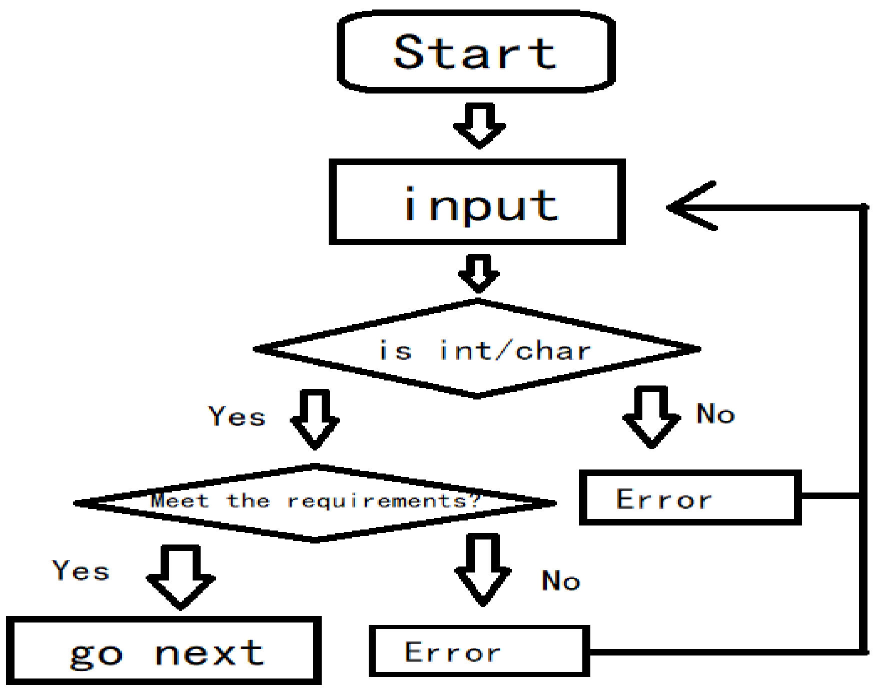
`void clear()`, to clear the screen.

`void finish()`, prompt you to input ‘0’ and then end the function. It has an error test system, to test if your input is 0.

2.1.2.2 Judge function

These functions are used in error test system.

Error Test System



When the input doesn’t meet the format, an error message will be presented.

`int lengthofchar(char* str)` to check the length of the input

`bool is_number(char* str)` to judge whether the input is number

ASCII is used to judge. Char will be change into ASCII, if they are bigger than 48 and small than 57, they are number.

`bool is_character(char* str)` to judge whether the input is character

ASCII is used to judge. Char will be change into ASCII, if they are bigger than 65 and small than 90, or bigger than 97 and small than 122, they are character.

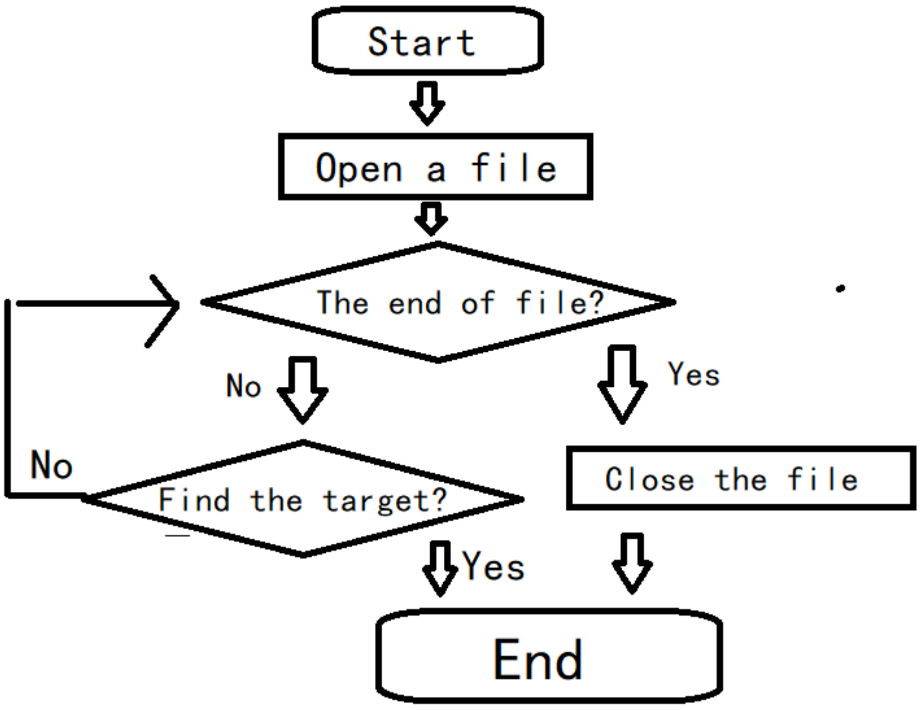
`bool rangejudge(int up, int down, char* teststr)` to judge the range of the input

Up is the upper bound and down is the lower bound. The teststr will be changed into int first, and judge whether it’s out of range.

2.1.2.3 Search function

These functions are the subfunction of the `IndividualView function`. Using these functions, you can search information by different labels.

```
void searchname(char* str)

void searchID(char* str)

void searchmath(char* str)

void searchenglish(char* str)

void searchcomputer(char* str)
```



It will find the target until it's found or the end of the file.

