



**Java语言程序设计**

Introducing to Java Coursework: Student Course Management System

**Major:** Computer Science

**Student′ Name (In Pin-Yin):** Han Yichen

**Student′ Name (in Chinese):** 韩一尘

**Student′ ID No.**： 22722051

**Tutor′ Name**： Yuan Jidong

6 December 2023

Introducing to Java Coursework: Student Course Management System

Contents

**1. Abstract…………………………………………………….3**

**2.** Basic Program Structure and UML Diagram…………..3

**3.** Introduction to Each Module of the Program …………4

1. Coursework………………………………………………4
2. Person…………………………………………………….5

i. Teacher…………………………………………………5

ii. Student………………………….…………………….6

1. Course……………………………………………………7

4. Introduction to Program Operation Process………..8

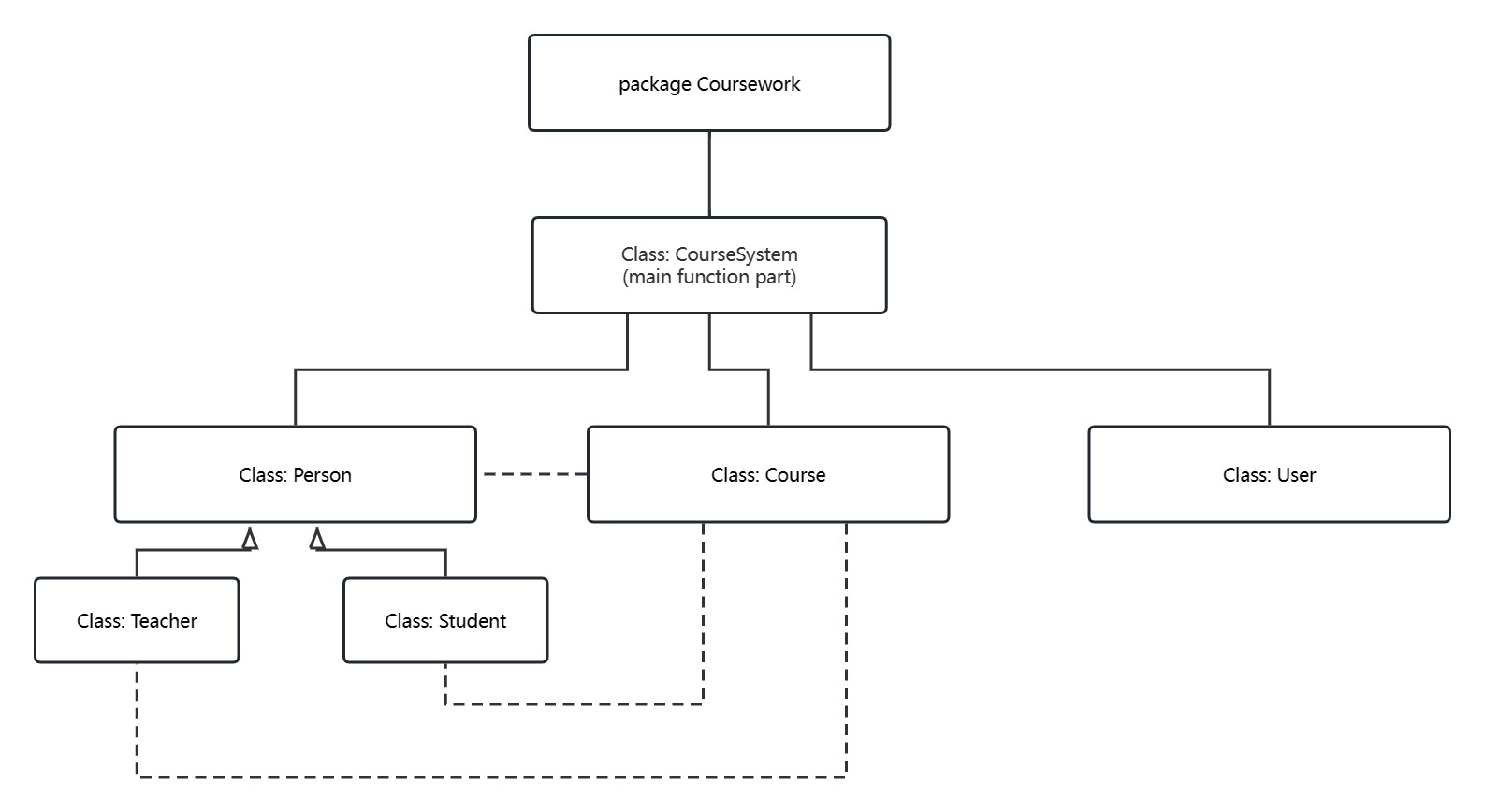
