****

Capital University of Science and Technology

Department of Computer Science

|  |  |
| --- | --- |
| CS3713 – Theory of Programming Languages  Assignment # 03 | |
| **Semester: Fall** 2024 | **Max Marks:** 10 |
| **Instructors:** Mamoona Bilal Date: 28 Dec 2024 – **Due Date:** 01 Jan 2025 | |

Please carefully read and follow the instructions before creating/submitting your assignment.

**Instructions:**

1. Create a program in JAVA, and upload the zip file containing following files on portal
   1. Java source code file ( .java)
   2. Screenshorts of your output in PDF file (.pdf)
2. If your submission is late or doesn’t follow these instructions, it will affect your grades.
3. Due Time: 10:00 PM

Problem:

The final exam at CUST university is approaching. The registrar’s office wants to prepare the grade reports as soon as the students’ grades are recorded. However, some of the students enrolled have not yet paid their tuition fee.

1. If a student has paid the tuition, the grades are shown on the grade report together with the grade point average (GPA).
2. If a student has not paid the tuition, the grades are not printed. For these students, the grade report contains a message indicating that the grades have been held for nonpayment of the tuition. The grade report also shows the billing amount.

The registrar’s office and the business office want your help in writing a program that can analyze the student’s data and print the appropriate grade reports. The data is stored in a file in the following format:

3 345

Aqsa 890238 Y 4

Mathematics MTH345 4 A

Physics PHY357 3 B

ComputerSci CSC478 3 B

History HIS356 3 A

.

.

.

The first line indicates the number of students enrolled and the tuition rate per credit hour. The students’ data is given thereafter.

The first line indicates that the input file contains three students’ data, and the tuition rate is $345 per credit hour. Next, the course data for student “Aqsa**”** given: Aqsa’s ID is 890238, then Y represents, she had paid the tuition, and she is taking four courses.

The course number for the mathematics class she is taking is MTH345, the course has four credit hours, her final grade is A, and so on.

The desired output for each student should be display like following. Correct GPA should be calculated from grade points.

Student Name: Aqsa

Student ID: 890238

Number of courses enrolled: 4

Course Number Course Name Credits Grade

------------ ----------------- ---------- ----------

CSC478 ComputerSci 3 B

HIS356 History 3 A

MTH345 Mathematics 4 A

PHY357 Physics 3 B

Total number of credits: 13

Semester GPA: 3.54

It is clear from this output that the courses must be ordered according to the course number. To

calculate the GPA, we assume that the

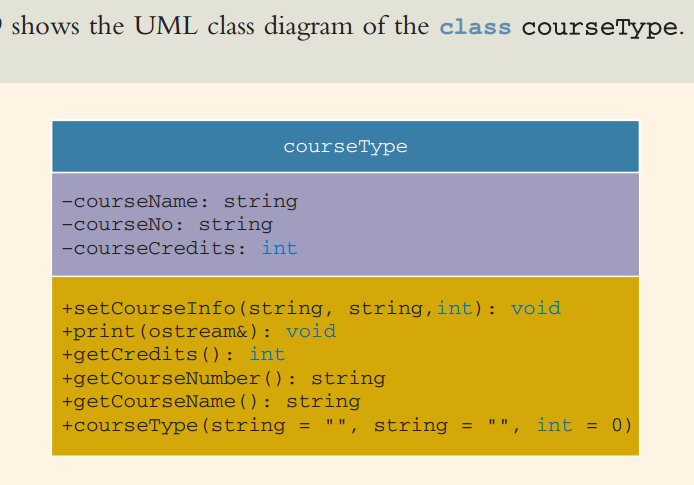
* Grade A is equivalent to four points
* Grade B is equivalent to three points
* Grade C is equivalent to two points
* Grade D is equivalent to one point
* Grade F is equivalent to zero points.

We must first identify the main components of the program. The university has students, and every student takes courses. Thus, the two main components are the student and the course.

Let us first describe the course component.

The main characteristics of a course are the course name, course number, and number of credit hours. some of the basic operations that need to be performed on an object of the course type are:

1. Set the course information.
2. Print the course information.
3. Show the credit hours.
4. Show the course code.



Next, functions to implement the operations of the Class courseType. These definitions are quite straightforward and easy to follow. The function setCourseInfo sets the values of the private member variables according to the values of the parameters. Its definition is:

The function print() prints the course information. It should

1. Print the course number.
2. Print the course name.
3. Print the credit hours.

Also, we print the course name and course number left-justified rather than right-justified (the default).