2.1.2.4 Sort function

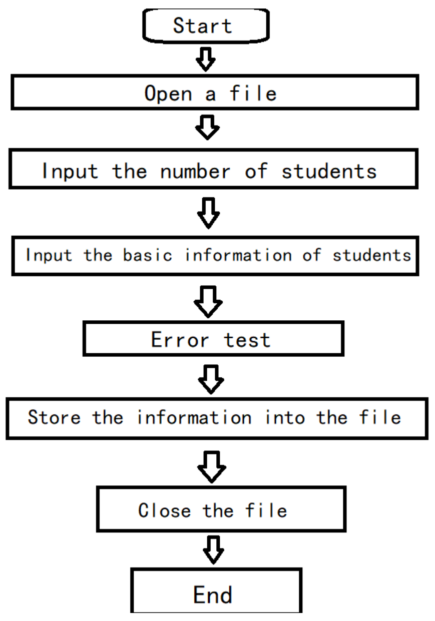
```
static bool IDCompare(const Student& a, const Student& b)

static bool MathGradeCompare(const Student& a, const Student& b)

static bool EnglishGradeCompare(const Student& a, const Student& b)

static bool ComputerGradeCompare(const Student& a, const Student& b)
```

These functions are the subfunction of the `SortStudent function`. Using these functions, you can sort information by different ways. They are **comparators** of **sort(STL function) function**.

2.1.2.5 Teacher member function

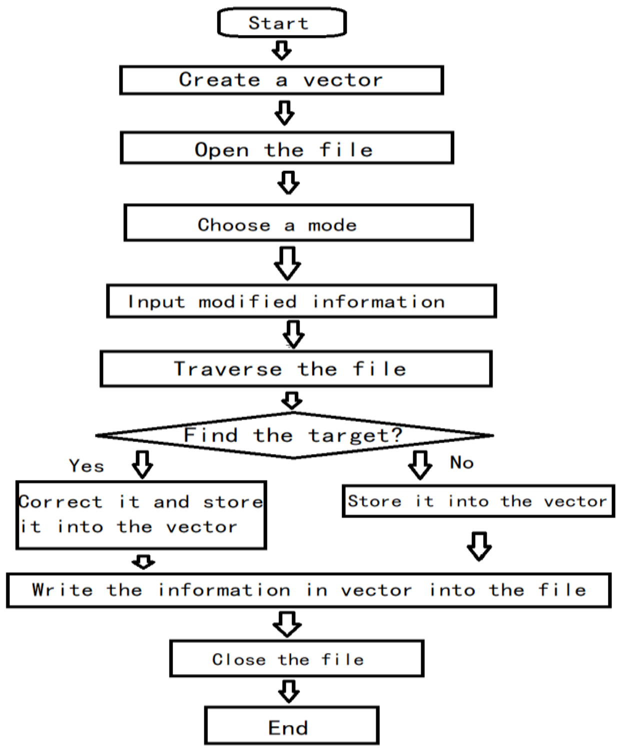
These functions are the basic functions of the system. Every function has error test system.

```
void TeacherFunction::AddStudent() //to add new student
```


Start
Open a file
Input the number of students
Input the basic information of students
Error test
Store the information into the file
Close the file
End

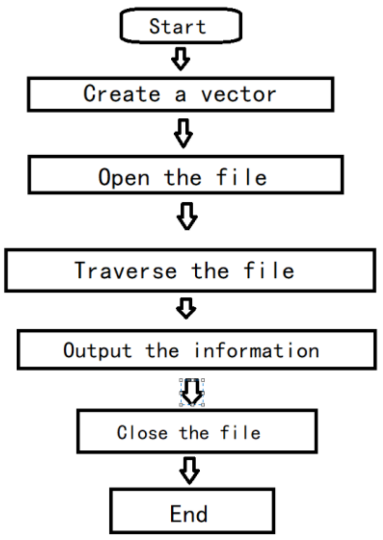
Fstream is used to write the information into the file.

```
VoidTeacherFunction::ModifyStudent()
```



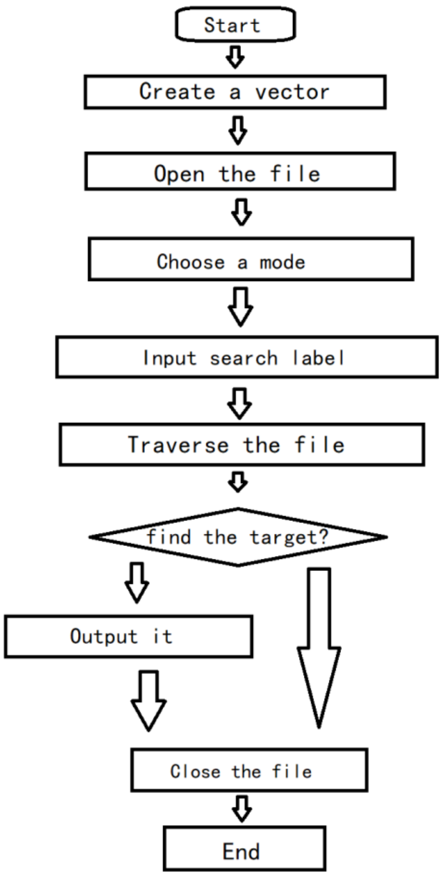
First, the user will choose the type of information he wants to modify. Then input the modified information and the program will find the student you want to find in the file. Correct the information in the vector, and store the new vector into the file.

```
void TeacherFunction::ShowAllStudent()
```