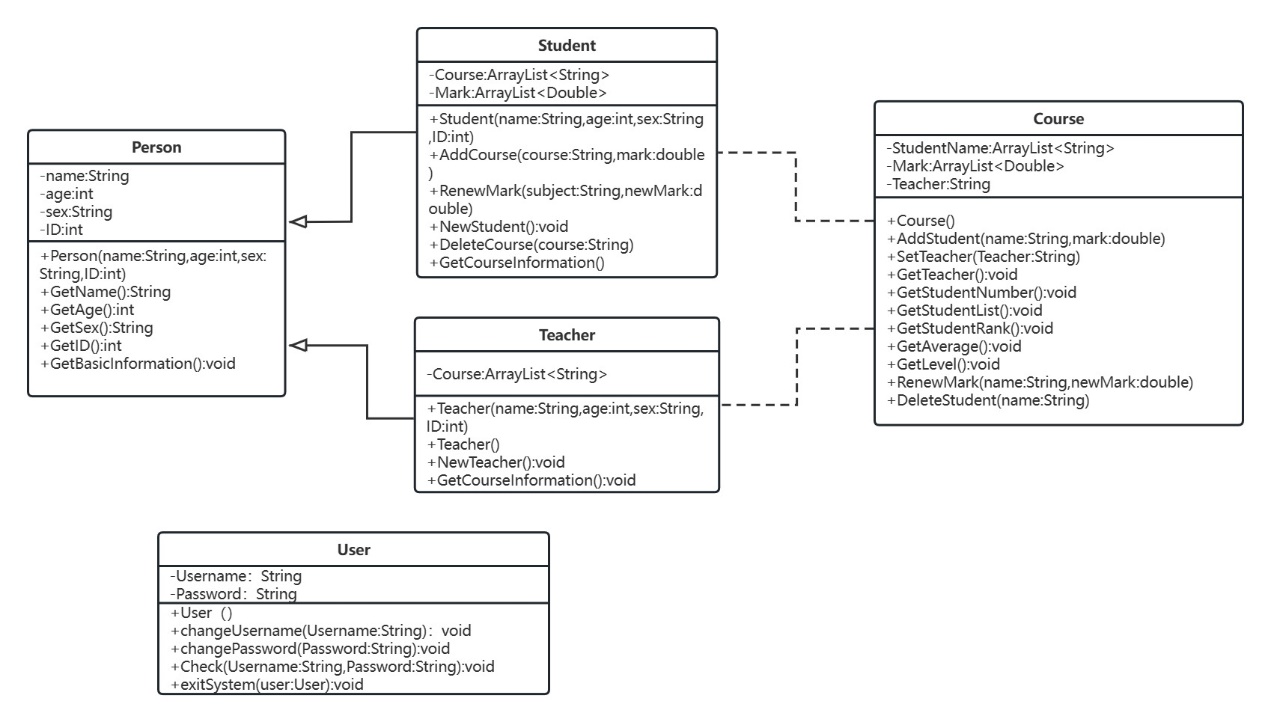
**5. Summary…………………………………………….…11**

**6. Thanks……………………………………………….…11**

1. **Abstract**

The present program is a student course management system designed in Java language. It aims to design and implement the management of student information by utilizing concepts such as class inheritance and polymorphism, as well as utilizing inherent classes like ArrayList and HashMap. This system provides convenient functionality, including the ability to add, delete, modify, and search for student, teacher, and course information, as well as account login and logout features. Through this system, users can effectively manage student and course information, fulfilling the required basic functionalities.