The constructor is also declared with the default values. If no values are specified when a

CourseType object is declared, the constructor uses the default to initialize the object. Using the default values, the object’s member variables are initialized as follows:

courseNo to blank, courseName to blank, and courseCredits to 0.

Otherwise, the values specified in the object declaration are used to initialize the object.

Also implement getter methods like

getCredits()

getCourseNumber()

getCourseName()

Next, we will discuss the student component.

The main characteristics of a student are the student name, student ID, number of courses in which enrolled, courses in which enrolled, and grade for each course. Because every student has to pay tuition, we also include a member to indicate whether the student has paid the tuition.

Every student is a person, and every student takes courses. We have already designed a class personType to process a person’s first and last name. We have also designed a class to process the information for a course. Thus, we see that we can derive the class studentType to keep track of a student’s information from the class personType, and one member of this class is of type courseType. We can add more members as needed.

The basic operations to be performed on an object of type studentType are as

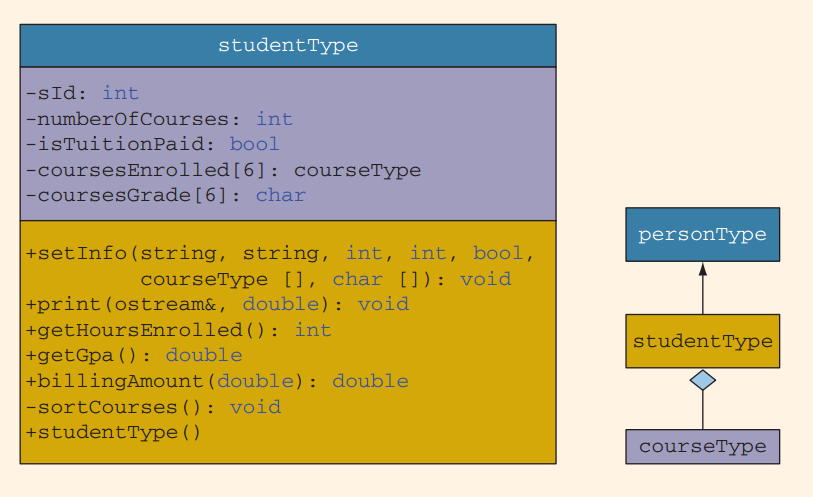
follows:

1. Set the student information.
2. Print the student information.
3. Calculate the number of credit hours taken.
4. Calculate the GPA.
5. Calculate the billing amount.
6. Because the grade report will print the courses in ascending order, sort the courses according to the course number.

We assume that a student takes no more than six courses per semester.

Following figure shows the UML class diagram of the class studentType together

with the inheritance and relation.



Note that the member function sortCourses to sort the array coursesEnrolled is a private member of the class studentType. This is due to the fact that this function is needed for internal data manipulation, and the user of the class does not need to access this member.

The function setInfo first initializes the private member variables according to the incoming parameters. This function then calls the function sortCourses to sort the Array coursesEnrolled by course number. The class studentType is derived from the class personType, and the variables to store the first and last name are private member variables of that class. Therefore, we call the member function setNameof the class personType and pass the appropriate variables to set the first and last names.

The default constructor initializes the private member variables to the default values. Note that because the private member variable coursesEnrolled is of type courseType and is an array, the default constructor of the class courseType executes automatically, and the entire array is initialized.

The function print prints the grade report. The parameter outF specifies the output device. If the student has paid his or her tuition, the grades and the GPA are shown. Otherwise, three stars are printed in place of each grade, the GPA is not shown, a message indicates that the grades are being held for nonpayment of the tuition, and the amount due is shown. This function has the following steps:

1. Output the student’s name.
2. Output the student’s ID.
3. Output the number of courses in which the student is enrolled.
4. Output the heading:

**Course No Course Name Credits Grade**

1. Print each course’s information. For each course, print:

a. **Course No, Course Name, Credits**

b. if isTuitionPaid is true,

Output the grade

else

Output three stars.

