|  |  |
| --- | --- |
| Submission Deadline | Marks and Feedback |
| Before 10am on:  02/11/2023 | **20 working days after deadline (L4, 5 and 7) 15 working days after deadline (L6) 10 working days after deadline (block delivery)**  30/11/2023 |





|  |  |
| --- | --- |
| Unit title & code | Object Oriented Programming and Software Engineering CIS016-2  Desktop Application Development and Software Engineering CIS0116-2 |
| Assignment number and title | Assignment 1: Control an Elevator - A C# Project |
| Assessment type | WR-I (Individual Coursework) |
| Weighting of assessment | 30% |
| Unit learning outcomes | 1. Demonstrate the following knowledge and understanding  • Systematically categorize the concepts, principles and techniques for object-oriented programming and software engineering process.  2. Demonstrate the following skills and abilities  • Analyse a complex problem and apply a programming language to design, develop and critically evaluate a software solution to the given problem; and analyse and apply a range of concepts, principles and techniques to the software development process. |





|  |
| --- |
| What am I required to do in this assignment? |
| Aim This assignment aims to develop students’ capabilities and skills of solving real-world problems with C# programming language.    A simple version of the elevator is illustrated below:   Task description: **Task 1:** To create a GUI which contains **(20 marks)**   * Two request buttons corresponding to the two floors, respectively * One control panel with two buttons and a display window * Two display areas that display the status of the elevator, i.e. which floor the elevator currently stays, one is of each floor * A log button, which trigger the historical information of the elevator operation to be displayed   **Task 2:** To create a control program that processes the events published by the GUI. **(10 marks)** That is,   * When any request button is pressed,   + the elevator appears in the corresponding floor   + then the display areas and the display window on the control panel show the corresponding floor number at the same time * When the floor number buttons on the control panel are pressed,   + the elevator appears to the corresponding floor   + then the display areas and the display window on the control panel show the corresponding floor number at the same time   **Task 3:** To create a log which records all operations. **(30 marks)** That is,   * The corresponding status of the elevator and the time information is stored in a database (it can be MS Access or other equivalent databases) * To display the information which has been stored in the database on the GUI * Optimise the portability of the application, for example by using relative path instead of absolute path * Optimise the maintainability, for example by avoiding any duplication among the event handlers over the database related functions   **Task 4:** To animate the events described in Task 2 using delegation and timer **(10 marks)**  **Task 5:** Additional Marks (This task is only considered when Task 1-4 are fully completed)  *To further optimise the programs developed in Tasks 1 to 4.* ***(20 marks)*** *That is,*   * *5.1 Improve the robustness of the application by eliminating logical errors and handling unexpected situations with exception* * *5.2 Optimise the efficiency of the application by separating GUI operation and database operation into multiple tasks concurrently via BackgroundWorker* * *5.3 The elevator might have more than two floors, apply the “state design pattern” and “the open–closed principle (OCP)” in the program to improve the modularity of the application. The state is expected to be dispatched dynamically rather than pre-determined with if/switch*   **Task 6:** To prepare a test report, the “Marking Matrix with Self-Assessment” table (which is listed at the end of document) should be included in the report **(10 marks)** F2F Demonstration and reflection: Demonstrating and explaining your project and source code on 02/11/2023 or 03/11/203 during the practical sessions. Your time slot  will be provided before the week of the demonstration.   Submission: What to submit – ***Testing*** ***Report*** and ***The Entire C# Project Code***    The entire C# project including all folders and files, the database and the test report  Name your submission – Your submission should be named with the following format: studentIDAssignmentOne.zip  Where to submit – Under the assignment 1 folder in the Assessment menu of this Unit’s BREO site, called “Assignment One C#” |
| Is there a size limit? |
| There is no limit on the size of the source code.  The testing report should not exceed 3000 words (source code in the report is not counted as the part of the work count). |
| What do I need to do to pass? (Threshold Expectations from UIF) |
| Design and produce a demonstrable artefact using C# within the .NET using the object-oriented paradigm. Check the marking matrix at the end of the brief for more details.  The demonstration is compulsory. You will not be able to pass the assignment if you miss the presentation in person. |
| How do I produce high quality work that merits a good grade? |
| Detailed in the marking matrix at the end of the brief. |
| How does assignment relate to what we are doing in scheduled sessions? |
| Implementation of the knowledge and skills delivered from Week 1 to Week 6. |



|  |
| --- |
| How will my assignment be marked? |
| Your assignment be marked according to the threshold expectations and the criteria on the following page.  You can use them to evaluate your own work and estimate your grade before you submit. |

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Lower 2nd – 50-59%** | **Upper 2nd – 60-69%** | **1st Class – 70%+** |

Marking Matrix with Self-Assessment

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Task Number** | **Sub-tasks** | **Possible Marks** | **Self-assessment (completed Yes/No)** | **Reference to your testing report** | **Mark Awarded** |
| **Task 1** | Complete GUI for Task 1 | 10 |  |  |  |
| Skeleton of event handlers in place for all buttons | 10 |  |  |  |
| **Task 2** | All event handlers are functional | 10 |  |  |  |
| **Task 3** | Database (DB) is designed and can be connected | 5 |  |  |  |
| Log Information can be retrieved from DB and displayed in the GUI | 5 |  |  |  |
| When the log button is pressed, log information is sent to and stored in the DB | 5 |  |  |  |
| Use the disconnected model rather than connected model (Data source is updated via DataAdapters Update() method instead of ExecuteNonQuery() method) | 5 |  |  |  |
| Using relative path instead of absolute path | 5 |  |  |  |
| Avoiding any duplication among the event handlers over the database related functions | 5 |  |  |  |
| **Task 4** | Events described in Task 2 animated using delegation and timer | 10 |  |  |  |
| **Task 5** | Eliminating logical errors and handling exceptions with try and catch | 5 |  |  |  |
| Optimise the efficiency of GUI by implementing multiple tasks concurrently via BackgroundWorker | 5 |  |  |  |
| Use state patterns instead of if-else statements to accommodate future changes of the requirement | 10 |  |  |  |
| **Task 6** | Testing report | 10 |  |  |  |
| **Total** |  | 100 |  |  |  |