Computer Programming 2
Final Exercise 2

Problem Statement:

Given a CSV (comma separated values) file that contains the following information about students: ID number, name, course and year, gender, age, and general weighted average. Write a program that processes the given data into "useful" information. Processing may involve sorting, filtering, generating a summarized value, etc.

Sample Program:

In the attached zipped project folder, the data file data01.csv, the Employee reference class and the EmployeesRecords class are included. The data inside the text file are as follows: employee id, employee name, gender (M or F), age, department, years of service, salary, and employment status. You may check the content of the file at this point to verify the content of the file.

The **`EmployeesRecords`** class reads data of the input file and converts the data into a list of **`Employee`** objects (check the content of the **`Employee`** class). After the conversion, the list of objects is processed to produce the following results:

a. List of departments and corresponding number of employees in the department.
b. The five youngest (in terms of years of service) permanent employees. This list is printed such that the employees with the lowest number of years of service are shown first.
c. List of employees having less than 5 years of service. This list is printed such that the employees with the highest salary are shown first.
d. Computed total years of service within the IT Services department.
e. List of the department where in each department employees are classified according their employment status. List of employees within each employment classification are shown in alphabetical order according to their name.

Open the sample program provided and run its output.

Required topic readings:

For this activity, refer to your copy of the Java tutorials particularly on the Collections (Java Collections Framework) trail. Inherently, the use of interfaces and generics have been used in the program. Feel free to go back to these topics in case you have forgotten them. In addition, lambda expressions (and method references) have been used in the sample program.

Links to these topics are provided below for those who don't have a copy of the Java tutorials:

- Java Collections Framework: https://docs.oracle.com/javase/tutorial/collections/index.html
- Interfaces: https://docs.oracle.com/javase/tutorial/java/IandI/createinterface.html
- Generics: https://docs.oracle.com/javase/tutorial/java/generics/index.html
- Lambda Expressions: https://docs.oracle.com/javase/tutorial/java/javaOO/lambdaexpressions.html
- Method references: https://docs.oracle.com/javase/tutorial/java/javaOO/methodreferences.html

Requirement:

Create a main class called <YourName>StudentsRecords(e.g. DelaCruzJuanStudentsRecords.java) and a reference class called Student class. The main class should use the content of data02.csv (also in the given zipped project folder).

