|  |  |  |
| --- | --- | --- |
| **Software Implementation Checklist** | **Faculty of Computing, Engineering and the Built Environment** | New Logo Tiny |

|  |  |
| --- | --- |
| Please fill in your name and student ID in the table below. | |
| **Student Name** | *ShiyuanWang* |
| **Student Number** | 1924230209 |
| **Course and Year** | 2019/2020 |
| **Module Code** | CMP4266 |
| **Module Title** | Computer Programming |
| **Module Leader** | Dr. Shadi Basurra |
| **Assessment items:** | 1. Computer programme solution and testing **(Total weight 70%)**. This must include the following parts; 2. Design documentation submitted online (**Weight 15%**). This must include the two flow chart diagrams. 3. Programme source code submitted online (**Weight 40%**). 4. Testing and evaluation inclusive of test cases submitted online. (**Weight 15%**). |

|  |  |
| --- | --- |
| Students are required to complete this software implementation checklist for their Bank System developed in Python. You need to select only the features that were implemented in your code. You can select all/some features from any marking range as long as they have been implemented in the code submitted for this assessment. For example, you can select all/some features from the marking range 40%-50% and all/some features under the range 70%-79% etc. This implementation checklist should be submitted alongside the implementation source code of the system and the testing and evaluation documentation.  **Important notice**: This checklist will assist the lecturer when marking your code, hence, you should only select the feature requirements that have been implemented in your code. Even if some features are not working correctly, you can still select them as long as there is evidence in your code showing the implementation attempt. However, it is not acceptable for a student to claim the implementation of features that were not attempted/implemented in the system. False claims is a clear indication that the student does not understand the submitted code, hence, the submission will be investigated further for plagiarism, and the lecturer marking the assessment may invite the student to explain all/parts of the submitted code. | |
| **Software implementation checklist (Compulsory).** | |
| **Achieving a mark to maximum of 40%** | |
| The application **must** implement **all** the above and the following: | |
| **Create the necessary classes and functions which allow Admins to perform the following tasks:** | |
| * Admin Login |  |
| * Search for a particular customer to perform various banking operations on a customer bank account i.e. check balance, deposit or withdraw money etc. |  |
| * Deposit money into a customer account |  |
| * Withdraw money from a customer account |  |
| * Check current balance for a customer |  |
| * View customer details e.g. name and address |  |
| * Update customer information e.g. name and address |  |
| **To achieve a mark of 41% to maximum of 50%** | |
| The application **must** implement **all** the above and the following: | |
| * Close a customer account i.e. remove customer from the system |  |
| * Update admin own information i.e. name and address |  |
| * Print all customers details |  |
| **Achieving a mark of 51% to maximum of 70%** | |
| The application **must** implement **all** the above and the following: | |
| * Customers can have different types of bank account. Accounts will differ in their name, interest rate and overdraft limit etc. |  |
| * The bank system should be able to store and load all customers’ data from and into a file. |  |
| **Achieving a mark of 71% to maximum 80%** | |
| The application **must** implement **all** the above and the following: | |
| * Transferring funds/money from one account to another. For example, an Admin can transfer funds from one customer account to another customer account. |  |
| * Admins can request a management report. This should show the following information; |  |
| * + Total number of customers in the system |  |
| * + The sum of all money the customers currently have in their bank account. |  |
| * + Using individual account balance, the report should calculate the sum of interest rate payable to all accounts for one year, note, if customers have different account types, their rates will vary. |  |
| * + Total amount of overdrafts currently taken by all customers. |  |
| **To achieve a mark of 80% and over** | |
| The application **must** implement **all** the above and the following: | |
| * Development of a suitable Graphical User Interface (GUI) to perform all the above banking functions. |  |

**Note:** students need to submit their system alongside the necessary objects to test their software by the tutors when marking. For example, if your system is able to deal with different types of bank accounts (e.g. business and savings accounts), then you should populate the system with at least two customers that have two or more different account types.