**H17135 – Object Oriented Programming Student: Brodie Harkins**

**Evidence Requirements**

**LO1: Design should demonstrate**

|  |  |
| --- | --- |
| Abstraction, encapsulation and information hiding used where appropriate | x |
| Polymorphism used if appropriate to the solution | x |
| All class-wide variables are private to prevent content coupling | x |
| Class-wide variables are kept to a minimum to ensure a minimum of common coupling | x |
| Data coupling is used (parameter passing) in preference to content or common coupling | x |
| Inheritance used if appropriate to the solution | x |
| Program does not contain a lot of unnecessary data coupling | x |
| Classes are highly cohesive | x |

**LO2: Implementation should demonstrate**

|  |  |
| --- | --- |
| Declaring and initialising variables | x |
| Using operators | x |
| Implementing control structures | x |
| Defining data structures | x |
| Accessing and manipulating data structures | x |
| Using parameter passing | x |
| Creating Classes | x |
| Creating instances of classes | x |
| Creating relationships between classes | x |
| Creating Constructor methods | x |
| Overloading methods | x |
| Use of exceptions | x |
| Use of standard object libraries | x |
| Documenting code | x |
| The program must consist of at least four classes, and at least one ‘one-to-many’ association must be implemented. | x |
| There must also be correct use of encapsulation and inheritance. | x |

**LO3: Testing should demonstrate**

|  |  |
| --- | --- |
| Test Strategy | x |
| Test Plan | x |
| Test data | x |
| Expected Results - Actual Results - ActionTaken | x |

**H17135 – Object Oriented Programming**

**Target Functionality:**

|  |  |
| --- | --- |
| At first start-up: | |
| Pre-populate program with instances of: | |
| * + 1. One Admin user | x |
| * + 1. Three of each other user type | x |
| * + 1. Three of each type of Stock item | x |
| At each subsequent start-up: | |
| * 1. Populate with User objects previously made persistent | x |
| * 1. Pre-populate program with instances of: |  |
| * + 1. Three of each type of Stock item | x |
| 1. At shut-down – make all User objects persistent | x |
| 1. Admin Login | x |
| 1. User Login | x |
| 1. For each type of user – display screen(s) with options that are appropriate for that type of user | x |
| For Admin Login: | |
| For Users: | |
| * + 1. Add new User | x |
| * + 1. Record Loan |  |
| * + 1. Record Loan being returned |  |
| * + - 1. Calculate fine if appropriate |  |
| * + 1. List all Users | x |
| * + 1. List all Logins | x |
| * + 1. List all Loans |  |
| For Stock: | |
| * + 1. Add new Stock Item (all types) | x |
| * + 1. Edit existing Stock Item | x |
| For User Login: | |
| * 1. Edit profile excluding own ID |  |
| * 1. View Logins history |  |
| * 1. View Loan history (if appropriate) | x |
| * 1. Reserve Stock item |  |
| * 1. Search catalogue by Stock item Title | x |

Sample output from trying to lend an item. The item could later be returned, so it seems it was borrowed successfully:

A screenshot of a computer

Description automatically generated with medium confidence

There also seems to be some weird output after returning an item. I’m not entirely sure how this should look:

Text

Description automatically generated

Sample output after attempting to return an item late:

Text

Description automatically generated

Strange output on reserving an item – unclear whether the item has been reserved (once or twice!)

Graphical user interface, text

Description automatically generated