COMP503/ENSE502/ENSE602 _ Assignment 2
Sem 2 2020

Develop your own unique OOP Application

Due date: Friday, 16th October 2020 at 5:00 pm

This assignment has 100 marks and is worth 20% of your final grade.

Brief

Individually develop your own unique OOP application which solves a meaningful problem. It is up to you to decide on the purpose of your program.
Be creative! Be inspired by technology you use every day, for example:

- Film streaming service that keeps track of content, users and what they view
- Location tracking that registers users and allows them to store location check ins
- Online shop that keeps track of customers and allows product purchases

Think of the functionality of these kinds of applications: adding and removing objects from collections, searching and counting.

For full marks, your application must feature:

- All Functionalities implemented by appropriate class design and object instantiation
- Interaction with User (Console input/output)
- Interaction with Files (Stream input/output)
- Usage of **Java Collection** framework (e.g., ArrayList,Queues, etc) + algorithms (e.g., sorting, searching, etc.)
- Excellent documentation, including Javadoc and a Development Log
- Advanced Object-oriented techniques
- **See marking scheme for complete requirements**

Development Log

In an electronic document, log work completed in design, implementation and testing phases. Each time you work on your assignment create a new entry detailing:

- When you did the work
- How long it took
- What you did
- Any problems you had.

Design Phase

Design the organization of classes, data and methods needed for your application and their relationships Draw your application's UML class diagram using Microsoft Visio or similar tool.

COMP503/ENSE502/ENSE602 _ Assignment 2
Sem 2 2020

Implementation Phase

Once you have completed the deisgn phase, begin programming. Apply good development and documentation standards. Document difficulties in the implementation phase in your development log. Did unforeseen problems in implementation cause you to change your design?! So, document this as well.

Testing Phase

During implementation, regularly test your program's functionality. Record how you identified and corrected bugs in your application.

Javadoc Commenting

1. Your classes must have commenting of the form:

```
/**
 * Comment describing the class.
 * @author yourname student id
 **/
```

2. All methods must be commented with appropriate Javadocs metatags. For example:

```
/**
      * A comment to describe the method
      * @param a first parameter description
      * @param b second parameter description
      * @return a description of the returned result
      * @author student id
      * */
```

