Inventory Management System

Introduction

This assignment is due by 10:00 pm Friday of Week 7 (7th September, 2018). It is worth a total of 15% of the marks for your final assessment in this unit. **Heavy penalties will apply for late submission.** This is an individual assignment and must be your own work. You must attribute the source of any part of your code which you have not written yourself. Please note the section on plagiarism in this document.

This assignment is structured as a 10% + 5% assessment. The code submitted for this assignment will be assessed for 10% of the marks and a unit test during the tutorial time in Week 8 will be assessed independently for 5% of the marks allocated. Please note that the unit test is a hurdle requirement for the assignment. Failing to complete the unit test will result in no marks being allocated for the entire assignment.

The assignment must be done using the BlueJ environment.

The Java source code for this assignment must be implemented according to the **Java Coding Standards for this unit.**

Any points needing clarification may be discussed with your tutor in the lab classes.

Specification

For this assignment you will write a program which will simulate an inventory management systems for a products company based in Melbourne, Victoria. The system will allow a salesperson to register products which are sold by the company. The salesperson can also register a sale for a customer which allows the customer to buy products which are sold by the company. Lastly, the system will keep track of the quantity of products available and will allow the salesperson to finalize the sale for the customer. Keep in mind that the company using this system operates as a wholesaler so there is a minimum quantity which must be ordered for all listed items. From here on, **the salesperson(s) are the referred to as the users of the system.**

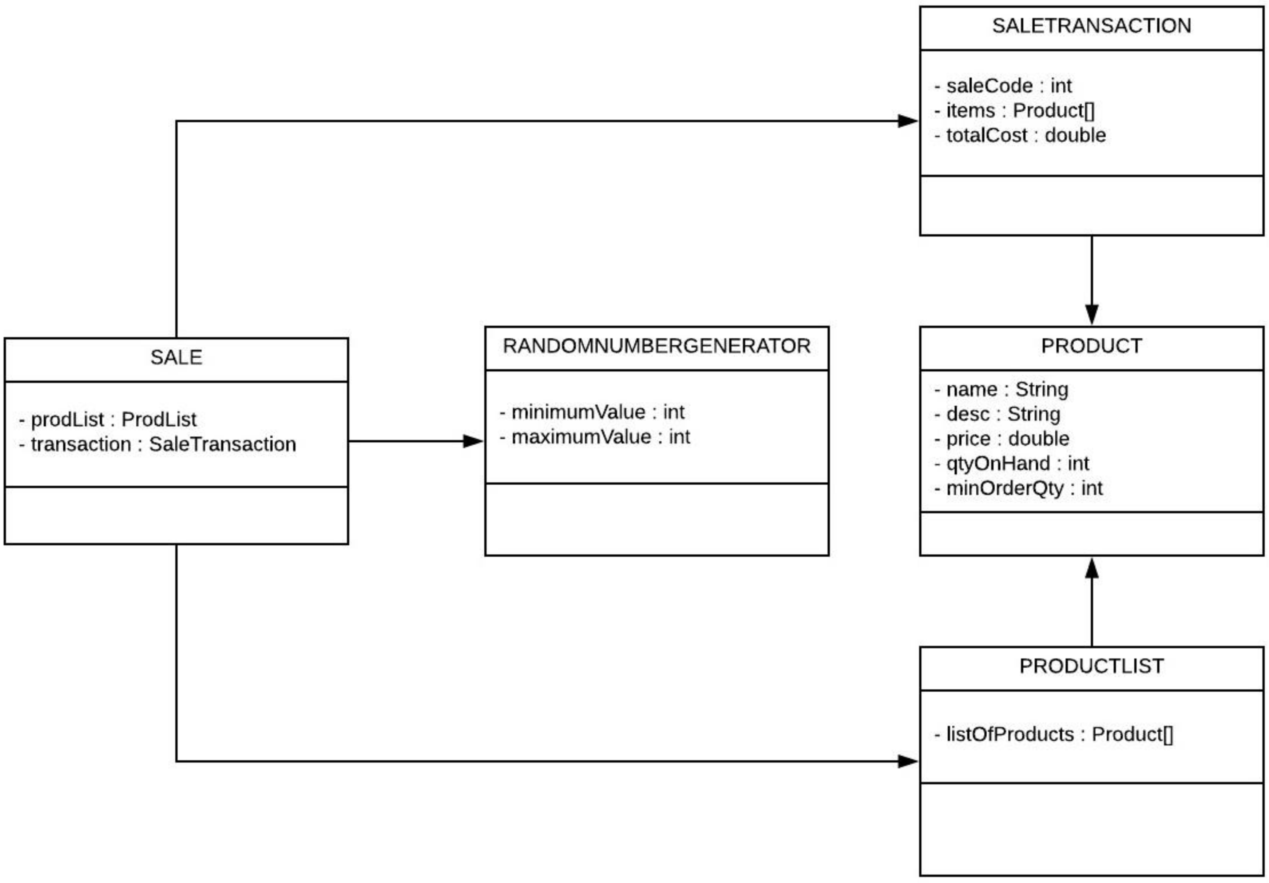
This section specifies the required functionality of the program. **Only a text interface is required for this program**; however, more marks will be gained for a program that is easy to follow with clear information/error messages to the user.

