10.09.2022

48024 Applications Programming Assignment 1

Topics:

OO Design, Standard Patterns, Lists

Learning Outcomes:

This assessment task addresses the following subject learning objectives (SLOs): 1, 2 and 3

Due date:

11:59PM Tuesday 20 September 11:59PM Friday 30 September

Weight:

35%

Individual Work

All work is individual. You may discuss ideas, approaches and problems, but you should write every line of code yourself except for code copied from the lecture notes, lecture code or lab code. You **MUST NOT** let another student see your solution code, and you **MUST NOT** look at another student's solution code. More information about Academic Misconduct can be found at:

http://www.gsu.uts.edu.au/rules/student/section-16.html

Working Language

You can choose either Java or Python to complete assignment 1. The higher mark between your Java solution and Python solution will be counted into your final grade. However, you are only credited with one of your solutions, either Java or Python, not both of them or the mixture.

The specification is illustrated based on Java. You can simply translate the Java syntax to Python for your Python solution. Detailed explanations about Python criteria will be posted on the FAQ page on ED.

Specification

The Supermarket Membership Management Department will consist of two main components for Membership Management System (MMS), an administrative component, and an MMS component.

The administrative section will allow, through text-based menus, the login and logout of administrators, the addition and removal of members in the system, the display of membership records, the display of a certain member with name, and the update of member's accounts.

The MMS component will store a list of all the membership information of the members and a list of the summary of the membership. The MMS component will also allow the creation and display of the collective tuition report as MMS record log. Finally, the MMS component will also handle the display of a certain member information.

Each membership record will include the member's ID, name, email, phone, address, type, credits, deduction rate, gas deduction rate, dollar available, pay per credits and supermarket fields.

Each supermarket record will include the membership's name, credit, gas deduction rate, dollar available, deduction rate, pay per credit, and type.

The User Interface and sample I/O will be posted on the Assignment page on ED for a detailed description.

An aside

While reading the first part of the specification, you will notice there is a lot going on.

- How many functions did you identify?

- How many classes did you identify?
- What are the fields in each class?
- How many goals did you identify?
- How many patterns did you think of that might be applicable?

This assignment will be challenging, and you will probably want to manage your time well.

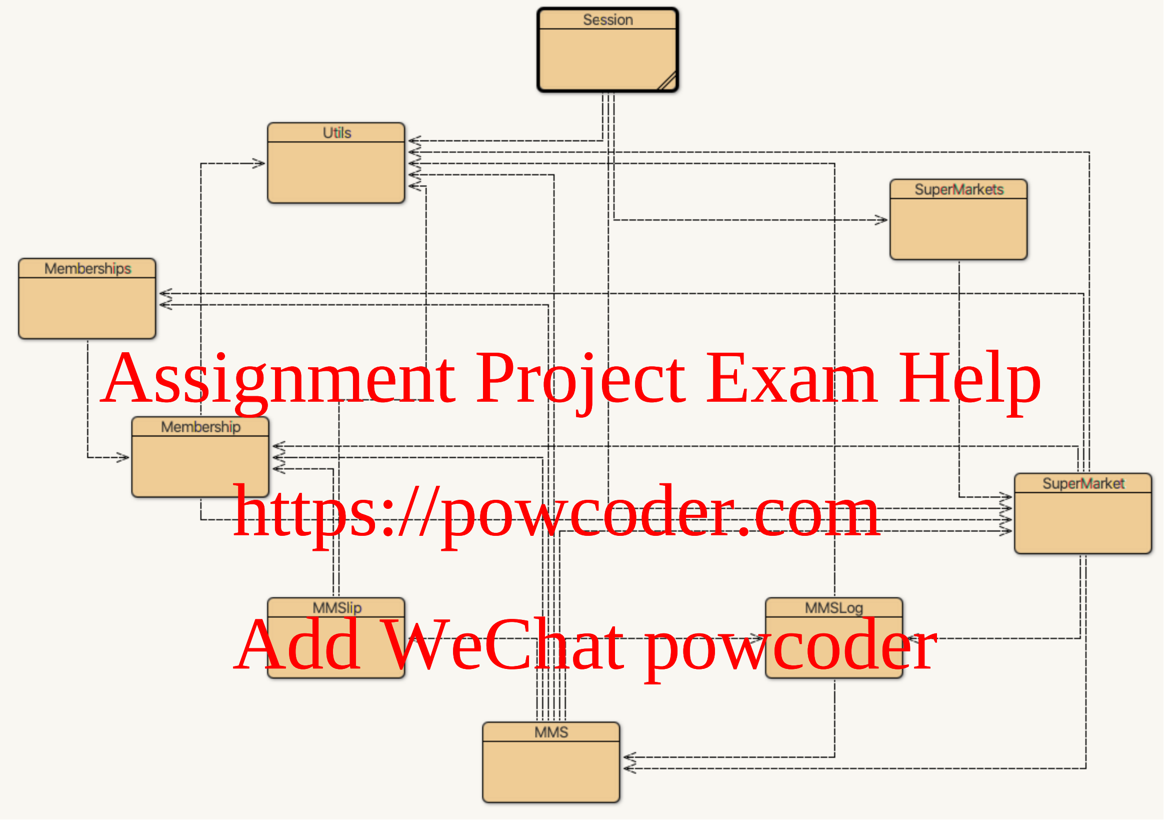
- How long do you think it will take you to code the functions?
- How long do you think it will take you to code each goal?
- A good rule of thumb is to think of an estimate, and then multiply that number by 3 or 4!
- To manage your time well, you may need to figure out which parts of the assignment you can start early.
- Which parts can you start now?
- Which parts can you start in week 6?

