CS251: Intro to Software Engineering

Homework 2 (4 marks)

Due Date 26 April 2021 @ 11:00 pm (submit on Blackboard)

Objectives

This assignments aims to:

1- Review basics of OOP
2- Train students on Java and its IDEs like Eclipse, IntelliJ, JCreator and NetBeans.
3- Train students on basic software engineering practices (code style and code commenting).

Setup

1- This assignment will be solved in **groups of 3 students from the same lab**.
2- Each team member will pick one problem from task 1 and develop a solution for it. And there is a group task for the **entire team** to do together.
3- The group will submit together **one combined** solution.
4- The entire group is responsible of helping any weak member to be able to do his/her task **by his/herself**, by providing the necessary support, knowledge, hands-on demos, etc.
5- TA can ask any group member about the work on any other group member.
6- **Only submit original work. Any copied work will be severely penalized.**
7- مسؤولية الفريق تضامنية عن عمله و أى غش من أى فرد سيكون مسؤولية الجميع و يخصم منهم ٣ أضعاف الدرجة
8- **It is very important you do Lab 1 to master Java OOP concepts.**
9- **Please read the marking criterion very carefully to understand how you will be marked.**

Task 1 (4%)

In this task, each team member will pick one of the following problems and solve it. **Team should ensure that they agree on the public methods of each class.** So, when they integrate their work (put the classes together), it works well. All team members must:

- **Apply OOP principles** properly.
- **Use the coding style uploaded with the assignment**

Problem 1

We are going to implement a banking system. It's very important to maintain data of each account in the bank. So, we need to create an **`Account`** class. We will also create a **`SpecialAccount`** class that inherits from **`Account`** class and has a special feature.

- Create **`Account`** class.
- Create **`balance`** and **`account number`** attributes in account class (private not public)
- Create a suitable constructor that sets the **`account number`** and the initial **`balance`** value.
- Create setter and getter for each attribute.
- Override the method **`toString ( )`** inherited from class Object to make it return a meaningful string representation of the account information.

CS251: Intro to Software Engineering

Homework 2 (4 marks)

Due Date 26 April 2021 @ 11:00 pm (submit on Blackboard)

- Create methods **`withdraw`** and **`deposit`** to be able to take or put money. To withdraw, enough balance should be available.
- Create another class **`SpecialAccount`** that inherits from class **`Account`**.
- Use polymorphism to override method **`withdraw`** to allow over drafting with maximum limit of 1000 LE. اوفر درافت معناها السحب على المكشوف يعنى ممكن يسحب فلوس من الحساب حتى لو رصيده صفر بحد أقصى ١٠٠٠ جنيه يعنى ممكن يصل رصيد الحساب إلى -١٠٠٠ جنيه
- Write a main class to test class **`Account`**. It should create two accounts of two types and test all the functions you created in them.

Problem 2

We are going to implement a bank system. It is required to implement a **`Client`** class to represent the bank's clients. So, you are required to do the following:

- Create **`Client`** class.
- Add **`name`**, **`nationalID`**, **`address`**, **`phone`** and **`account`** private attributes for the class.
- Create a suitable constructor that sets these attributes. (Note that before creating a new **`Client`**, you will need to create an object of type **`Account`** and pass it as a parameter to the client object's constructor to be his account).
- Create setter and getter for each attribute.
- Override the method **`toString ( )`** inherited from class Object to make it return a meaningful string representation of the client information. String representing Client information, should also include his account's information.
- Create another class **`CommercialClient`** that inherits from class **`Client`**. A commercial client is a company not a person. It has an extra attribute **`commercialID`** and setter and getter for it. Its **`NationalID`** is set to 00000000000000.
- Use polymorphism to override method **`toString`** print the commercial client details including his commercial ID instead of the national ID.
- Write a main class to create two clients of two types and test all the functions you created in them.