System Operation

The following section outlines the functionality of the program.

Class Diagram:

The program is to be designed using the following class diagram specification:

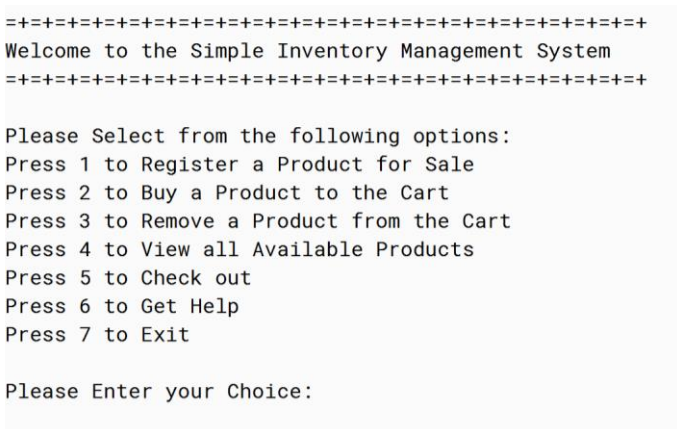


The above class diagram only outlines the various classes and the required attributes within each class. The methods which should be included are open to interpretation by the programmer. However, the required attributes cannot and must not be changed, added, or removed. The method should include a default constructor, a non-default constructor, accessor and mutator methods for each attribute, and a display method to present the state of the object.

The interaction between the classes must follow the outlined class diagram above as well. No additional objects should be created. The requirements is for students to demonstrate their understanding of using the pass by reference in Java, and the use of parameters to do message passing.

Program Logic:

When the Inventory Management System starts, the following options will be presented to the user:



```
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
Welcome to the Simple Inventory Management System
=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+

Please Select from the following options:
Press 1 to Register a Product for Sale
Press 2 to Buy a Product to the Cart
Press 3 to Remove a Product from the Cart
Press 4 to View all Available Products
Press 5 to Check out
Press 6 to Get Help
Press 7 to Exit

Please Enter your Choice:
```

Option 1:

Allows the user to register a new product to be sold. Every new product registered must meet the following criteria:

- Product Name – Name of the product
  - Must be between 3 and 25 characters long.
  - Cannot be repeated for two or more products.
  - Should be case insensitive, i.e. lower and upper case letters should be treated the same way.
- Product Description – Brief description of the product being sold
  - Must be between 1 and 50 characters long.
- Product Price – Unit price for a single item of the product
  - Must be a double value > 0
- Quantity On Hand – Quantity available for purchase
  - Must be an integer between 0 and 10.
  - Must be randomly generated by the system when a product is created.
- Minimum Order Quantity – Minimum Quantity to be Ordered
  - Must be an integer between 1 and 5
  - Must be randomly generated by the system when a product is created.

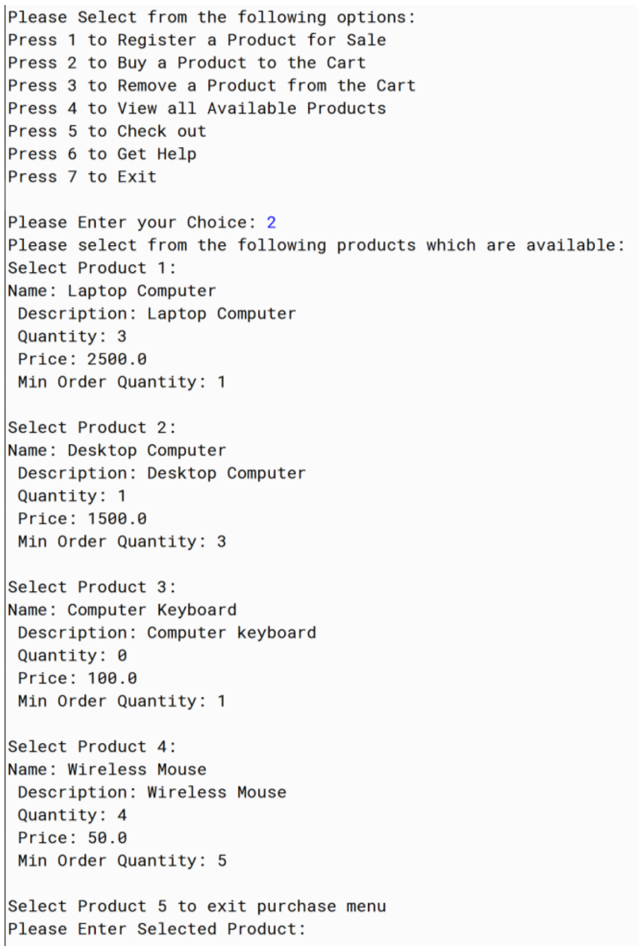
- The system only allows a maximum of FIVE products which can be registered.