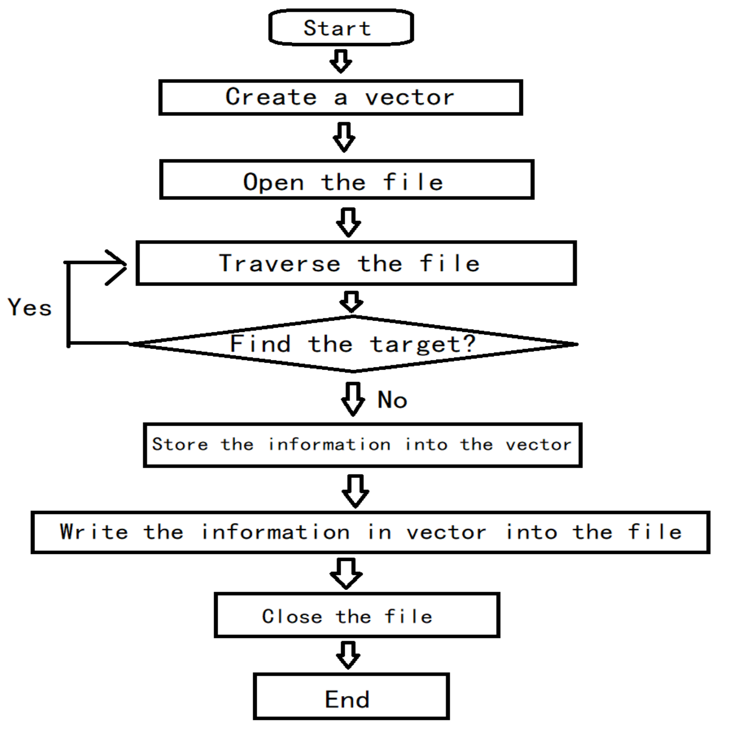
Traverse the file and output every student the program can find.

```
void TeacherFunction::IndividualView()
```



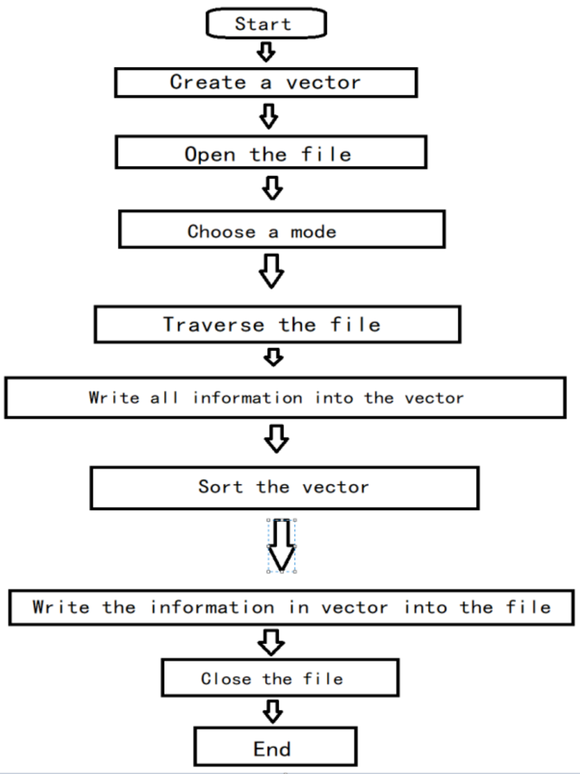
First, the label of searching needed to be decided. Then the program searches the file according to the information the user supply. When the student is found, his information will be show.

```
voidTeacherFunction::RemoveStudent()
```



The target won't be stored in the vector but others will, so finally the vector doesn't store the information of target. And then write the information in vector into the file.

```
void TeacherFunction::SortStudent()
```



In “Sort the vector”, “sort” function in STL is used, and the comparators are the search function.

2.1.2.6 Student member function

```
void StudentFunction::IndividualView()
```

It’s the same as it in Teacher function, but it can only search by classID.

2.2 Technical problem

2.2.1 File input and output

At first, I was not so familiar with the file input and output in C++, I didn’t know how to find the information in the file, and how to store the information into the file. But finally, with the help of internet, I knew how to make it by reading some article in CSDN.

Another problem I encountered about file input and output was that I found that the loop of file traversal would go one more time, and it caused many disasters. At the end, I set a “if” to prevented it from happening again.

2.2.2 The data type of the student information

As many people think, int may be a good type of student information like age and classID. But I found that it’s quite complex to do error test by **is_number** and **is_character** functions, so I changed them into char at the end

2.2.3 The use of Visual Studio and Dev-C++

I got used to using devc++ in the past. VS was another world for me. So, I decided to program in devc++ at first and copied the code to VS when I finished. But in the process of programming, I met some really strange problems. I couldn’t run the program in devc++, but in VS I made it. Therefore, I tried to use VS from that time. It took me lots of time to get used to using VS.

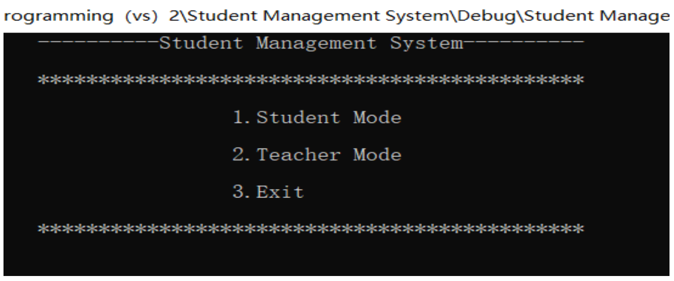
2.3 Testing of the program

I run the program again and again to test every function of the system, ensuring they are correct. For the specific problem, I will try to run the program step by step.