**2. Basic Program Structure and UML Diagram**

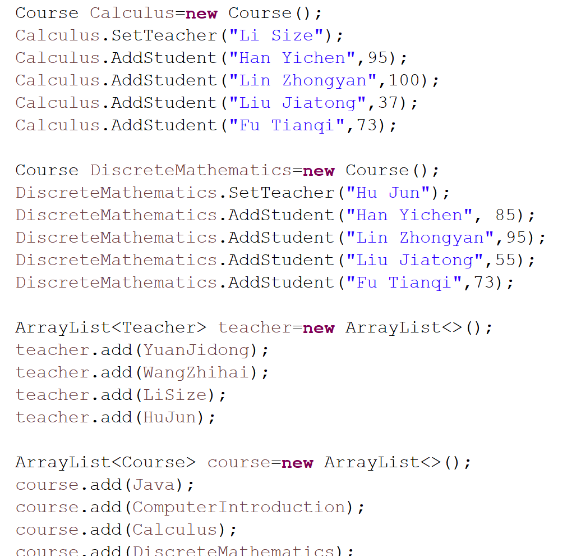
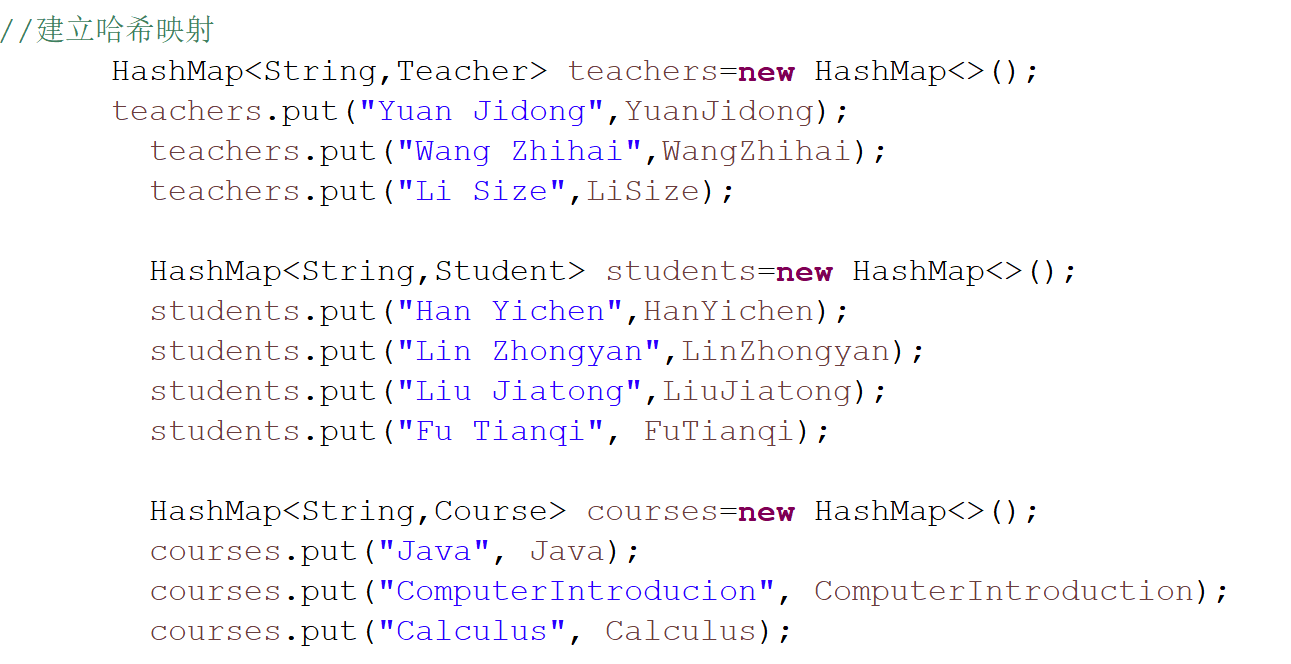
As shown in Figure 1, this figure shows the basic structure of this program: all classes are encapsulated in a package Coursework, with Class CourseSystem as the main body responsible for the function of main method. This Class has three subclasses: Person, Course, and User. The Person class also includes Teacher and Student subclasses, and the information and functionality of the Course class are closely related to Person and its subclasses. As shown in Figure 2, this diagram shows the UML diagrams of the five classes designed in the program. From the diagram, it is clear to see the methods, their calling rules, and visible ranges contained in each class. In the following article, we will describe our program from two perspectives: the functions of each module and the entire program operation process.

