INTRODUCTION

In your role as a software developer, you have been contracted by QUB's Logistics department to create a Java application capable of monitoring the availability of products sourced from a range of different suppliers. Specifically, you are required to build and test a prototype system that meets the deliverables outlined in the sections below.

PART 1: CORE FUNCTIONALITY (50 MARKS AVAILABLE)

In order to demonstrate that the proposed Java application is feasible, the following features must be implemented.

1. A validated console menu system with the following options should be provided:

Menu Option	Description of Functionality
Print All Products	This option will print all products stored in the application grouped by their supplier to the console window.
Add New Supplier	This allows a new supplier to be added to the list of suppliers currently stored by the application. Note that a new address object will have to be created as part of this process.
Add New Product	This allows a new product to be added to the list of products available from a specific supplier.
Exit Application	This option allows the user to terminate the application.

2. Implementation of the object classes and enumerator class shown in Figure 1 below.

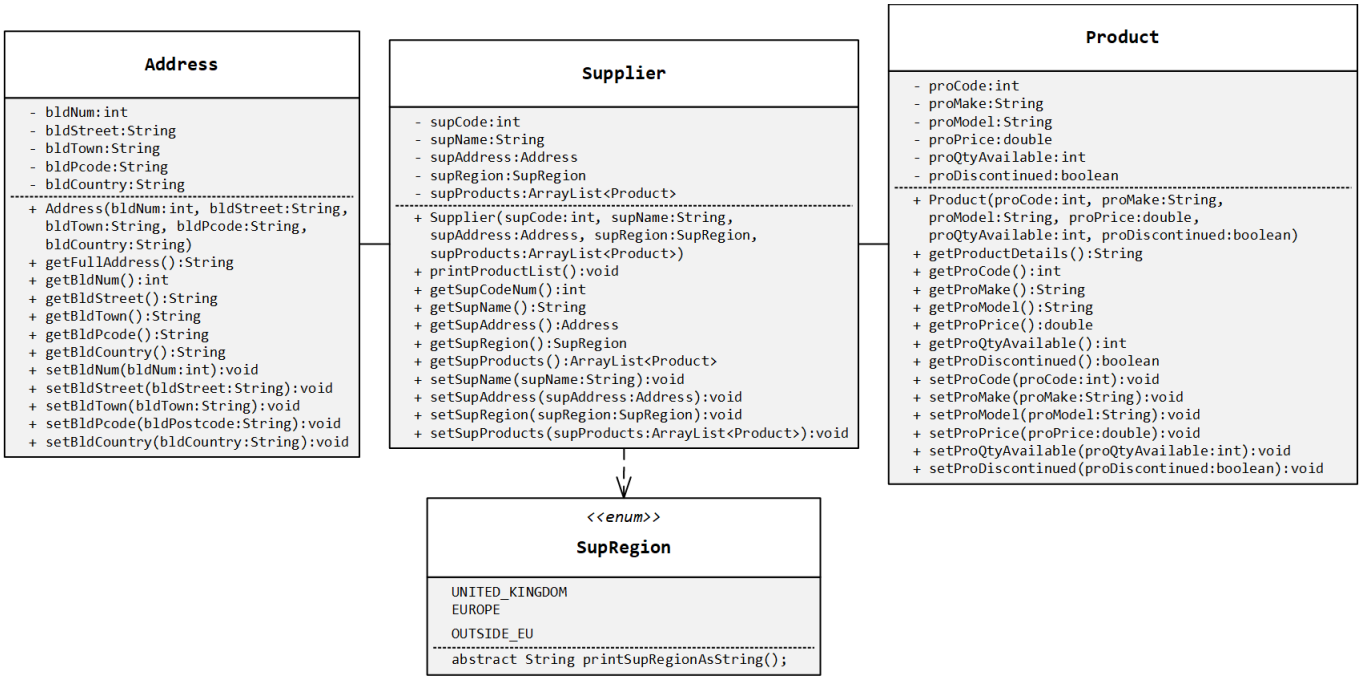

Figure 1: UML Class Diagram for QUB Logistics Department Supplier Application.

PART 2: ADDITIONAL FEATURES (30 MARKS AVAILABLE)

Further to the features outlined above, QUB Logistics is interested in evaluating any additional functionality that may be of benefit to the application. Accordingly, within the time and resource constraints of the project, you have been asked to investigate and where possible implement additional functionality. This may include, but is not limited to:

- Enabling the user to modify content or delete instances of the object classes shown in Figure 1.
- Providing search functionality, for example, a list of all products between a specified price range. Alternatively, producing a list of all discontinued products.
- Providing the facility to check the stock of a given product, or produce a quote for ordering a number of products.
- Validating user inputs beyond simple menu selection.

