### 

|  |  |  |  |
| --- | --- | --- | --- |
| **Programme Name** | Undergraduate Computing Suite | | |
| **Module Title** | Advanced Programming | | |
| **Module Code** | MCOMD3ADP | | |
| **Module Start Date** | 23rd September 2019 | | |
| **Module Level** | 6 | **Assessment Type(s)** | Programming Assignment |
| **Word Length / Duration (equivalent** | 4,000 | **% contribution to module mark** | 100% |
| **Submission Date** | **6/1/2020** | **Submission Location** | Blackboard/Turnitin Assignment |
| **Feedback Date** | 3/2/2020 |  |  |
| **Module Leader** | Muna Al-Jepoori | **email** | Muna.al-jepoori@canterbury.ac.uk |
| **Module moderator and Second Marker** | Richard Henson |  | Richard.henson@canterbury.ac.uk |

General advice:

* You are required to back up your work regularly onto your N: drive and on removable storage devices
* Always check the date-stamp on your files before submission, and submit the correct versions
* You **must** submit your work using the software versions currently on the University’s network.

The program must be developed in Java as described below

* You should not use any packages or software tools to automatically generate code. Only Eclipse should be used.

**You should carefully read the Submission instructions later in this document before you submit any work.**

# Learning Outcomes (LOs)

By the end of this module students should be able to:

1. Demonstrate a deep working knowledge and understanding of the software design process using MVC and design patterns;
2. Develop and demonstrate a critical understanding of advanced software programming features available in popular object-oriented programming languages such as Java and C# and how they are used in implementing software;
3. Design and implement a software solution applying advanced OOP features ensuring that the programs employ efficient execution based on sound algorithmic design;
4. Demonstrate an appreciation of the need for design documentation and software testing.

LOs tested by this assignment are: LO1, LO2, LO3 and LO4

# Introduction

Your task is to simulate the operation of a Lift using Java. You are expected to follow the software design process using design patterns and advanced features in java such as multithreading, GUI components, and events generation and handling. You are required to use the MVC architectural pattern including at least Observer, Strategy, and Composite patterns.

# Problem Statement

Create a Java program to simulate the operation of a lift. The lift serves a two-story building (like the one we have in Invicta building at CCCU). This lift is in a busy underground station so it is utilised frequently. To simplify the problem, assume the following:

1. The ground floor has one “Up” button, the first floor has one “Down” button.
2. The controller is to generate users and place them on the floors following the test plan provided.
3. When the lift has no requests, it remains at its current floor with doors closed.
4. There is no maximum capacity of the lift in this assignment.
5. The lift does not have a door attached to it, the only doors we have are the doors of the lift shaft on each floor.
6. The time required to travel between floors is 5 seconds.
7. The door to the lift remains open for 1 second per person to allow for people to enter or exit.
8. The lift starts the day on the ground floor with closed door, after a button is pressed the door is opened to let people enter. If the lift is called in the first floor, then it departs to first floor where the door on the first floor will open on the lift arrival to the floor.
9. A person may be in one of several states;

