

SELF-DIRECTED TASK for CN4001 students

ASSESSED TASK: 18 marks

Consider the following UML design for a **Product** class:

Product
-name: String -stockLevel: int -price: double
+Product(String, int, double) +buyStock(int): double +sell(int): boolean +setPrice(double) +getName() : String +getStockLevel() : int +getPrice(): double

The **attributes** of this class allow details of a given product (name, stock level and price) to be stored. The **methods** of this class should behave as follows:

+Product(String, int, double)	The <i>constructor</i> receives three parameters and uses them to initialise the name of the product, the number of items initially stocked and the price per item respectively.
+buyStock(int): double	Receives the number of new items of stock bought, updates the total value of stock and returns the value of stock bought.
+sell(int) : boolean	Receives the number of items sold, reduces the stock level accordingly (first checking to see if enough items are in stock) and returns true if the sale was successful and false otherwise.
+setPrice(double)	Receives and sets a new price for the product.
+getName() : String +getStockLevel() : int +getPrice(): double	These methods return the values of the respective attributes.

The Task

- a) Implement the **Product** class.

(3 marks)

- b) Develop a **ShopApp** program that holds **5 Product objects in an array**, representing products sold in a shop. The **ShopApp** program should provide a allow users to **display** all products, **buy** stock, **sell** stock, **re-price** an item of stock and display the **total value** of all stock in the shop.

Your student number should be displayed at the top of the interface and extra credit will be given for developing a **menu driven interface**, for some **input validation** and the use of **methods**.

(7 marks)

*Note, you can tackle a simpler task to display, buy, sell and re-price a **single Product** object (rather than 5 Product objects in an array) – but this can only earn a maximum of **3 marks**.*

- c) Once the **ShopApp** program is complete, fill in the program review table on the following page.

(8 marks)

. THIS IS AN INDIVIDUAL TASK

You should upload **only the following three files**:

- **Program review table**
- **Product.java** file
- **ShopApp.java** file

to Moodle (using your group's submission link) by **Thursday 19th December 2024** (no later than 4:00 pm).

Student ID:		
Programming Concept (1 mark each = 8 marks)	Example code in ShopApp.java	Comment
variables		
input/output		
choices		
loops		
input validation		
methods		
arrays		
classes/objects		