UTS Insearch
UNIVERSITY OF TECHNOLOGY SYDNEY

Project B

Diploma of IT

IAPP001 - Applications Programming

SEMESTER 03, 2022

Weight: 10%

DUE : Week 10, Friday 11:59pm

ASSESSMENT TASK #3: PROJECT B

TASK DESCRIPTION

This project requires students to write an Object Oriented solution for a given application, using all the concepts taught in the subject.

You may choose to complete this assessment with at most one (1) other person. To work as a pair; you and your partner MUST be in the same tutorial class. You will need to spend approx. 6 hours together working on the assessment (outside of tutorial hours). If you do not intend to spend this time working together – choose to complete the assessment on your own. Both partners must put in an equal amount of work and effort, or the group will be dissolved by the tutor.

Your code will be assessed against the learning outcomes below using techniques taught in the subject.

LEARNING OUTCOMES

1. Demonstrate a working knowledge of lists in Java.
2. Design a good OO solution from a specification.
3. Use inheritance in Java.
4. Construct a GUI interface

BRIEF

This is a take home assessment to evaluate your understanding of object-oriented concepts taught so far in IAPP001**. All code must compile and run in BlueJ version 4.1.2 to be marked.**

You will use your Project A as the starting point for this project.

You will design a Graphical User Interface to support your application, using the domain classes from Project A.

Your GUI design should outline the main use cases for the application, the basic pane design and situations to handle for the application. You may use a main window and sub windows or a main window with a stacked tab pane.

You will design the windows and panes for your application. You will implement all of the GUI components following your design. You will implement the Observer pattern and test that your solution satisfies the requirements of your application.

The assessment is comprised of three (3) parts: the GUI design, the code and the explanation.

GUI design:

Create your design for the GUI components – think about input needed from the user. Make sure your GUI design shows the GUI layout for your application. You can use paint or word tools to create the design. Make sure each panel/pane has at least 1 button to handle events.

The code:

Copy your project and create a model package. Move all the domain classes to the model package. Add MyObserver and Updater to the model package. Delete the In class and changes all dependencies to use parameters for input. Write your GUI components. Make sure you are using the Observer pattern to refresh each window when the data changes in the models. You need to implement the 4 steps for the observer pattern – all model methods that change data need to call updateViews(). All panels have an implemented update() method etc.

The explanation: (300 words)

Write a detailed explanation of how the user would interact with your GUI to perform each of the usecase tasks. Explain how the observer pattern works by following the flow of control for one of the usecases.

This is an open book assessment. You may use lecture notes and solutions to workshop exercises – however, you cannot use solutions to assessments from a previous or current semester.

Every resource you use (other than subject materials) MUST be documented.

Any student caught using prohibited materials will receive a fail grade for this assessment.

All code submitted for assessment must be written by you (and your partner).

Any attempt to copy, move, alter or reproduce code files will be viewed as misconduct.

As an exception: you may use the Gladiator Prime solution (week 6) as a replacement of your project A if you failed and/or cannot use your own project A as a base for Project B. Please note: If you use Gladiator Prime you can only receive a maximum of a Credit for this project.

SUBMISSION

You will submit your assessment in MS Word document format – yourName-studNo.doc.

The document will have 3 sections:

GUI Design: Your designed GUI including buttons, panels, labels and textfields to support your application.

The code: You will **copy** the code from your BlueJ project into this section

The explanation: you will explain your code choices and reasoning. You should place comments in the code to identify the steps in the observer pattern. You also need to include at least one screenshot of testing your code. Include any issues you had and how you resolved them. Minimum 300 words.

(You will also submit a copy of your blueJ project as a zip file)

The deadline for submission of this project is Friday 11:59pm in week 10.

Draft Project B: due Tutorial B Week 10 – draft GUI design review by tutor

All projects submitted after the deadline will incur late penalties of 1 grade band every 2 days (or part thereof).

REMINDER: ALL of the code must be your own unique work using concepts and techniques taught in the subject to satisfy academic integrity.

ASSESSMENT CRITERIA

CRITERIA LINKS

CRITERIA	WEIGHT	SLOs	PLOs
Functional Specifications	35%	1, 3	A1, A3, B1, B2, E1
Code Layout	35%	3, 4, 5	A1,A2,A3, B1, B2, E1
Explanation	30%	1, 3	A1, A3, B1, B2, E1

SLOs: subject learning outcomes
PLOs: program learning outcomes

Assessment Criteria		HD	D	C	P	F
Functional Specifications	35%	The program works correctly in every way and meets all of the functional specifications using MVC architecture. The interface is very intuitive and easy to use.	The program runs and produces mostly correct results using Observer architecture. It also meets most of the other specifications. Some error checking may be missing. The interface is intuitive and easy to use.	The program produces mostly correct results using Observer architecture. Some minor functionality maybe missing. The interface is somewhat intuitive.	The program is producing some incorrect results or a part of the program is incomplete. Some Observer architecture is present but may not be working correctly. The interface and colour usage is ok.	The program is producing all incorrect results, does not run or mostly incomplete
Code Layout	35%	The code is exceptionally well organised and the flow of control is very easy to follow. No design rules or patterns have been broken. Good use of inner panels and inheritance.	The code is easy to follow and well organised. Some minor design rules or patterns have been broken. Some inner panels have been used.	The code is fairly easy to follow and quite well organised. Some design rules or patterns have been broken. Some attempt at using inner	The code is a little hard to read at times. The flow of control is difficult to follow. Some design rules and patterns have been followed.	No enough code written to access correctness.

				panels has been made.		
Explanation	30%	Demonstrates a clear and comprehensive understanding of GUI concepts The code contains comprehensive comments that explain steps in the Observer pattern	Demonstrates a clear understanding of GUI concepts The code contains many comments that explain steps in the Observer pattern	Demonstrates a moderate understanding of GUI concepts The code contains some comments that explain steps in the Observer pattern	Demonstrates some understanding of GUI concepts The code contains minimal comments that explain steps in the Observer pattern	Demonstrates little understanding of GUI concepts The code contains few or no comments that explain steps in the Observer pattern