As for the error test, I try to input error information in the system in different operation interface to confirm that the error test system is correct

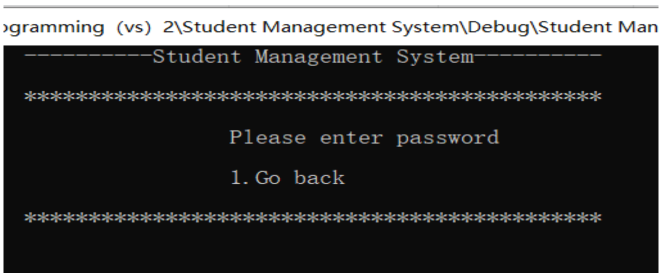
3. Results

3.1 Start inferface



```
rogramming  (vs)  2\Student Management System\Debug\Student Manage
  ----------Student Management System----------
  *********************************************
                1.Student Mode
                2.Teacher Mode
                3.Exit
  *********************************************
```

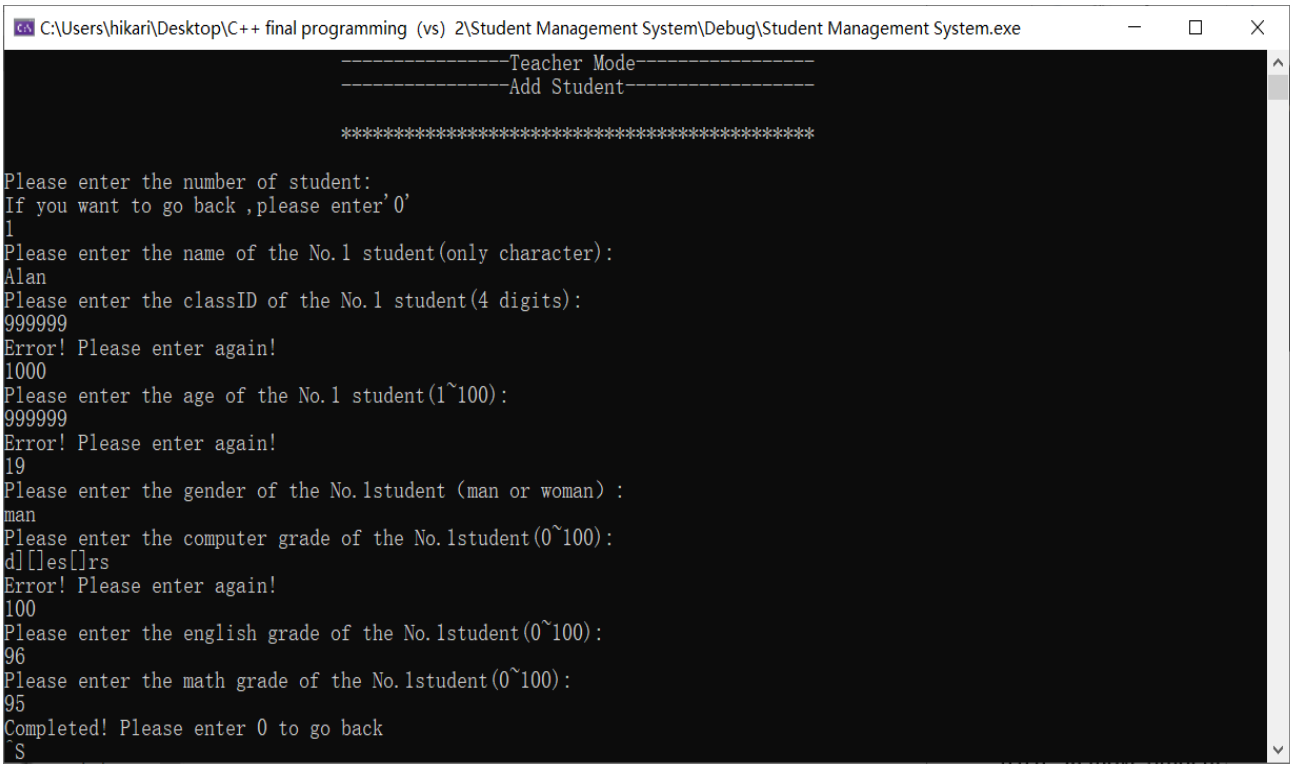
3.2 Password



```
›gramming  (vs)  2\Student Management System\Debug\Student Man
 ----------Student Management System----------
 *********************************************
                 Please enter password
                 1.Go back
 *********************************************
```