**Figure 1**

**Figure 2**

1. **Introduction to Each Module of the Program**

**1.Coursework**

This Class is the core part of the entire program, calling intrinsic classes such as java. util. Scanner, java. util. HashMap, and java. util. ArrayList, which assume the main method function of the program. In this Class, in order to facilitate program testing and usage, we first entered some initial basic information and entered it into the ArrayList. And the relationship between instances of different classes and Strings was established through Hash Map to facilitate calling and searching for corresponding content in the later stages of the program. Next, we use processes such as Scanner, loop nesting, and if-else statements to prompt users to complete corresponding operations according to system prompts to add, delete, add, modify, and log in and log out of the information. (As Figure 3)

**Figure 3-1**

**Figure 3-3**

**Figure 3-2**

**2.Person**

 This class contains basic personal information content such as name, age, sex, ID, etc. It also includes initialization methods for Person and implements functions such as returning all kinds of basic information. Class Person contains two inheritance classes: Student and Teacher, which clearly share many common properties, so this inheritance relationship is reasonable.

**Figure 4**

* 1. **Teacher**

This class inherits from Person, therefore it has the same methods inherited from Person, and includes a private ArrayList<String> type Course which is used to record the teacher's teaching course. At the same time, the Teacher class also implements its own initialization methods called through super with and without parameters. AddCourse() implements adding course names to Course, NewTeacher() implements basic information for creating new teachers, and GetCourseInformation() returns the teaching information of the teacher.



**Figure 5**

* 1. **Student**

This class also inherits from Person, so it also has methods that inherit from Person. But he also includes a Course for ArrayList<String>and a Mark for ArrayList<Double>, which are used to record the student's course and score, respectively, and to merge these two lists using HashMap. The Student class also has its own instantiation method. AddCourse() implements adding courses to students, RenewMark() modifies specified course scores, NewStudent() creates basic student information, DeleteCourse() deletes students' courses, and GetCourseInformation() returns students' course and score information.****

**Figure 6**

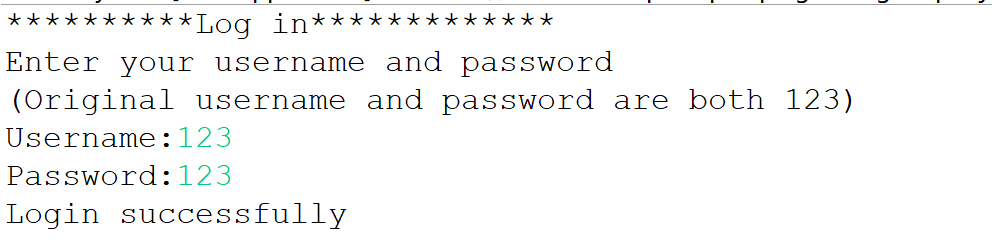
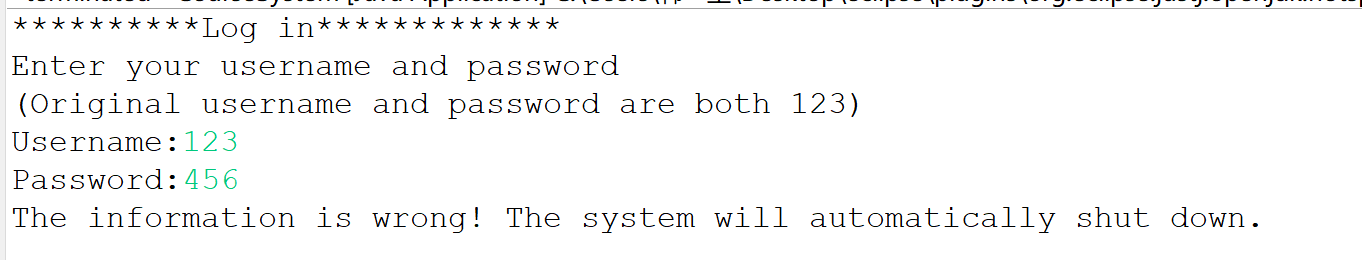
**3.Course**