0 standing on the floor after it has been created,

1 pressing the button on the floor,

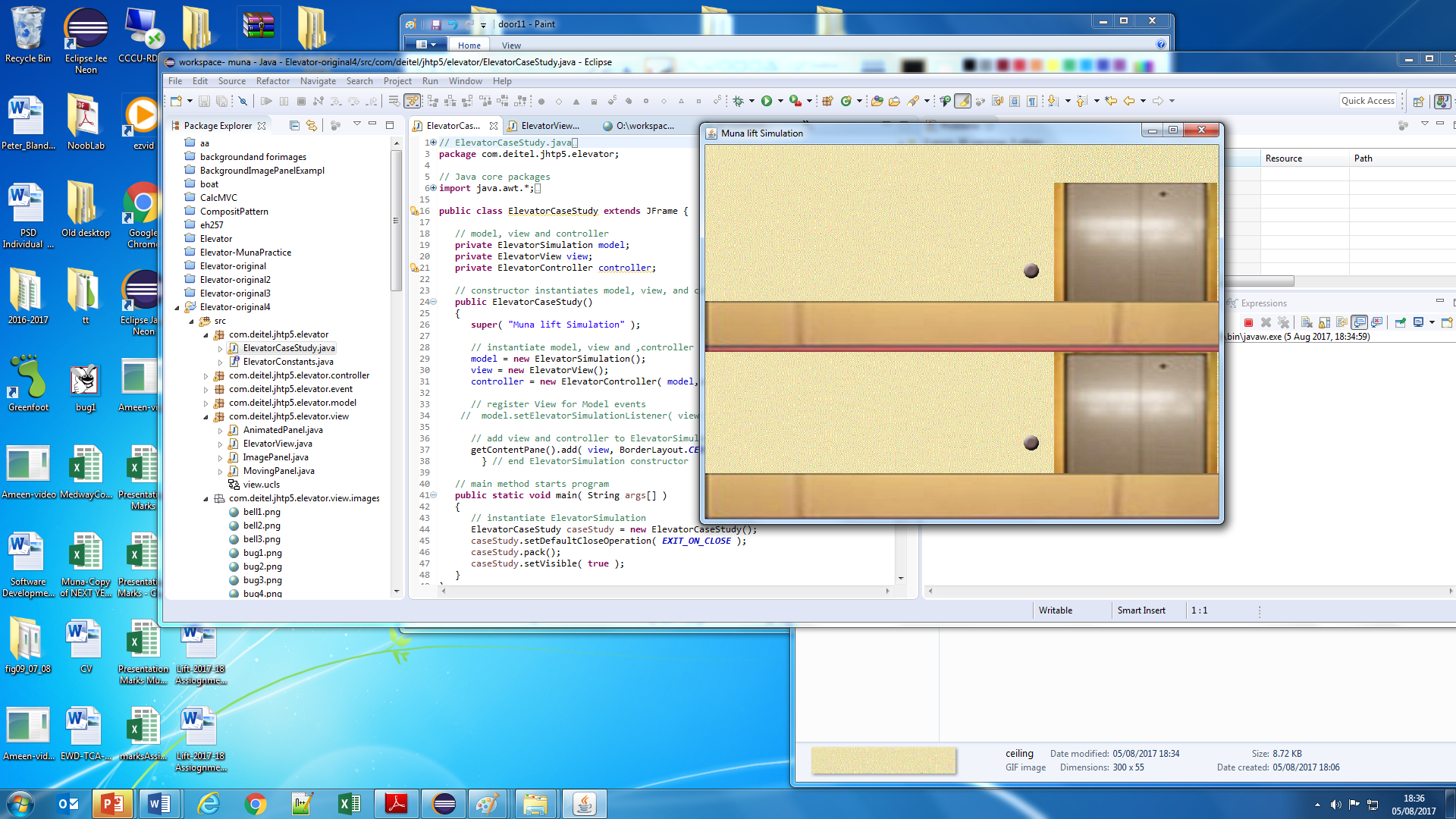
2 pressing the button in the lift,

3 standing in the lift,

4 and exiting lift when the lift reaches the destination and opens the door.