If you complete parts in the same week that you learn the topics (while they are fresh in your mind), they will take less time to complete.

Requirements

- Your design will consist of exactly the following classes with the listed fields, declared as indicated. You may not add or remove classes or fields; however, you may add constructors, functions and procedures to complete your design (in fact, you will have to!). You should pay careful attention to the tests on ED, as these will help guide you with some (but not all) of these methods.

- To help visualize the design, a partial class diagram has been provided



- Classes – your design will consist of these 8 classes:

  1. Session
  2. Supermarket
  3. Supermarkets
  4. Membership
  5. Memberships
  6. Slip
  7. MMS
  8. MMSLog
  9. Utils (this is the class to facilitate format and simple I/O, do not change)

- Fields – All the fields have been clarified in each class and they should not be modified. The fields also have some additional requirements and structures:

  Lists all have the abstract type of List<>, but must be instantiated with a concrete type that implements the List<> behavior (you can choose either – you may also want to think about why you might do things this way).

- Constructors – the constructors of the class have the following requirements:

1. The main method of the program will be in the Session class.

2. All constructors initialize the fields of their classes.

3. The Session, Supermarkets, Memberships, and MMSLog constructors take no parameters.

4. The Supermarket constructor takes three parameters, the name, email, and password, corresponding to the three fields identically named. Other fields require no parameters but the constructor methods.

5. The slip constructor takes membership as the parameter to initialize the fields identically named.

6. The MMS constructor takes supermarket and memberships as the parameters.

7. The Membership constructor requires the following formula to set the value of the fields:

   Type: set by expense

   payPerCredit: set by Type

   totalCredit: payPerCredit * expense

   deductionRate: set by Type

   GasdeductionRate: set by Type

   DollarAvailable: totalCredit / 200

expense	type	payPerCredit	deductionRate	GasdeductionRate
expense <500	Bronze	20	0.05	0.1
500 <= expense < 1500	Silver	10	0.1	0.15
1500 <= expense < 3000	Gold	8	0.15	0.2
3000 <= expense < 5000	Diamond	6	0.2	0.25
expense >= 5000	Platinum	4	0.25	0.3

*When updating the expense for a member, the input value will be added to the existing expense of the member. When the value is positive, the expense is accumulated to the membership. Otherwise, the expense is mitigated to the membership.

8. Supermarkets has the following data for Login:

John Smith	john.smith@uts.com	user222
Jane Tyler	jane.tyler@uts.com	super123

9. Memberships has the following data.

name	email	phone	address	ID	expense
Thomas Muller	thomas.muller@uts.com	99991111	3 Byern St. Sydney 2001	13697480	2134.5
Alice Stefan	alice.stefan@uts.com	88881111	24 Pitt St. Sydney	14517880	4512.2

			2001		
Lucy Lu	lucy.lu@uts.com	98981100	11 Hunter St. Sydney 2100	13267102	158.4
Andreas Brehme	andreas.b@uts.com	90001222	91 Sussex St. Sydney 2100	13678020	7596.3
Ruddy Voller	ruddy.v@uts.com	98980000	15 Stan St. Sydney 2100	13972870	1100.0
Monica Shwarz	monica.s@uts.com	92241188	155 Jones St. Sydney 2001	13859610	6741.2

- Utils Class – the Class defines the following formatting:

1. Membership Format:

   `MembershipHeader( )` defines the output format of the header:

```
+--------------------+-------------------------+----------+----------+
| Membership Name    | Email                   | Phone    | Type     |
+--------------------+-------------------------+----------+----------+
```

   `System.out.format(Utils.membershipFormat,name,email,phone,type)` will produce a string of the form:

```
name,email,phone,type
```

e.g.

```
| Alice Stefan       | alice.stefan@uts.com    | 88881111 | Bronze |
```

Note: "| %-20s | %-25s | %-10s | %-10s |%n" defines the space for the string.

2. MMS Format:

   `slipHeader( )` defines the output format of the MMS header:

```
+----------------+--------------+--------------+--------------+---------+
| Membership Name | Expense      | Credit       | Dollars      | Type    |
+----------------+--------------+--------------+--------------+---------+
```

   `System.out.format(Utils.mmsFormat,name,expense,totalCredits,DollarAvailable,type)` will produce a string of the form:

```
name,expense,credit,dollars,type
```

e.g

```
| Lucy Lu            | 158.40           | 3168.00          | 15
| Bronze    |
```

Note: " | %-18s | %-12.2f | %-18.2f | %-16d | %-14.2f | %-14d | %-10s |%n " defines the space for the string.