COMP503/ENSE502/ENSE602 _ Assignment 2
Sem 2 2020
Marking Scheme

Criteria:	Weight	Grade A Range: 100% ≥ x ≥ 80%	Grade B Range: 80% > x ≥ 65%	Grade C Range: 65% > x ≥ 50%	Grade D Range: 50% > x ≥ 0%
Development Log: **-Accurate** **-Descriptive** **-Honest**	**15 %**	➢ Comprehensive development log. ➢ An honest, accurate and detailed account of the development is described.	➢ A development log is present. ➢ An honest, accurate and detailed account of the development is described.	➢ A development log is present. ➢ An honest and accurate account of the development is described.	➢ Development log .pdf file is not present. ➢ The development file is too weak.
Java Programming: **-Console I/O** **-File I/O** **-Enumeration**	**10 %**	➢ Console input and output present. ➢ Multiple files input and output. ➢ Multiple fit for purpose enums present.	➢ Console input and output present. ➢ File input and output. ➢ Multiple fit for purpose enums present.	➢ Console input and output present. ➢ File input and output present. ➢ One enum present.	➢ Missing any one of the following: -Console I/O -File I/O -Enumeration
Data Storage: **-Variables** **-Arrays or Collections** **-Algorithms**	**25 %**	➢ Multiple collections used for collection-like purposes. ➢ Multiple algorithms used comprehensively on collections.	➢ Collection used for collection-like purposes. ➢ Multiple algorithms used on a collection.	➢ One collection present. ➢ One algorithm utilised.	➢ Missing any one of the following: -Variable -Array or Collection -Algorithms
OO Techniques: **-Classes** **-Relationships** **-Inheritance** **-Abstraction** **-Interfaces** **-Modularity** **-Reuse** **-Encapsulation** **-Exceptions**	**25 %**	➢ More than nine purposeful classes are present. ➢ Relationships are sensible. ➢ Inheritance hierarchy is present. ➢ Abstraction and interfaces are utilised. ➢ Code is encapsulated, modular and features reusability. ➢ Exceptions are present.	➢ More than six purposeful classes are present. ➢ Relationships are sensible. ➢ Inheritance hierarchy is present. ➢ Abstraction and interfaces are utilised. ➢ Code is encapsulated, modular and features reusability.	➢ Classes are present. ➢ Relationships are sensible. ➢ Inheritance hierarchy is present. ➢ Abstraction and interfaces are utilised. ➢ Code is encapsulated, modular and features reusability.	➢ Missing any one of the following: -Classes -Inheritance -Abstraction -Interfaces
Runtime Execution: **-Interactive** **-Meaningful** **-Non-trivial** **-Bug Free**	**15 %**	➢ Meaningful, non-trivial problem solved by the program. ➢ Program is interactive. ➢ Program is bug free.	➢ Non-trivial problem solved by the program. ➢ Program is interactive. ➢ Program contains generally bug free.	➢ Non-trivial problem solved by the program. ➢ Program is interactive. ➢ Program contains minor bugs.	➢ Any one of the following is present: -Program breaking bug. -Not interactive. -Solves a trivial problem, the program is simple.
Code Quality: **-Whitespace** **-Naming** **-Javadoc**	**10 %**	➢ Whitespace is comprehensively consistent. ➢ All naming is sensible and meaningful. ➢ Comprehensive Javadoc throughout the entire codebase.	➢ Whitespace is consistent. ➢ Majority of naming is sensible. ➢ Majority of the codebase features Javadoc.	➢ Whitespace is consistent. ➢ Some Javadoc is present.	➢ Whitespace is inconstant and hence difficult to read.

Dr. Seyedjamal Zolhavarieh, Auckland University of Technology, szolhava@aut.ac.nz

COMP503/ENSE502/ENSE602 _ Assignment 2
Sem 2 2020

Authenticity

Remember!

- It is unacceptable to hand in any code which has previously been submitted for assessment (for any paper, including Programming 2) or available online
- **All work submitted must be unique and your own**!

The following actions may be deemed to constitute a breach of the General Academic Regulations Part 7: Academic Discipline, Section 2 Dishonesty During Assessment or Course of Study

- 2.1.1 copies from, or inappropriately communicates with another person
- 2.1.3 plagiarises the work of another person without indicating that the work is not the student's own – using the full work or partial work of another person without giving due credit to the original creator of that work
- 2.1.4 Unauthorised collaboration in Assessment - collaborates with others in the preparation of material, except where this has been approved as an assessment requirement. This includes contract cheating where a student obtains services to produce or assist with an assessment
- 2.1.5 resubmits previously submitted work without prior approval of the exam board
- 2.1.6 Using any other unfair means

Submission Instructions

Submit the following documents as an archive **.zip** file of your documents on Blackboard before the deadline:

- Your full java project for assignment.
- Sample console output demonstrating your program in use (**Text** file)
- Development Log file (PDF file)
- UML diagram file (PDF or VSD file)

Zip structure and file naming requirements. Please ensure your submission matches the following:

- **lastname-firstname-studentid.zip**
  - **Project folder (can be found in your workspace)**
  - **studentid-console-sample.txt**
  - **studentid-development-log.pdf**
  - **studentid-console-uml.pdf or .vsd**

Replace the underlined text with your details.

COMP503/ENSE502/ENSE602 _ Assignment 2
Sem 2 2020

An extension will only be considered with a Special Consideration Form approved by the School Registrar. These forms are available at AUT Blackboard.

You will receive your marked assignment via Blackboard. Please look over your entire assignment to make sure that it has been marked correctly. If you have any concerns, you must raise them with the lecturer. You have **one week** to raise any concerns regarding your mark. After that time, your mark cannot be changed.

Do not go to the lecturer because you do not like your mark. Only go if you feel something has been mismarked.