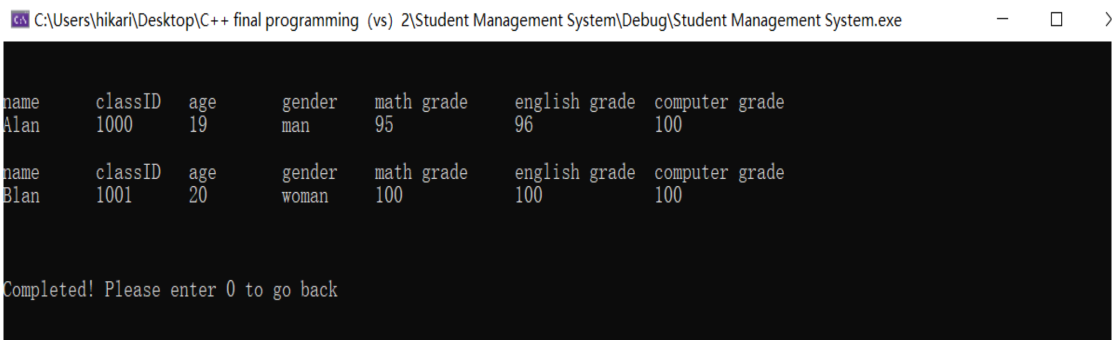
3.3 Teacher Mode

3.3.1 Add Student



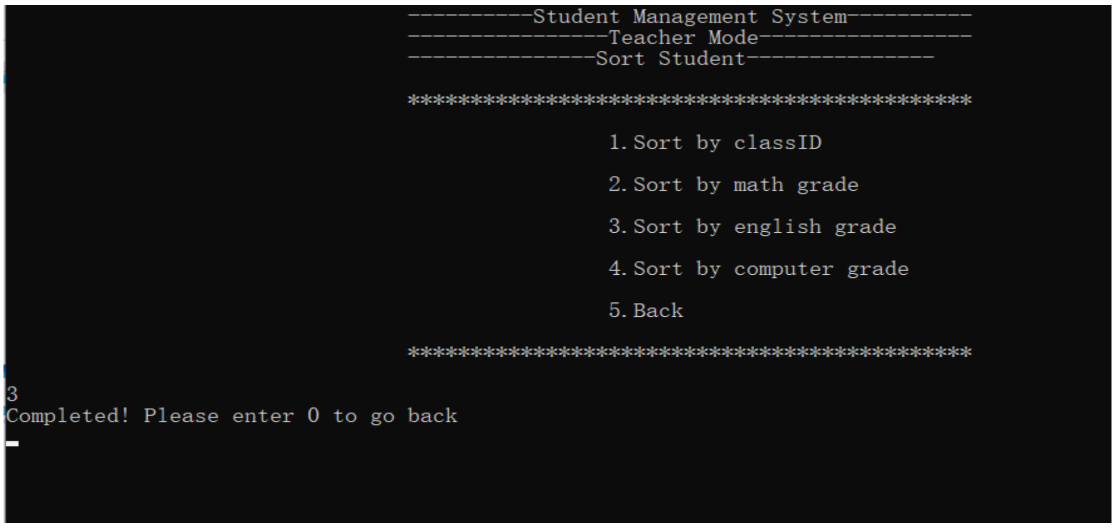
```
C:\Users\hikari\Desktop\C++ final programming  (vs)  2\Student Management System\Debug\Student Management System.exe
                                ---------------Teacher Mode----------------
                                ---------------Add Student-----------------

                                *********************************************
Please enter the number of student:
If you want to go back ,please enter'0'
1
Please enter the name of the No.1 student(only character):
Alan
Please enter the classID of the No.1 student(4 digits):
999999
Error! Please enter again!
1000
Please enter the age of the No.1 student(1~100):
999999
Error! Please enter again!
19
Please enter the gender of the No.1student (man or woman) :
man
Please enter the computer grade of the No.1student(0~100):
d][]es[]rs
Error! Please enter again!
100
Please enter the english grade of the No.1student(0~100):
96
Please enter the math grade of the No.1student(0~100):
95
Completed! Please enter 0 to go back
^S
```

3.3.2 Show All Student

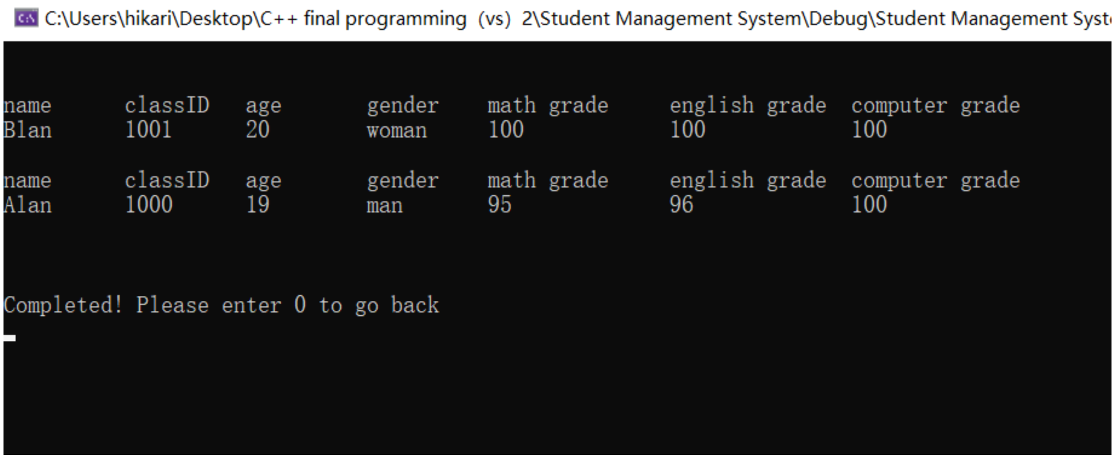


```
C:\Users\hikari\Desktop\C++ final programming  (vs)  2\Student Management System\Debug\Student Management System.exe

name      classID   age       gender    math grade    english grade  computer grade
Alan      1000      19        man       95            96             100

name      classID   age       gender    math grade    english grade  computer grade
Blan      1001      20        woman     100           100            100


Completed! Please enter 0 to go back
```

