

# Programming Assignment 1/2

**Scenario:**

You have been hired by a university to create and manage their course registration portal. The department of Information Science and Technology of the university is offering three 100-level and two 200-level courses in the Spring 2022. Your first task is to develop a program that will create and track all these courses in the portal. Each course has the following properties: a number, name, credit hour, and current enrollment.

Each course must have at least a course number. The maximum enrollment for each course is 30 students. The current enrollment should be no greater than the maximum enrollment. A course can have a maximum of 4 credit hour. The tuition fee for the undergraduate student is $670.00 per credit hour.

Your object-oriented application should use a data-definition class to model each course with an implementation class requesting the properties of the course. The data definition class should track course number, name, credit hour, current enrollment, seats available and compute the total cost for the course. To calculate the cost, multiply the course credit hour by the per credit tuition fee. Accessors and Mutators must be created for all the instance variables.

With a completed data definition class, your implementation class must instantiate one object from the data definition class. Your solution will gather course number, name, credit hour, and current enrollment from the user. The application should print out a report containing the course id, name, credit hour, current enrollment, seats available, and total cost. Then the application should continuously prompt user for creating another course util all the five courses have been entered. Once all the courses are created, the application should print out a weIl-formatted message stating the total number of courses, total enrollment and average enrollment. It should also display the course with the largest enrollment.

**Other Requirements:**

* Your solution must use object-oriented techniques (No points earned for a procedural solution).
* Your solution must be designed as a modular solution using methods other than main, with each method performing one task. The main method should contain only a very minimal amount of code, calling methods instead.
* Your solution must include appropriate constants, constructors, accessors, mutators, and special purpose methods with exception handling, as necessary in support of the problem.
* Your solution may not use any functions or language constructs not covered during IT 106 or this semester’s IT 206 without prior authorization from your instructor, even if you know other functions or language constructs. We want everyone to be on the same "playing field", regardless of previous programming exposure, and get practice with algorithmic design to solve problems (the intent of the course). Using something existing not discussed in class does not give you as much practice as solving the problem yourself. **Doing this may lead to a substantial grade penalty, a grade of zero, or an Honor Code inquiry.** When in doubt, ask!

**Hints:**

* You do not need to use arrays to solve this problem. Keep in mind once you have finished entering the information for one course, it is not necessary to keep the information.
* Think about what type of validations might be appropriate for this application. Make sure these are all handled.
* Think about any special purpose methods that might be needed in the data definition class. Special purpose methods are important when using values in instance variables to perform a specific process.

**To Do (Check Blackboard for Due Dates):**

**Programming Assignment 1: Solution Design**

1. List and describe the purpose of each class that will be needed to solve this problem
   1. You must separately identify (list) and describe the purpose of each class. One or two sentences per class should be sufficient. You do NOT need to include the CircleDrawer class, since this class was provided for you.
2. *Data Definition Class(es)* - Create a detailed UML Class Diagram, listing and explaining all class variables, accessors, mutators, special purpose methods, and constructors associated to each data definition class.
3. *Implementation Class* - Create a table that lists all methods that will be used to create the implementation class. For each method identified, provide the following:
   1. A 1-2 sentence describing the purpose of the method
   2. A list of the names, data types, and brief description (1-2 sentences) for each input variable into the method, if there are any
   3. The name and data type of the variable to be returned from the method, or void if nothing will be returned

**Note:** You do not need to provide any pseudocode. However, you should create a plan for yourself as to how you will perform the logic for each method. If you do not do this, you will have great difficulty in completing the solution implementation.

An example format to use for this table is as follows. You can format this table in any way you’d like, so long as the information is clearly presented. Completing this table correctly will help you build your documentation and code for your solution implementation.

**Method:** getMagazine

**Purpose:** The method will allow for the creation and population of a Magazine object based on user input of a title, cost, and number in stock

**Inputs:** none

**Return:** magazine : Magazine – The Magazine object created and populated from user input

**Method:** checkout

**Purpose:** The method will allow for the purchase of a magazine. It takes into account the sales tax that will be charged as part of the cost calculation

**Inputs:** magazine : Magazine – The magazine to be purchased

salesTax : double – The sales tax percentage to be charged as part of the checkout

**Return:** void

Upload a Word document containing only items above to Blackboard.

|  |  |
| --- | --- |
| **Grading Criteria** | |
| **Requirement** | **Points** |
| List and describe the class(es) needed to solve the problem | 10 |
| Data Definition Class Design – Detailed UML Diagram | 40 |
| Implementation Class Design – Table Listing All Methods | 50 |

**Programming Assignment 2: Solution Implementation**

Write a well-documented, efficient Java program that implements the solution design you identified. Include appropriate documentation as identified in the documentation expectations document.

To Blackboard, **submit ONLY ONE .zip file** containing all of the .java files part of your submission for your solution implementation. As you are naming your Java files, make sure they are indicative to the purpose of the file. **Do not include your name in the file name or use generic names, such as Assignment2.java**. Your .zip file should contain only the .java files in your solution. Be careful that you do not submit .class files instead of .java files.

**Warning!** You must submit **ONLY** **ONE** .zip file containing **ONLY** your .java files. Failure to follow this instruction precisely will result in a 10 point deduction of the assignment score. **No exceptions!  
*Why is this important?* The goal is to teach you how to properly package your IT solutions into a “customer-friendly” format while paying attention to “customer” requirements provided to you.**

**Your program must compile using jGrasp**. Any final program that does not compile, for any reason, will receive an automatic zero. Other IDEs often place in additional code that you are unaware of, doing too much of the work for you. **You are strongly discouraged from using IDEs other than jGrasp.**

|  |  |
| --- | --- |
| **Grading Criteria** | |
| **Requirement** | **Points** |
| Implementation of object-oriented Java program, using efficient practices, such as the use of constants, good variable names, information hiding, no redundant code, etc. | 80 |
| Appropriate objective-style documentation | 10 |
| Appropriate intermediate comments | 10 |