3. Sum Format:

   `System.out.format(Utils.sumFormat,"title",value)` will produce a string of the form:

   Title, value

e.g.

```
| Total credits               |  122667.20  |
```

Note: " | %-18s | %-9.2f |%n " defines the space for the string.

4. Log Format:

`logHeader( )` defines the output format of the MMS header:

```
+--------------+----------------+
|  MMS Record  |     RecordID   |
+--------------+----------------+
```

`System.out.format(Utils.logFormat,MMSRecord,name)` will produce a string of the form:

```
MMSRecord,name
```

e.g.

```
| MMS 1           |   Jane Tyler1    |
```

Note: "| %-12s | %-9s |%n" defines the space for the string.

Advanced Requirements

To achieve a mark of >84, you must implement the record archive function and retrieve function (or set of functions). This corresponds to 2 options in the MMS menu. Choosing archive option should create a mmsLog named after the supermarket and the version number. If the new archive is the same as the previous archive, the previous archive will be overwritten with the new version number. In this way, the system avoids archiving the same record with different names. Choosing retrieve option should prompt the user for a valid record name and display the mmsLog matching the name. If the system does not archive the report with the name, system will show “No MMS is recorded as: <name>”. Detailed specification please refers to the User Interface and sample I/O page on ED.

Expected Workload

The time to do the assignment to a credit/distinction level has been estimated at 35 hours for a student of average ability who has completed all the tutorial and lab exercises.

Online Support

A FAQs (Frequently Asked Questions) page will be posted in the Assignment page on ED. If you have a question, check the FAQ page first, it may already be answered there. You should read the FAQ at least once before you hand in your solution, but to be safe check it every couple of days. Anything posted on the FAQ is considered to be part of the assignment specification. The FAQ will be frozen (no new entries) two days before the due date; no questions will be answered after it is frozen.

If anything about the specification is unclear or inconsistent, check the FAQ first, then contact the subject coordinator who will try to make it clearer by replying to you directly and posting the common questions and answers to the FAQ. This is similar to working on the job, where you ask your client if you are unsure what has to be done, but then you write all the code to do the task.

The preferred way to ask assignment related questions is to participate in the lectures, lab activities, UPASS, and consultation sessions. Students are expected to seek help through the following steps:

Step 1: Check the FAQ
Step 2: Ask peers or via the group discussion board
Step 3: Ask your tutor
Step 4: Forward to the subject coordinator

The Subject Coordinator may be contacted by email if you have matters of a personal nature to discuss, e.g., illness, personal issues or other matters of importance. All emails sent to subject coordinators, tutors or lecturers must have a clear subject line that states the subject number followed by the subject of the email [e.g. Subject 48024, Appointment Request], and must be sent from your UTS email address.

Requests for appointments outside the given consultation hours may be arranged where circumstances require, and to do so please contact the subject coordinator by email.

ED Marking

Your solution is marked for correctness by comparing the output of your system to the output of the benchmark system in ED. You can submit a solution to ED many times by press "MARK"; I urge you to do this, so you receive credit for your work. Any code hasn't been "MARK" by ED won't be credited.

ED will test the features of your program in a certain order, but it cannot test the more advanced goals until the basic goals are working. To receive marks, you must pass ED's test cases in the order in which ED tests them.

Your code is marked by software, so there are chances to get a good mark by fooling or spoofing the software. If you spoof a task worth N marks, you receive a penalty of 2*N marks. Design rules and spoofy check will be manually performed after due. Detailed specification about design rules and spoofy check refers to the document "Spoofy Check".

Submission and Return

Your provisional mark and feedback is generated immediately each time you submit to ED. However, it takes time for the analysis of spoofing, plagiarism, collusion and general cheating, which will start two weeks following the due date. If you are suspected of Academic Misconduct, I will forward your case to the Misconduct Committee and will notify you by manual feedback in ED.

The due date won't be extended unless under extenuating circumstances. Submission extension within one week will be granted automatically with a late penalty. An extension CANNOT be given after the due date. For any extension beyond one week or extension without penalty, you will need to submit a Special Consideration following the Special Consideration process for special consideration for reasons including unexpected health, family or work problems.

Marking Scheme

The marks for the assignment are divided into the following functionality components (note that individual tests may test several functionality components, and a functionality component may be tested by several tests):

Functionality Component	Mark Allocation
Main Menu	4
Supermarket Menu	5

MMS Menu	5
View Membership	10
Add Membership	10
Retrieve Membership	10
Update Membership	10
Delete Membership	10
Slip Report	10
MMS Report	10
Archive Report	6
Retrieve Report	10

This adds to a mark out of 100, at makes up 35% of your final assessment mark.