1. Assume that the light in the lift is always “On” so we don’t have to write any code for controlling light inside the lift.
2. In the GUI, buttons on the floor illuminate when pressed. The illumination is cancelled when the lift arrives at the floor.
3. In the GUI, each user on the floor has an image of a person. No need to use different images of people; the same image of a person is enough; it can be used for all instances of Person’s generated, the person image is removed from the lift on exit.
4. In the GUI, when the door is open, you should see the people inside the lift before the door is closed.
5. Your program is required to provide three views;
   1. View 1 is normal report on the console, reporting on every event that is happing in the simulation,
   2. View 2 is the controller GUI view, where you provide components that can be used to execute each test case,
   3. View 3 is an animated GUI reflecting the lift simulation, this worth 20% bonus mark.
6. The above is only a brief outline of the required information system and it is your task to analyse the system and identify requirements before designing and implementing a suitable solution. If you feel that the description is ambiguous in any way, then you must list the assumptions you make, and discuss your ideas with the tutor in tutorial sessions.
7. You might benefit from the following image to get an idea about the initial view before people are generated and the lift is used. Here, we have two doors, two buttons, two floors, wall and the lift behind one of these doors. You need to have your own images.
8. Use the computer time to start each text message with the time where each event takes place ; display the time using the format “yyyy-MM-dd HH:mm:ss.SSS”

*Images used here are modified versions of images used in lift simulation example provided in Kolling, M. (2009) “Introduction to Programming with Greenfoot: Object-Oriented Programming in Java with Games and Simulations.”*



# Test Plan:

You need to test the following test cases where generated users use the lift to reach their destination. Provide screen shots of the views for each test case;

* Test case 1- The lift is standing on ground floor with its door closed; User1 is created by the controller and placed on the ground floor and his destination is first floor.
* Test case 2- The lift is standing on first floor with its door closed; User2 is created by the controller and placed on the ground floor, so and his destination is first floor.
* Test case 3- The lift is on first floor, and user3, user4, and user5 are generated by controller and placed on ground floor, at the same time, User6 is generated on the first floor, both floor buttons were pressed.

# Evaluation and Conclusions: *no more than 300 words*

* You are required to reflect on the design patterns used in your program, discuss how such patterns helped you prepare your code for reuse and maintenance.
* Evaluate your submission indicating limitations and any areas that do not work.
* State any suggested improvements to the program.

# Marking Scheme and Submission

The total mark of this assignment is **120 to be capped at 100, hence you have 20% bonus marks in the marking scheme**. Your submission will be marked per the marking scheme included on the last page of this assignment brief.

## Submission

* The report should have an appropriate Title Page and a Contents Page, and each page is to be numbered.
* The title page should have the following information:
* Course Code, Course Title, Assignment Title, Lecturer’s Name, Student’s Name
* Academic Year, Hand in Date, and the number of hours you spent working on this assignment; this should be an estimated number, keeping a record of the time spent working on this assignment will help you progress.

### Submit via Turnitin:

1. **CODE-Submission only** - submit the project code via the link provided for -a compressed folder that include the full java project code. The program will be marked using Eclipse IDE.
2. **The Report** - Only One MS word or PDF document, do NOT submit a separate file for each section, please. The report will be marked on Turnitin and it should include:

* User manual
* Testing details for the th ree test cases including screen shots of the views for each test case
* Final class diagram generated using ObjectAid UML Explorer for Eclipse
* Evaluation and Conclusions.
* Your java code should be included in your report, in the appendix as TEXT--- **NOT images** please. And before you insert your code in the report, please leave at least three blank lines at the top of each java file, and use the 4th line for the name of the file such as/\* Person.java\*/; this way the code will be easily located by markers of your report.

**Notes**,

* To easily add your code to the report in MS word; use the menu item “Insert---Object—Text from File----*then,* browse to your project folder—, *and select all your java source files*, then click **Insert** button, all your code will be inserted as text. Please, m*ake sure your code has been inserted properly*.
* If code is not submitted as a text, this will require your tutor to do a time-consuming similarity check. Including your code as a text in your report is worth 3 marks as a per the marking scheme