3.3.3 Sort Student

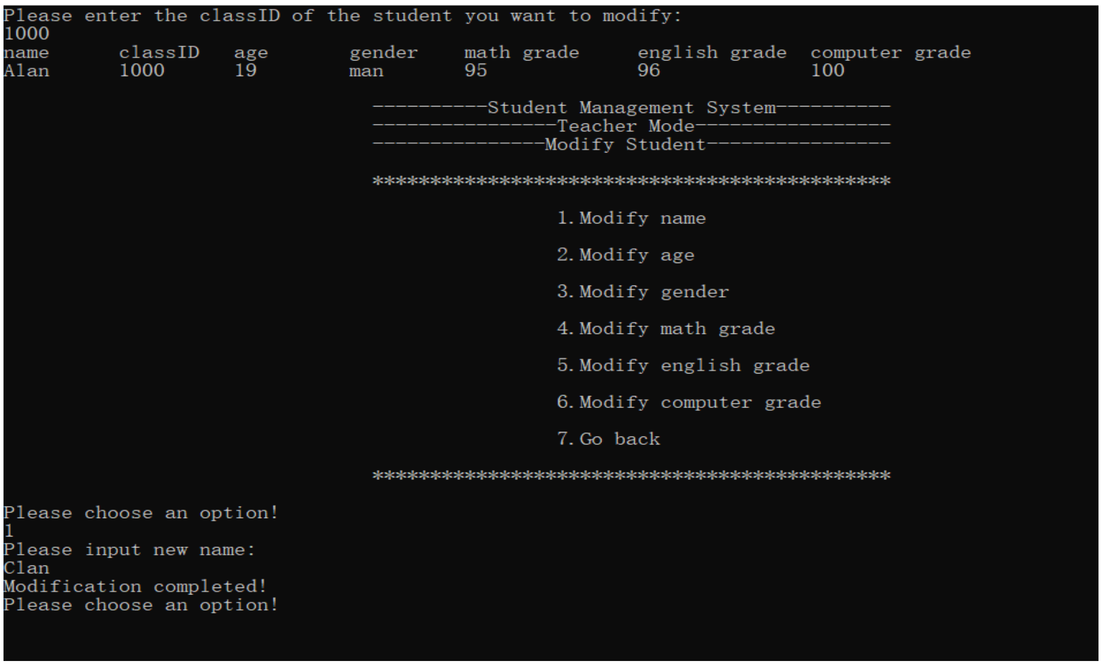


```
                                   ----------Student Management System----------
                                   ----------------Teacher Mode-----------------
                                   ---------------Sort Student---------------

                                   *********************************************

                                                1.Sort by classID

                                                2.Sort by math grade

                                                3.Sort by english grade

                                                4.Sort by computer grade

                                                5.Back

                                   *********************************************

3
Completed! Please enter 0 to go back
```

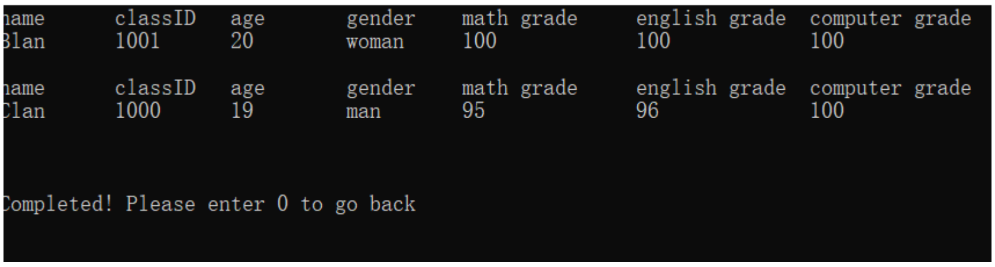


```
C:\Users\hikari\Desktop\C++ final programming  (vs)  2\Student Management System\Debug\Student Management Syst

name      classID   age       gender    math grade    english grade  computer grade
Blan      1001      20        woman     100           100            100

name      classID   age       gender    math grade    english grade  computer grade
Alan      1000      19        man       95            96             100


Completed! Please enter 0 to go back
```

3.3.4 Modify Student

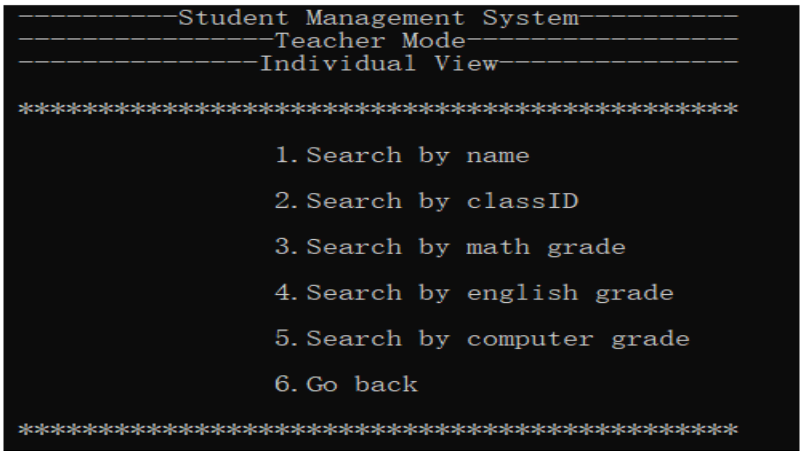


```
Please enter the classID of the student you want to modify:
1000
name      classID   age       gender    math grade     english grade  computer grade
Alan      1000      19        man       95             96             100
                                ----------Student Management System----------
                                ----------------Teacher Mode-----------------
                                ---------------Modify Student---------------

                                *********************************************

                                                1.Modify name

                                                2.Modify age

                                                3.Modify gender

                                                4.Modify math grade

                                                5.Modify english grade

                                                6.Modify computer grade

                                                7.Go back

                                *********************************************
Please choose an option!
1
Please input new name:
Clan
Modification completed!
Please choose an option!
```

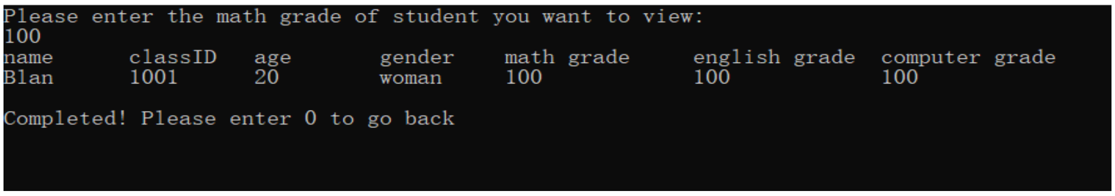


```
ame       classID   age       gender    math grade     english grade  computer grade
lan       1001      20        woman     100            100            100

ame       classID   age       gender    math grade     english grade  computer grade
lan       1000      19        man       95             96             100


ompleted! Please enter 0 to go back
```

