1.

The Problem –

I need to create a basic banking system that will allow users to input their personal details, create a current account or and ISA account and have some sort of algorithm of logic applied to the ISA to ensure the ISA deposit limit is not reached. I must also create a demo of a GUI program that can display an accounts balance.

* 1. Identify a set of classes and their interrelationships to address the problem.

In my program I will use the following classes;

Account – Which will be the subject of polymorphism as it can be an ISA or current account.

Balance Displayer – This will create an account automatically and display a test balance in a GUI format.

Bank Console – This will be the basis of the user interaction and will implement the current account and ISA account.

Current Account – This will inherit from Account and be used in the Bank Console.

Isa Account – This will inherit from Account and be used in the Bank Console.

Customer – This will hold all customer details and be used in the Bank Console.

* 1. Make effective use of encapsulation, inheritance and polymorphism.
  2. Select and re-use pre-existing objects and templates, specializing as required.
  3. Structure the design so the objects communicate efficiently
  4. Specify the properties and behavior of classes to allow efficient implementation, selecting appropriate data types , date and file structures and algorithms.
  5. Record the design using well established notation.

JAVADOC, UML

2.

2.1 Make effective use of basic programming language features and programming concepts to implement a program that satisfies the design specification.

2.2 Make effective use of the features in the programming environment.

2.3 Make effective use of user interface components in the implementation of the program.

2.4 Make effective use of a range of debugging tools.

3.

3.1 Apply standard naming and comment conventions.

3.2 Apply appropriate data validation and error handling techniques.

4.

4.1 Develop and apply a test strategy consistent with the design identifying appropriate test data.

4.2 Apply regression testing consistent with the test strategy.

4.3 Use appropriate tools to estimate the performance of the program.

5

5.1 Record the final state of the program in a form suitable for subsequent maintenance.

5.2 Provide end-user documentation that meets the users needs.