****

GMS Computing Suite 2019-20

### [MCOMD3ADP](https://blackboard.canterbury.ac.uk/bin/common/course.pl?course_id=_2286_1) Advanced Programming

### Assignment – Lift Simulation

**Course Tutor: Muna Al-Jepoori**

**Feedback Sheet**

**Student’s Name:**

|  |  |  |
| --- | --- | --- |
|  | **Mark** | **Awarded** |
| Report |  |  |
| * Code clearly included as appendix in the report | 5 |  |
| * User manual | 3 |  |
| * Class Diagram | 3 |  |
| * Testing Details | 8 |  |
| * Evaluation and Conclusions: | 6 |  |
| Program | 0 | 0 |
| * MVC design pattern | 10 |  |
| * The Complete Model | 20 |  |
| * Use of Multithreading | 10 |  |
| * Proper type and number of events, observers and their implementation | 15 |  |
| * View1- Produce a text report on the console of each event in this simulation | 10 |  |
| * View2- GUI that shows Controller components and Controller logic | 10 |  |
| * View3-Produce is an animated GUI reflecting the lift simulation -bonus   + Static view of the lift world   + Button pressing animation   + Person animation   + Lift movement animation | 20 |  |
| **Final Total: 120 Capped at 100** | 120 |  |

**Comments:**

First Marker: Muna Al-Jepoori

Second Marker: Richard Henson

##### Mark Breakdown – GUIDELINES ONLY.

The markers may reduce marks from those shown for students who use poor coding techniques or use inappropriate solutions.

Report

Code included as appendix in the report 5 marks

0 Not Present or Completely inappropriate content.

1-2 Report OR code present, or partial of both

3 Both Present and Complete

4-5 Both Present, Complete and suitably formatted , file names included

User manual 3 marks

0 No User manual or inappropriate user manual

1-2 Basic User manual that requires understanding beyond the intended audience to use

3 good user manual that helps intended audience without reference to other items.

Class Diagram 3 marks

0 No Class diagram

1-2 Basic Class diagram showing classes, may be missing elements such as relationships, multiplicity, attributes, method, access modifiers or may contain inappropriate elements

3 Good class diagram that matches solution and contains all required element, appropriate solution.

Testing Details 8 marks

Test 1- *3 marks*

0 – no test present

1-2 for partially complete test

3 marks for complete test

Test 2- *2 marks*

0 – no test

1 – partial complete test

2 – complete test

Test 3 – *3 marks*

0 – no test present

1-2 for partially complete test

3 marks for complete test

Evaluation and Conclusions: 6 marks

0 no Evaluation and Conclusions

1-2 for partially attempt

3-4 marks for strong reflection and evaluation

5-6 Complete Evaluation and Conclusions including any suggested improvements to current submission.

Program

MVC design pattern 10 marks

0-3 MVC pattern not used, or such a weak attempt that pattern not really used

4-6 Basic attempt at Pattern use

7-9 Strong attempt at use of pattern, minor errors at most

10 Complete and strong use of MVC

The Complete Model 20 marks

0-7 no or basic attempt to code classes and methods

8-12 at least one class completely coded and partial attempt with other classes

13-16 strong attempt at coding most identified classes

17-20 complete model as reflected by class diagram

Use of Multithreading **10 marks**

0-3 Threads not used, or such a weak attempt that Multithreading not really used

4-6 Basic attempt at Multithreading use

7-9 Strong attempt at use of Multithreading, minor errors at most

10 Complete and strong use of Multithreading

Proper type and number of events, observers and their implementation 15 marks

0-5 no or weak attempt to define and observe events

6-10 Good attempt at identifying reasonable number and types of events

11-15 Strong or complete events and observers

View1- Produce a text report on the console of each event in this simulation 10 marks

0-3 no or unrelated and weak attempt

4-6 reports on at least one actor’s events

7-9 reports on more than one type of events produced by different actors

10 Complete report

View2-GUI that shows Controller components and Controller logic 10 Marks

0-3 no or weak attempt

4-6 some components with partial implementation

7-9 good implementation of major components

10 Strong and complete implementation

View3-Produce is an animated GUI reflecting the lift simulation –bonus 20 marks

0-5 marks each depending on details of implementing each of the following:

* Static view of the lift world
* Button pressing animation
* Person animation
* Lift movement animation