3.3.5 Individual View

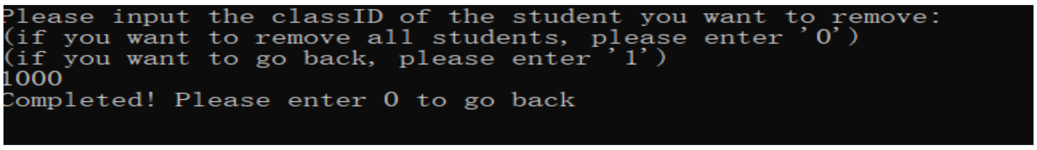


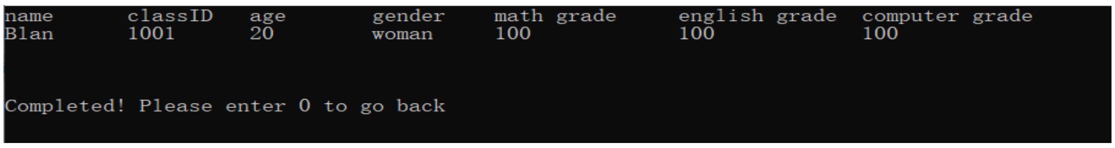
```
 ----------Student Management System----------
 ----------------Teacher Mode-----------------
 ---------------Individual View---------------

 *********************************************

                1.Search by name

                2.Search by classID

                3.Search by math grade

                4.Search by english grade

                5.Search by computer grade

                6.Go back

 *********************************************
```



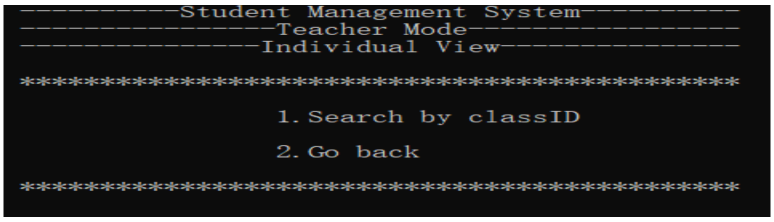
```
Please enter the math grade of student you want to view:
100
name      classID   age       gender    math grade     english grade  computer grade
Blan      1001      20        woman     100            100            100

Completed! Please enter 0 to go back
```

3.3.6 Remove Student


```
Please input the classID of the student you want to remove:
(if you want to remove all students, please enter '0')
(if you want to go back, please enter '1')
1000
Completed! Please enter 0 to go back
```


```
name      classID   age       gender    math grade     english grade  computer grade
Blan      1001      20        woman     100            100            100


Completed! Please enter 0 to go back
```

3.4 Student Mode(Individual View)


```
----------Student Management System----------
----------------Teacher Mode-----------------
---------------Individual View---------------

*********************************************
                1. Search by classID

                2. Go back

*********************************************
```

4. Conclusion and future development

4.1 Summary

During the process of programming, I gained lots of experience.

The first thing is the experience of using VS and C++ code. Now I can program in VS by C++ code more skillfully than past.

Another thing is the points to note when writing large programs. We need to decide many things at first such as the important class. These things just like the foundation of a building, we need to take care of them. Templated programming ideas is also significant, it can reduce hundreds of works we need to do.

Last but not least, it's the spirit of programming. We need patience and carefulness during the process of programming. A miss is as good as a mile, we must care every small error, or maybe it will waste us lots of time.

It's a **fabulously interesting** process, thanks to teachers for giving me this chance to do such a project.

4.2 Extension

If I have more time, I can do much improvement.

More functions of the system. Such output the information into a new file, show the average score or excellent rate of the students, etc.

More beautiful interface. The interface now is too simple.

More concise code. It's my first time to code large program, so there are many codes which are used repeatedly. To simplify my program, I can change these codes into template.

5. Appendix：User Manual

5.1 Introduction

5.1.1 The purpose of this user manual

This user manual was written to explain to the user how to use the system.