Problem 3

Now you have Account and Client classes and their subclasses (assume they exist until your colleagues finish doing the real ones). Let's make bank class

- Create **`Bank`** class.
- Create **`name`**, **`address`** and **`phone`** attributes in **`Bank`** class.
- Create a constructor and suitable setters and getters.
- Create an **`ArrayList`** of accounts in **`Bank`** class, this array maintains all accounts' data.
- Create an **`ArrayList`** of clients in **`Bank`** class, this array maintains all clients' data.

CS251: Intro to Software Engineering

Homework 2 (4 marks)

Due Date 26 April 2021 @ 11:00 pm (submit on Blackboard)

- Create methods to (1) add a new client and his account (can be special client or commercial account also) and (2) display existing clients and their accounts. Each time a new client and account are created, they are added to the two **`ArrayLists`** of accounts and clients.
- **Integrate your Bank class with `Client` and `Account` classes and use them in it.**

Group Task

Your team is required to:

1- Integrate the classes they created for Task 1 together to make an integrated working application.
2- Create a Main class that includes the main method.
3- This class will maintain a bank and allow the user to deal with it.
4- It will include a method to display a menu of options that allows the user to choose his options.
5- Write Java Doc documentation for the code and generate html pages.
6- Each team member will **explain in great details all the work** he did for the assignment including his code and documentation. He will explain the code and all related OOP and Java concepts and syntax he used. His colleagues should fully understand the code and be able to change or modify it if needed. They should be able to explain it to the TA.

Deliver and Assessment

1- **Submission is only accepted through Blackboard**. All members should upload the same file.
2- Students should add header to their file with author name and date.
3- Add the programs of each task together in a folder for this task.
4- Then they should provide one ZIP file named CS251_HW2_ID1_ID…..zip
5- Each student **must** submit the form in the previous page, filled and completed. **No discussion if you do not bring your form with you.**
6- Marking will be as follows:

1.5 mark	For implementing your class and it works properly
1.5 mark	For an integrated working application and it works well
1 mark	For good use of OOP, Javadoc comments, following coding style
Cheating	
-9 marks	**FOR ALL TEAM members if any piece of code is similar to any other source**.

Policy Regarding Plagiarism:

١. تشجع الكلية على مناقشة الأفكار و تبادل المعلومات و مناقشات الطلاب حيث يعتبر هذا جوهريا لعملية تعليمية سليمة
٢. ساعد زملاءك على قدر ما تستطيع و حل لهم مشاكلهم فى الكود و لكن تبادل الحلول غير مقبول و يعتبر غشا.
٣. أى حل يتشابه مع أى حل آخر بدرجة تقطع بأنهما منقولان من نفس المصدر سيعتبر أن صاحبيهما قد قاما بالغش.
٤. قد توجد على النت برامج مشابهة لما نكتبه هنا أى نسخ من على النت يعتبر غشا يحاسب عليه صاحبه.
٥. إذا لم تكن متأكدا أن فعلا ما يعد غشا فلتسأل المعيد أو أستاذ المادة.
٦. فى حالة ثبوت الغش سيأخذ الطالب سالب درجة المسألة ، و فى حالة تكرار الغش سيرسب الطالب فى المقرر.

CS251: Intro to Software Engineering

Homework 2 (4 marks)

Due Date 26 April 2021 @ 11:00 pm (submit on Blackboard)

TA Name: …… Mark: ……..

CS251 – Introduction to Software Engineering, 2021

Each student fills this form for his program and gives it to TA

The questions to answer about each program are included in the following form. **Print and fill this form and bring with you to the discussion.**

Student name: …… **ID:** …… **Group:** ……

Which Program (Idea for task 2) did you choose? ……

Which of the following Java / OOP features did you use in your program?

1. How many classes did you create and their names? ……
2. How many different access specifiers did you use and their names? ……
3. How many Java coding style rules did you use and which ones? ……
   ……
4. How many Javadoc tags did you use and which ones? ……
   ……
5. Did you use inheritance? When and why? ……
   ……
6. Did you use method overriding? When and why? ……
   ……
7. Did you use method composition? When and why?
   ……
   ……
8. Did you use method polymorphism? When and why? ……
   ……

Draw in the space below a simple UML class diagram that shows your main classes, their attributes and operations and their interactions with each other.