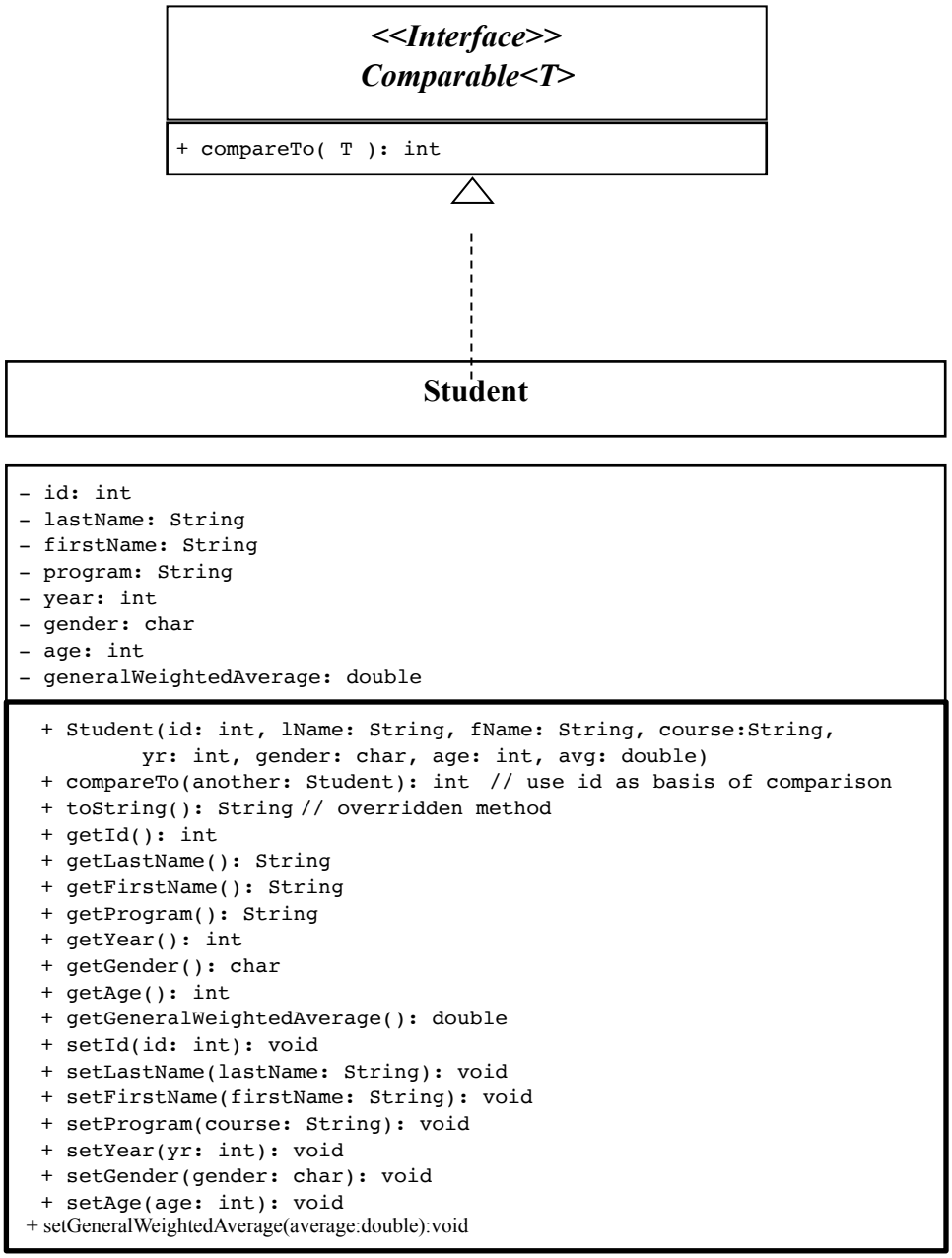
Although you need to create the source code for the Student class based on the given UML class diagram below so that you can test your main class, you should submit only your main class <YourName>StudentsRecords.java. The instructor will use his source code for the Student class. Hence, you have ensure that your program will follow the design of the student class as shown in the given UML class diagram.

The main class must contain the code that will convert the input file (**`data02.csv`**) into a list of **`Student`** objects and process the list to come up with useful information similar to the sample program. Use the **`given EmployeesRecords`** class as basis in creating your main class.

Before converting the data obtained from the input file into **`Student`** objects, you have to eliminate the double quote characters (“”) that are found in the file. Also, make sure that the data for names will have no unnecessary spaces (leading or trailing) before they are passed as values for the instance fields. These processes must be done programmatically. Meaning, you are to maintain (and NOT manipulate) the content of the file provided you.

Your program must sequentially display the following.

1. the students with ages that are less than 20 such that the list are sorted based on a descending order of the general weighted averages
2. the programs (Courses) and the total number of students for each program
3. Any relevant output that may be generated out of the data


<<Interface>>
Comparable<T>
+ compareTo(T): int
Student
- id: int
- lastName: String
- firstName: String
- program: String
- year: int
- gender: char
- age: int
- generalWeightedAverage: double
+ Student(id: int, lName: String, fName: String, course:String,
yr: int, gender: char, age: int, avg: double)
+ compareTo(another: Student): int // use id as basis of comparison
+ toString(): String // overridden method
+ getId(): int
+ getLastName(): String
+ getFirstName(): String
+ getProgram(): String
+ getYear(): int
+ getGender(): char
+ getAge(): int
+ getGeneralWeightedAverage(): double
+ setId(id: int): void
+ setLastName(lastName: String): void
+ setFirstName(firstName: String): void
+ setProgram(course: String): void
+ setYear(yr: int): void
+ setGender(gender: char): void
+ setAge(age: int): void
+ setGeneralWeightedAverage(average:double):void