5.1.2 The function of this system

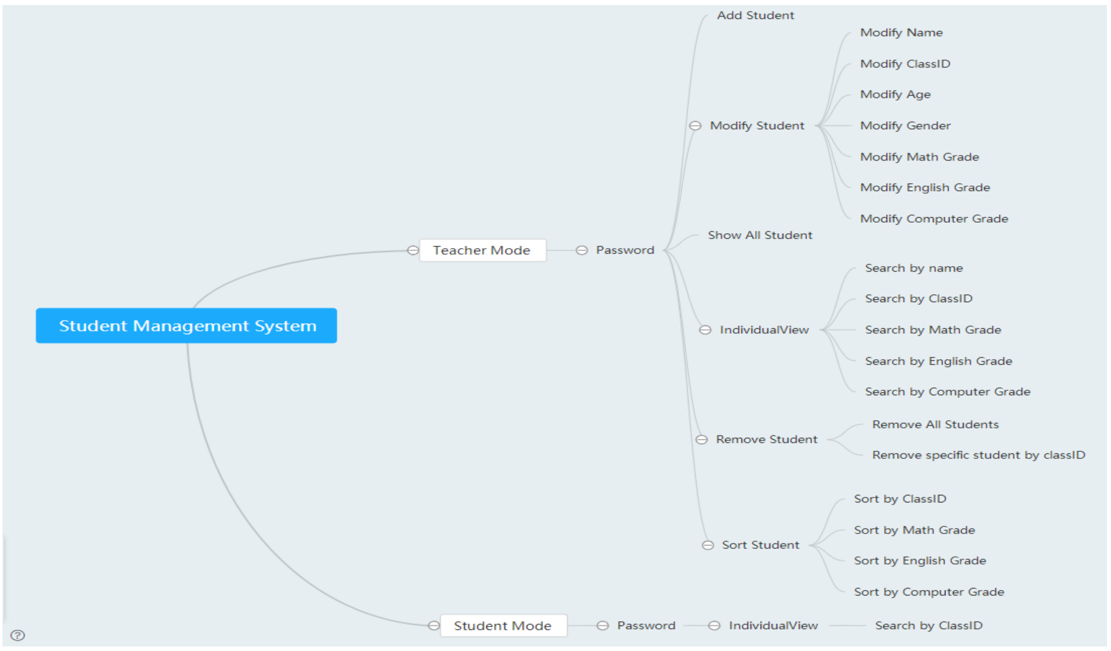
This student management system can help you store student information and management them. In this system, you can add, modify, remove the information of the students and so on.

5.2 The use of this system

5.2.1 Open the system

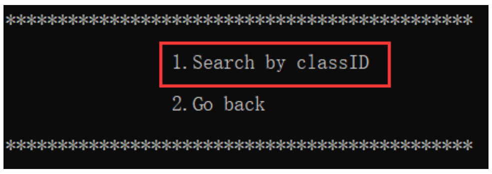
Find the .sln file and run it by visual studio in "\C++ final programming（vs）\Student Management System". Or you can find the .exe file in "\C++ final programming（vs）\Student Management System\Debug" and run it directly.

5.2.2 The Structure of the system



5.2.3 Mode choice

Just follow the tips and input number to choose the modes or functions you want.



```
*********************************************

                1.Search by classID

                2.Go back

*********************************************
```

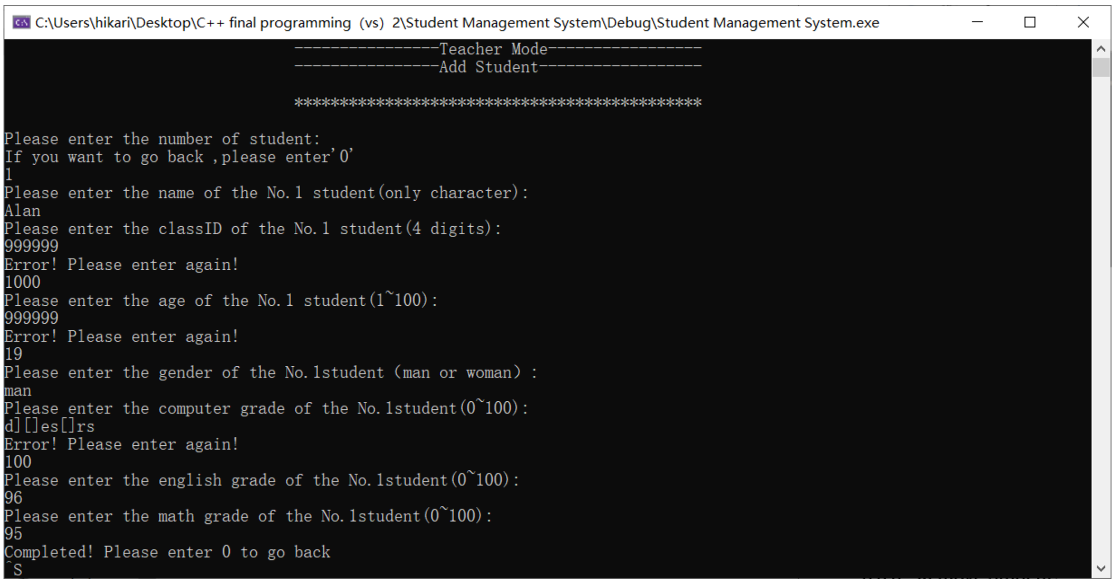
Like here, you need to input '1' to choose search function.

5.2.4 Password

To prevent stranger from entering this system, password is set. You can find the password in the "readme.txt" file.

5.2.5 Implementation of specific functions

Follow the tips and input the information it needs.



```
C:\Users\hikari\Desktop\C++ final programming (vs) 2\Student Management System\Debug\Student Management System.exe
                                ----------------Teacher Mode-----------------
                                ----------------Add Student------------------

                                *********************************************

Please enter the number of student:
If you want to go back ,please enter'0'
1
Please enter the name of the No.1 student(only character):
Alan
Please enter the classID of the No.1 student(4 digits):
999999
Error! Please enter again!
1000
Please enter the age of the No.1 student(1~100):
999999
Error! Please enter again!
19
Please enter the gender of the No.1student (man or woman) :
man
Please enter the computer grade of the No.1student(0~100):
d][]es[]rs
Error! Please enter again!
100
Please enter the english grade of the No.1student(0~100):
96
Please enter the math grade of the No.1student(0~100):
95
Completed! Please enter 0 to go back
^S
```

Like here, you need to follow the tipis and input specific information of students to add new student. This system can test error, so don't worry that you have a wrong input.

5.2.6 The storage of the information

The information will store in the file name "student.txt", but don't try to change the information by modify them in the file directly. It will cause error. You can find the file in "\C++ final programming（vs）\Student Management System\Student Management System".

5.3 Attention

Do not try to change the information in the "student.txt" file, it will cause error.