Option 2:

Allows the user to purchase a new product which has been registered. Every product bought must meet the following criteria:

- Only products added in Option 1 can be purchased. If no products have been registered, an error message must be shown to the user.

- User should be shown a submenu to select from the registered products as follows:



```
Please Select from the following options:
Press 1 to Register a Product for Sale
Press 2 to Buy a Product to the Cart
Press 3 to Remove a Product from the Cart
Press 4 to View all Available Products
Press 5 to Check out
Press 6 to Get Help
Press 7 to Exit

Please Enter your Choice: 2
Please select from the following products which are available:
Select Product 1:
Name: Laptop Computer
 Description: Laptop Computer
 Quantity: 3
 Price: 2500.0
 Min Order Quantity: 1

Select Product 2:
Name: Desktop Computer
 Description: Desktop Computer
 Quantity: 1
 Price: 1500.0
 Min Order Quantity: 3

Select Product 3:
Name: Computer Keyboard
 Description: Computer keyboard
 Quantity: 0
 Price: 100.0
 Min Order Quantity: 1

Select Product 4:
Name: Wireless Mouse
 Description: Wireless Mouse
 Quantity: 4
 Price: 50.0
 Min Order Quantity: 5

Select Product 5 to exit purchase menu
Please Enter Selected Product:
```

- When a product is selected, the quantity of hand should be greater than the minimum order quantity. If not, an error should be shown to the user. If yes, then it should be added to the array.

- A user can purchase a maximum of THREE items only.

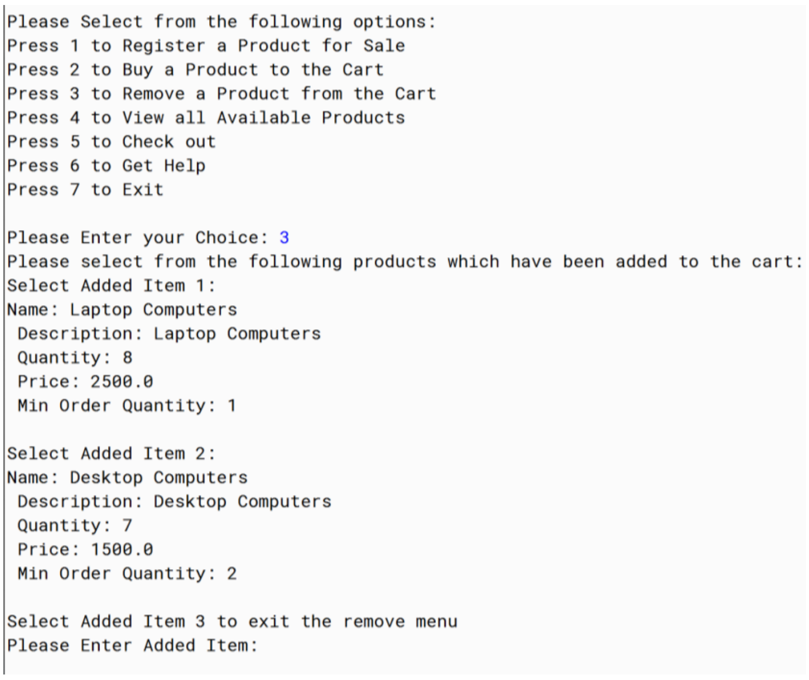
- On purchase a product, the Total Cost of the SaleTransaction should be updated. However, the Quantity on Hand should NOT be changed until option 5 is selected. Each order purchase must be for the minimum order quantity.

Option 3:

Allows the user to remove a product which has been purchased in option 2. Every product removed must meet the following criteria:

- Only products added to the array in Option 2 can be removed. If no products have been purchased, an error message must be shown to the user.