1. Print the total credit hours.
2. To output the GPA and billing amount in a fixed decimal format with the decimal point and trailing zeros, set the necessary flag. Also, set the precision to two decimal places.
3. If isTuitionPaid is true

Output the GPA

else

Output the billing amount and a message about withholding the grades.

The function getHoursEnrolled calculates and returns the total credit hours that a student is taking. These credit hours are needed to calculate both the GPA and the billing amount. The total credit hours are calculated by adding the credit hours of each course in which the student is enrolled. Because the credit hours for a course are in the private member variable of an object of type courseType, we use the member function getCredits of the class courseType to retrieve the credit hours.

If a student has not paid the tuition, the function billing Amount calculates and returns the amount due, based on the number of credit hours enrolled.

We now discuss the function getGpa. This function calculates a student’s GPA. To find the GPA, we find the equivalent points for each grade, add the points, and then divide the sum by the total credit hours the student is taking.

The function sortCourses sorts the array coursesEnrolled by course number. To sort the array, we use a selection sort algorithm. Because we will compare the course numbers, which are strings and private member variables of the class courseType, we first retrieve and store the course numbers in local variables.

Than we have to designed the classes courseType and studentType, we will use these classes to complete the program. We will restrict our program to process a maximum of 10 students.

Because the print function of the class does the necessary computations to print the final grade report, the main program has very little work to do. In fact, all that is left for the main program is to declare the objects to hold the students’ data, load the data into these objects, and then print the grade reports. Because the input is in a file and the output will be sent to a file, we declare stream variables to access the input and output files. Essentially, the main algorithm for the program is:

1. Declare the variables.
2. Open the input file.
3. If the input file does not exist, exit the program.
4. Open the output file.
5. Get the number of students registered and the tuition rate.
6. Load the students’ data.
7. Print the grade reports.

**MAIN: VARIABLES:**

This program processes a maximum of 10 students. Therefore, we must declare an array of 10 components of type studentType to hold the students’ data. We also need to store the number of students registered and the tuition rate. Because the data will be read from a file and because the output is sent to a file, we need two stream variables to access the input and output files. Thus, we need the following variables:

//array to store the students'data

studentType studentList[MAX\_NO\_OF\_STUDENTS];

intnoOfStudents; //variable to store the number of students

doubletuitionRate; //variable to store the tuition rate

Function getStudentData**()**

This function has two parameters, a parameter to access the array studentList, and a parameter to know the number of students registered.

For each student in the university,

1. Get the first name, last name, student ID, andisPaid.

2. if isPaid is ‘Y’

set isTuitionPaidto true

else

set is TuitionPaid to false

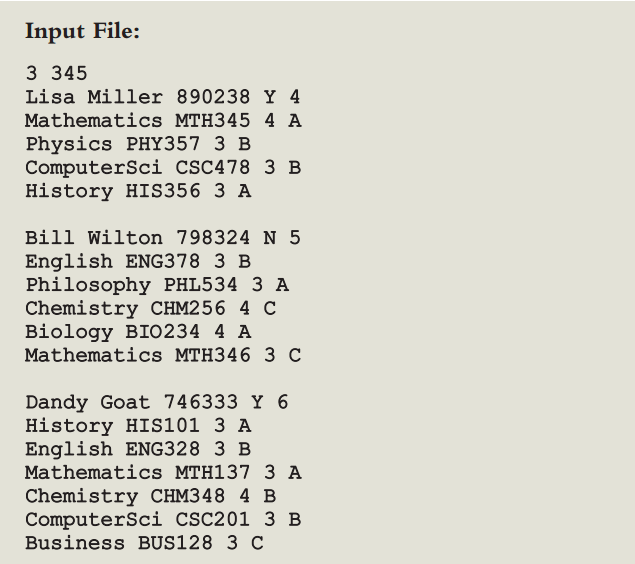
3. Get the number of courses the student is taking.

4. For each course: Get the course name, course number, credit hours, and grade. Load the course information into a courseType object.

5. Load the data into a studentType object.

We need to declare several local variables to read and store the data. This function prints the grade reports. For each student, it calls the function print of the class studentTypeto print the grade report.

Please check the sample input file in next page.



Sample Output for above file should be