NOTE: Each piece of additional functionality in line with those described above may be awarded up to six marks each depending on the quality of the implementation.

To help QUB review any additional features implemented, include a short Microsoft Word document with your submission that provides a title and short description (max 2-3 sentences) for each feature.

NOTE: Part 1 and Part 2 should be submitted as separate packages as described in the general instructions.

Part 3: Testing the Application (20 marks)

As part of QUBs quality assurance procedure, you are required to submit evidence that the application has been thoroughly tested. Specifically, QUB requires two narrated demonstration videos:

- Video 1 should demonstrate how the application meets the core requirements described in 'Part 1' of this document.
- Video 2 should demonstrate any additional features, which relate to 'Part 2' of this document.

Furthermore, a completed testing document, which describes the tests conducted, is also required. The scope of this testing should cover all areas of core functionality outlined in **Part 1** of this assignment specification. Please note that any testing documentation submitted for Part 2 will **not** accrue any additional marks. Additionally, testing documentation must be submitted using the template provided by QUB.

General Instructions

Code Implementation

1. Create a new Java project named 'supplierApp'.
2. Add two packages to 'supplierApp' named 'part01' and 'part02' respectively.
3. Implement your code as described below:
    - 'part01' should contain all code associated with the core requirements.
    - 'part02' should contain all code associated with the core requirements AND the code for any additional features you have implemented.

Video Presentation:

- Two separate video files should be recorded as described in 'Part 3' above.
- Each video should be recoded in MP4 format and not be more than 25MB (instructions will follow on how to compress video).
- The combined length of the videos should not exceed 10 minutes.
- Video files should be named 'video1.mp4' and 'video2.mp4' respectively.

Testing Documentation

- Ensure that the testing is documented using the template provided.

Submission Instructions

1. Create a new folder to hold all your assignment files for submission. The folder should be suitably named and include your student number.
2. Copy and paste the 'part01' and 'part02' folders from your java project into the folder created in step 1. ENSURE that the 'part01' and 'part02' folders contain all of the corresponding .java files.
3. Add your 'video1.mp4' and 'video2.mp4' files to the folder created in step 1.
4. Add your testing spreadsheet to the folder created in step 1.
5. Add the Microsoft Word document describing any additional features to the folder created in step 1.
6. Verify that you have all the required files included in the folder created in step 1.
7. Compress the folder to a zip file and upload it to the assignment location in the CSC1020 module of Queens Online by **4.30pm** on **Tuesday 12th December, 2017.**

NOTE: Please check that you have included the original development Java (.java) files. If you submit .class files these will NOT be marked. These submissions will be date-stamped and in accordance with University regulations, late submissions will be penalised. **Failure to follow the submission instructions may result in a reduced or zero mark for part or all of the assignment!**

CSC1020 MARK SCHEME

GUIDELINES USED IN ASSESSING YOUR CONTRIBUTION TO THE SUPPLIER APPLICATION

MARK OUT OF 100

FAIL: 0-39

The student does not seem to have made much effort or contributed much to the main assignment. Their submission does not compile and/or does not conform to the basic application requirements. The student demonstrates a lack of understanding and/or interest in programming.

PASS: 40-49

The student has something of merit to demonstrate that is a valid part of the Supplier Application but there are some deficiencies in terms of the execution of the program. Most of the basic application requirements have been implemented however, there are issues with some of them. The student is not fully confident about their submission.

SATISFACTORY: 50-59

A reasonable attempt has been made to implement all of the basic application requirements and has started to engage with additional functionality for the application. The student does not convince you that they fully understand the programming but there is evidence of potential.

GOOD: 60-69

The student demonstrates that all of the basic requirements have been implemented and there are some useful additional elements of functionality but perhaps these are lacking in flair. The student demonstrates understanding but does not demonstrate that they had other ideas that they were unable to execute in the time scale.

VERY GOOD: 70-89

The student demonstrates a section of the Supplier Application that is fully compliant with the basic requirements and the additional functionality/features are an integral part of the system. The application is easy to use and intuitive and validation has been fully incorporated. The student is fully able to fluently explain the programming used and they have other ideas for future improvements which they would like to implement.

EXCELLENT: 90-100

The student demonstrates a section that is excellent in terms of the assessment requirements; it is fully functional and well tested. There are desirable extra features that make for a good user experience. The student demonstrates enthusiasm, expertise and confidence in their abilities as a programmer.