- User should be shown a submenu to select from the purchased products as follows:



```
Please Select from the following options:
Press 1 to Register a Product for Sale
Press 2 to Buy a Product to the Cart
Press 3 to Remove a Product from the Cart
Press 4 to View all Available Products
Press 5 to Check out
Press 6 to Get Help
Press 7 to Exit

Please Enter your Choice: 3
Please select from the following products which have been added to the cart:
Select Added Item 1:
Name: Laptop Computers
 Description: Laptop Computers
 Quantity: 8
 Price: 2500.0
 Min Order Quantity: 1

Select Added Item 2:
Name: Desktop Computers
 Description: Desktop Computers
 Quantity: 7
 Price: 1500.0
 Min Order Quantity: 2

Select Added Item 3 to exit the remove menu
Please Enter Added Item:
```

- On removing a product, the Total Cost of the SaleTransaction should be updated. However, the Quantity on Hand should NOT be changed until option 5 is selected. Each order removed must update the Total Cost based on the minimum order quantity.

Option 4:

Allows the user to view all the currently registered products which are available to be purchased.

Option 5:

Allows the user to finalize a sale and checkout from the system. Every final sale must meet the following criteria:

- Only products added to the array in Option 2 can be finalized for the sale. If no products have been purchased, an error message must be shown to the user.

- For each item purchased, the system will

   - Check the product purchased in the SaleTransaction class with the Product class objects. If a match is found, the Quantity on Hand will be reduced by the minimum order quantity for the purchase.
   - If the same product is ordered multiple times and IF the Quantity on Hand falls below the minimum order quantity, the sale of that specific item is cancelled and not finalized and an error message is displayed to the user. The rest of the sale will be finalized.

Option 6:

- Allows the user to view a brief but descriptive explanation on how to use the system and what the various options do within the menu.

Option 7:

- Exits the system.

Program design

The objective of this assignment is for students to understand coding, as well as to get the basics of design.

Your program MUST consist of the following classes:

1. Product
2. RandomNumberGenerator
3. SaleTransaction
4. ProductList
5. Sale

You MUST follow good programming practices and use loops where required to ensure good program design.

ALL classes MUST have a default constructor, a non default constructor (if applicable), accessor methods for each field, mutator methods for each field, and a display method to provide the current state of the object.

The fields shown in the included class diagram on page 2 are the MAXIMUM fields expected. Your implementation may have fewer fields but should not exceed these.

The data type of each field has been shown in the class diagram and must adhere to the specifications provided.

The explanation for some of the fields for each of the classes has been provided under the Program Logic section above. The rest are included below.

Product class

The Product class will specify the attributes and behaviours of a product. The description of the fields are provided in the program logic section.

RandomNumberGenerator class

The RandomNumberGenerator class will specify the attributes and behaviours for generating a random number between a set minimum and a set maximum. An object of the RandomNumberGenerator class will have the following fields only:

- *minimumValue* – (int) stores the minimum value of the number to be generated
- *maximumValue* – (int) stores the maximum value of the number to be generated

ProductList Class

The ProductList class will store an array of product objects which are available to be purchased. An object of the ProductList class will have the following fields only:

- *listOfProducts* – (Product[]) array of Product objects of maximum size 5.

This class is responsible for adding products to the array, removing products from the array, etc.

SaleTransaction Class

The SaleTransaction class will specify the attributes and behaviours of a sale transaction. An object of the SaleTransaction class will have the following fields only:

- *saleCode* – (int) randomly generated sale code number between 1000 and 9999.
- *items* – (Product[]) array of Product objects of maximum size 3.

This class is responsible for recording products purchased and removed from the shopping cart.

Sale Class

The Sale class will be the main control class of the program and will specify the attributes and behaviours of the system. An object of the Sale class will have the following fields only:

- *prodList* – (ProductList) object of the ProductList class.
- *transaction* – (SaleTransaction) object of the SaleTransaction class.

This class is responsible for facilitating the main functioning of the program and for integrating the various classes together to be able to provide the required functionality for the program to run.

Hints and Suggestions

The following hints and suggestions may be useful in designing your program:

- To facilitate a clean layout of your program, consider looking at the tutorial notes in the homework exercises on how to clear the terminal screen in BlueJ, if required.
- Input validations for numeric inputs are NOT required at this stage. For numeric inputs, read the input from the keyboard as a String and then consider using the Integer.parseInt() method to understand how a String can be converted into an integer .
- An important aspect to understand to get this class to work correctly is how to PASS BY REFERENCE when using objects. Keep in mind Arrays are objects.
- Lastly, when working with arrays, keep in mind that arrays CAN store null objects. So while extensive validation does not have to be done for the program, your program MUST validate if an array element is null or not to prevent your program from crashing.