This class does not inherit from any class, but is closely related to Person and its inherited classes. In this class, it has a parameterless initialization of Course (). AddStudent (String name, double mark) implements adding students, SetTeacher (String Teacher) implements setting teachers for courses, GetTeacher() returns teacher names, GetStudentNumber() returns student numbers, GetStudentList() returns student ID tables, GetStudentRank() returns student ranking tables, GetAverage() returns average scores, GetLevel() returns distribution of course grades, RenewMark (String name, double newMark) implements the update of specified course scores, and DeleteStudent (String name) implements the deletion of specified student information

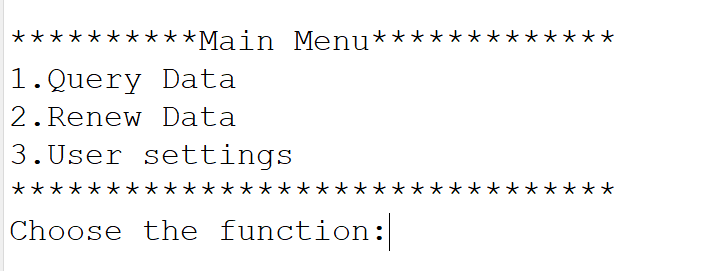


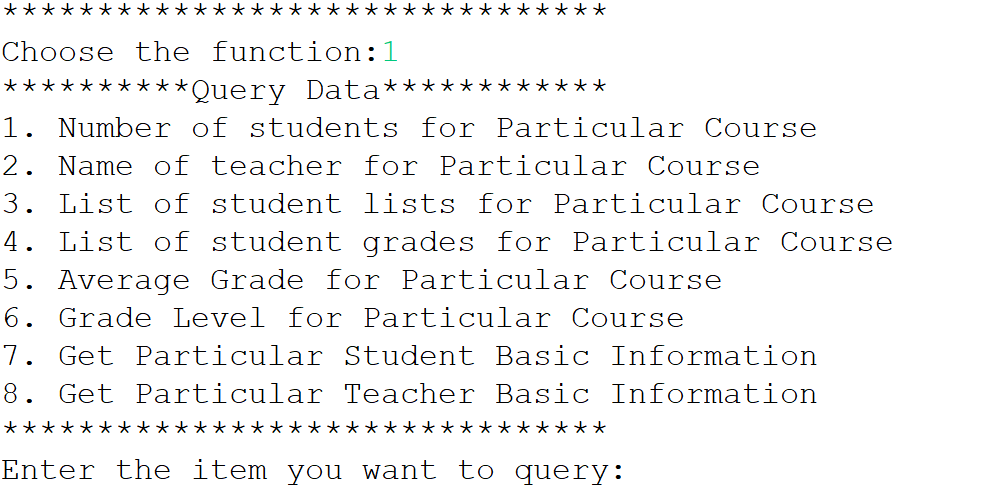
**Figure 7**

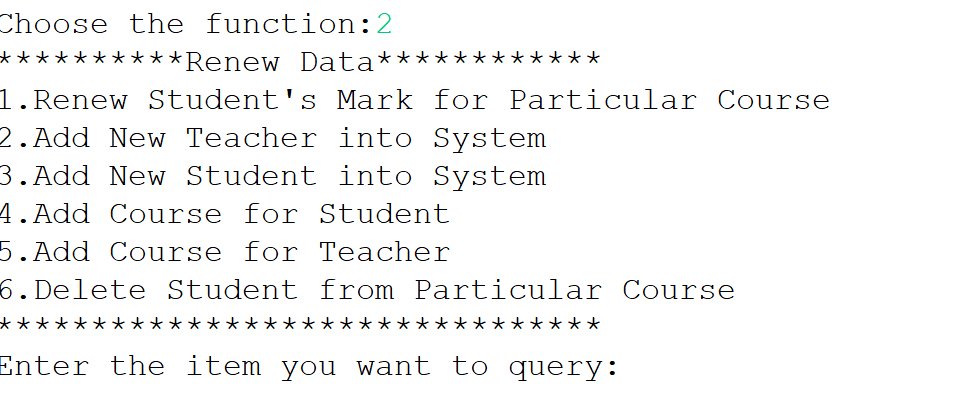
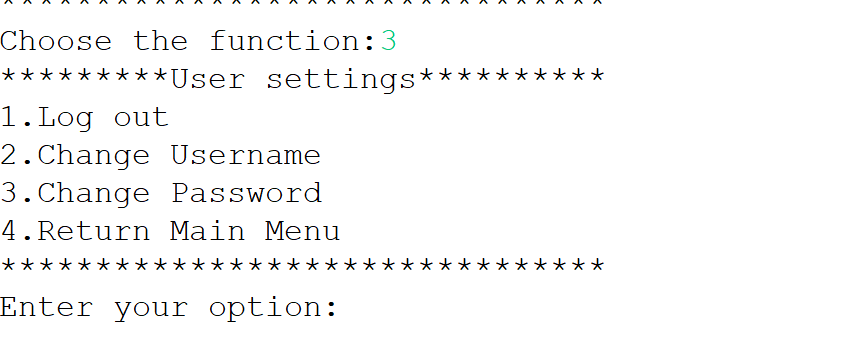
1. **Introduction to Program Operation Process**