Assessment

Assessment for this assignment will be done via an interview with your tutor and a unit test assessment.

Coding Assessment (10%)

The marks will be allocated as follows:

- 35% - Object-oriented design quality. This will be assessed on appropriate implementation of classes, fields, constructors, methods and validation of the object's state.
- 5% - Adherence to FIT9131 Java coding standards.
- 60% - Program functionality in accordance to the requirements.

You must submit your work by the submission deadline on the due date (a late penalty of 20% per day, inclusive of weekends, of the possible marks will apply - up to a maximum of 100%). There will be no extensions - so start working on it early.

Marks will be deducted for untidy/incomplete submissions, and non-conformances to the FIT9131 Java Coding Standards.

Unit Test Assessment (5%)

This assessment will be a written assessment and will be held during the tutorial times in Week 8 after the submission. Keep in mind that students are expected to have finished ALL their code by the submission deadline, and more importantly have to have done their own work.

The unit test will be a closed book test wherein students will be required to WRITE code to answer a few questions. The questions will cover material from Weeks 1 – 6 of the lectures. Students should bear in mind that BlueJ or any other electronic device will not be permitted, so students need to know HOW to write their own code in order to finish the unit test hurdle.

Unit test for DE / Off-Campus students will be scheduled for Week 8, but due to the examination environment of the unit test, an equivalent moodle quiz or a skype assessment may be administered which may be slightly different than the on-campus format of the assessment.

Interview

You will be asked to demonstrate your program at an "interview" following the submission date, in Week 8. At the interview, you will be asked to explain your code/design, modify your code, and discuss your design decisions and alternatives. Marks will not be awarded for any section of code/design/functionality that you cannot explain satisfactorily (the marker may also delete excessive in-code comments before you are asked to explain that code).

In other words, you will be assessed on your understanding of the code, and not on the actual code itself.

For **on-campus students**, interview times will be arranged in the tutorial labs in Week 7. It is your responsibility to attend the lab and arrange an interview time with your tutor. **Any student who does not attend an interview will receive a mark of 0 for the assignment.** The actual interviews will take place in Week 8.

For **off-campus learning (OCL)** students, the interviews will be organised during week 7 and will take place online via Skype or other video facility during Week 8. It is your responsibility to make yourself available for an interview time. **Any student who does not attend an interview will receive a mark of 0 for the assignment.**

Submission Requirements

The assignment must be uploaded to Moodle by 10:00 pm Friday of Week 7 (7th September, 2018).

The submission requirements for Assignment 1 are as follows:

A .zip file uploaded to Moodle containing the following components:

- the BlueJ project you created to implement your assignment.
- a completed **Assignment Cover Sheet**. This will be available for download from the unit's Moodle site before the submission deadline. You simply complete the editable sections of the document, save it, and include it in your .zip file for submission.

The .zip file should be named with your Student ID Number. For example, if your id is 12345678, then the file should be named 12345678_A1.zip. Do not name your zip file with any other name.

It is your responsibility to check that your ZIP file contains all the correct files, and is not corrupted, before you submit it. If you tutor cannot open your zip file, or if it does not contain the correct files, you will NOT be assessed.

Marks will be deducted for any of these requirements that are not complied with.

Warning: there will be no extensions to the due date. Any late submission will incur the 20% per day penalty. It is strongly suggested that you submit the assignment well before the deadline, in case there are some unexpected complications on the day (e.g. interruptions to your home internet connection).

Extensions

All requests for extensions must follow the faculty guidelines and submit the required forms as soon as possible. In addition, when emailing for an extension, please remember to include all code completed until that time. This assignment cannot be completed in a few days and requires students to apply what we learn each week as we move closer to the submission date.

Plagiarism

Cheating and plagiarism are viewed as serious offences. In cases where cheating has been confirmed, students have been severely penalised, from losing all marks for an assignment, to facing disciplinary action at the Faculty level. Monash has several policies in relation to these offences and it is your responsibility to acquaint yourself with these.

Plagiarism (http://www.policy.monash.edu/policy-bank/academic/education/conduct/plagiarism-policy.html)

Lastly, your assignment will be subjected to a similarity check against all current and past assignment submissions. Should your submission be flagged as highly similar to that of another student, both submission grades will be withheld and considered as per the university's policy on plagiarism and collusion. **All work submitted MUST be your own work. Do not share or borrow anything.**