 Run the program. The program first displays the interface for entering the username and password. Only by entering the password correctly can you enter the score system, otherwise you will exit the running.

**Figure 8**

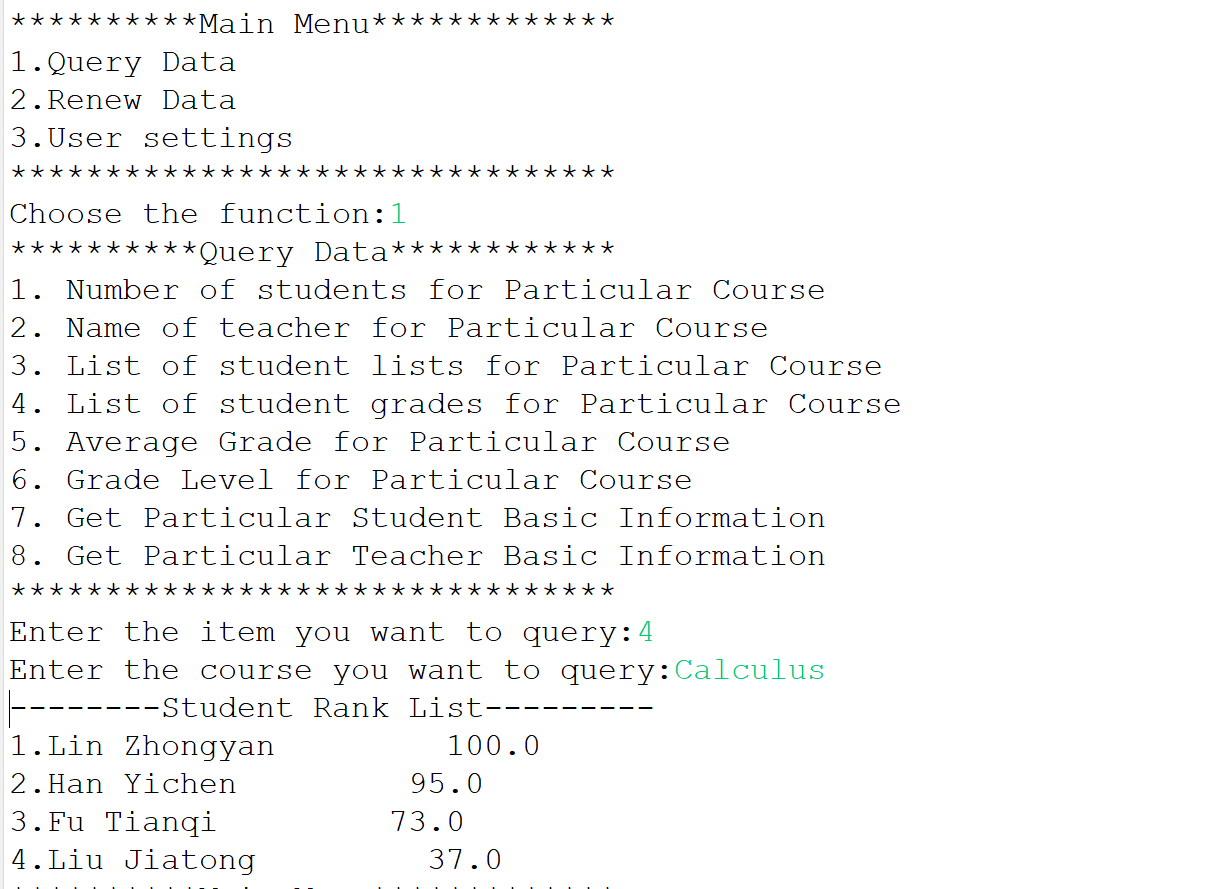
Then, we will see the Home screen of the program running, which respectively shows the functions of information query, modification and account setting. At this time, you can enter the corresponding options. If the option is wrong, it will automatically report an error and continue to return to the Main Menu. The following figure shows the interface of the main menu and different submenus. 



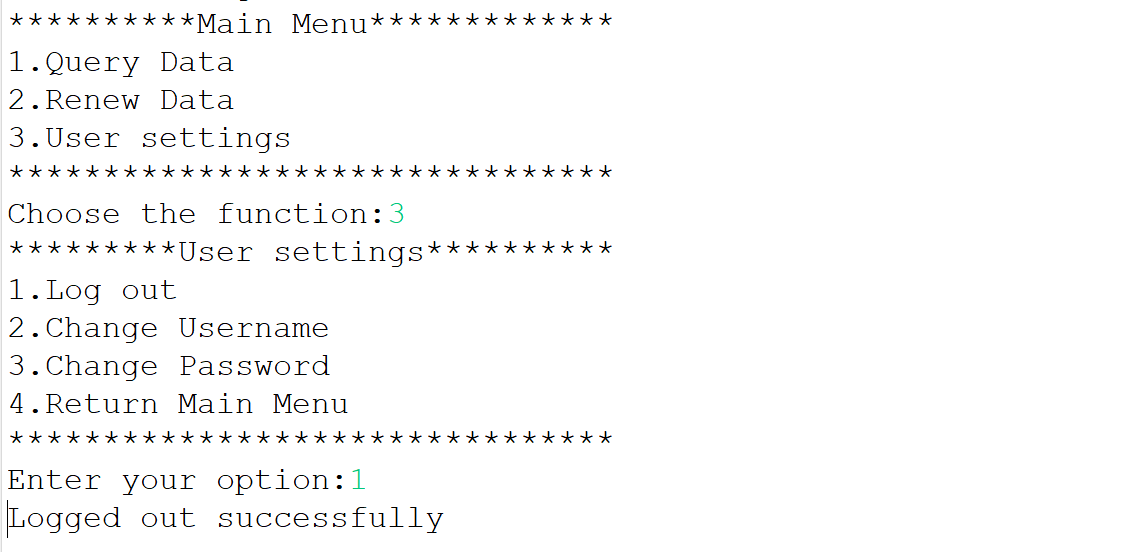
**Figure 9**

To illustrate the specific operating functions, let's take the course ranking of the query function as an example. Enter the main menu, enter option 1, enter the query submenu, and then enter option 4 to enter the search interface. Assuming we need to search for relevant information about calculus courses, we can enter "Calculus" and press Enter. The system will immediately display the student score ranking table for calculus courses.



**Figure 10**

Finally, we enter option 3 in the main menu interface to enter the user settings interface. By entering option 1, we can successfully exit the system.



**Figure 11**

**The above is a demonstration of the program's operation.**

1. **Summary**

**In this Java program, I designed a student information management system and implemented some basic functions related to information management. And it can make the program run basically normally. In summary, the Coursework of this Java class was extremely beneficial, as it not only helped me improve my programming skills, but also improved my learning ability. At the same time, it also allowed me to face my shortcomings. All of this has laid the foundation for my future computer learning.**

1. **Thanks**

Thanks to Mr. Yang Yining for his hard work in correcting. Thanks to Mr. Yuan Jidong for his diligent teaching. It can be said that this Java program was finished mediocrely, but I am still very grateful for the efforts of myself, the teaching assistant Yang and the teacher Yuan.

**End the Paper**