

**Course Title and ID :**

420-P16-AS Structured Programming

**FINAL PROJECT –**

EXERCISE 2 – C++

Arrays–User-defined functions–Pointers–Structures

**Instruction to submit your work –**

* Submit 2 files, as indicated below:

1) A .cpp file, named significantly, including the code of your program

(e.g. team1\_finalProject\_code\_exercise1.cpp)

2) The section 2 of is document, completed with appropriate text and screenshots, and named significantly

(e.g. team1\_finalProject\_report\_exercise1.pdf)

.

* Respect the deadline indicated.

Ask your questions, when there are about this project exercise, within the MS Teams channel DISCUSSION FORUM. Like this, the answers will be shared once with the whole class.

**General Technical Instructions**

* Create a **GitHub repository** and give access to all the teammates to it to make sure everyone will be able to access and complete the code in real time.
* **Arrays, User-defined functions, Pointers, Structures,** are the main toolto use in this program. However, do not use procedural programming.
* Use **loops** and **conditions** to write and read from arrays and perform comparisons. Do not write the indexes manually. E.g. array[i] instead of array[0].
* Do not use the instruction "**goto**" to navigate from one section of the program to another one because it has a bad reputation of creating "spaghetti code".
* Do not use **“object oriented programming”** because we didn’t learn it yet.
* Declare **variable names that are significant** and written in lower cases
* Use **the most appropriate data type** (e.g. int, float, char, string, struct) for each variable, in accordance with the context described.
* Declare the variables that are **constant as C++ constants and write them using Capital letters**
* Add **indents** to the code at the appropriate places
* Include **a comment in the beginning of the program** that describes he problem it solves and at least the **4 comments below at the correct place**:

*// Declare variables and constant and initialize*

*// Invite, Read, Save and Validate Inputs*

*// Calculate*

*// Display Outputs*

* **Test your program** with the data provided in the given screenshots. However, your program should not be able to work with only the test data, it must be able to work with other data. Consequently, use appropriate variables and constants to **create a complete and flexible program** that can evolve easily (e.g. the removal of a current product or the addition of a new product).

**Statement –**

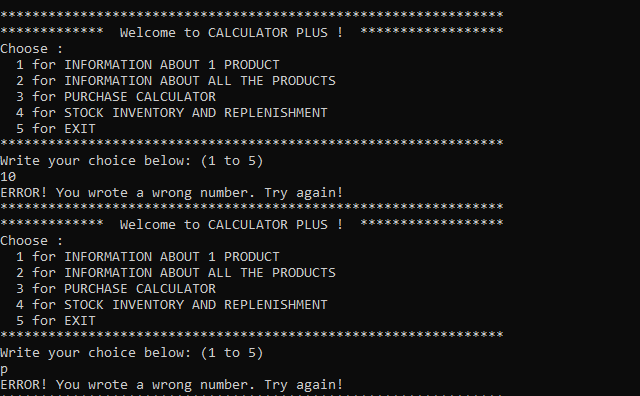
The maple syrup producer named Sirop d’Érable Montréal assigns a new contract to the company you are working for. This producer markets 4 dimensions of maple syrup bottles, as described below:

|  |  |  |
| --- | --- | --- |
| **Dimension** | **Quantity in ml**  **To be used as a product code** | **Unit price without taxes in Canadian dollars** |
| Small | 50 ml | $15.99 |
| Medium | 100 ml | $24.99 |
| Big | 250 ml | $49.99 |
| Extra | 500 ml | $89.99 |

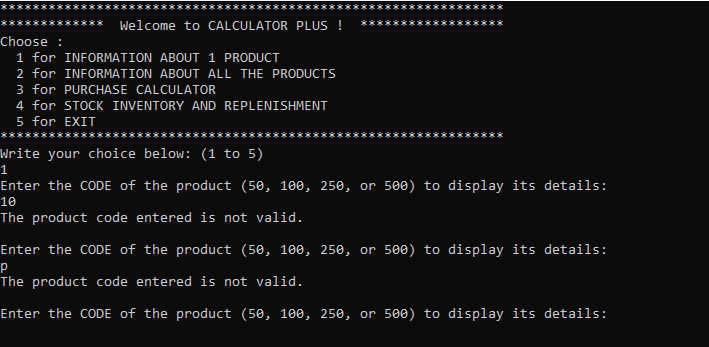
Accordingly, your supervisor receives the mission to develop a calculator allowing the cashier clerk of Sirop d’Érable Montréal to quickly calculate and/or display different information regarding the purchases and stock evolution. Then, your supervisor asks your colleagues and you to create a prototype to demonstrate the basic operation of this calculator.

Observe the operations of the program CALCULATOR PLUS shown in the screenshots below to find the features to include within the calculator you will develop. And ask your supervisor your additional questions when there are.

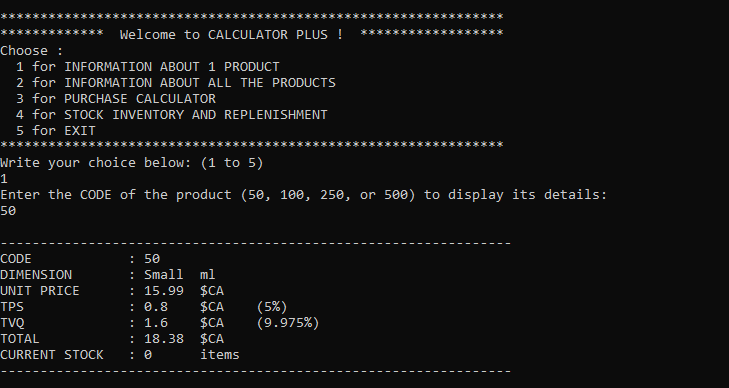
1. Entry. Validate the menu choice. Check that the program accepts only 1, 2, 3, 4, and 5.

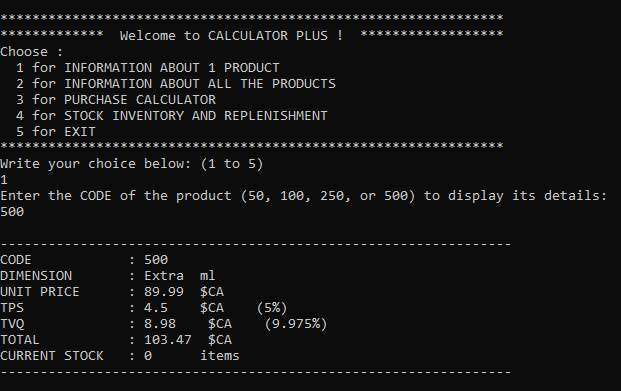


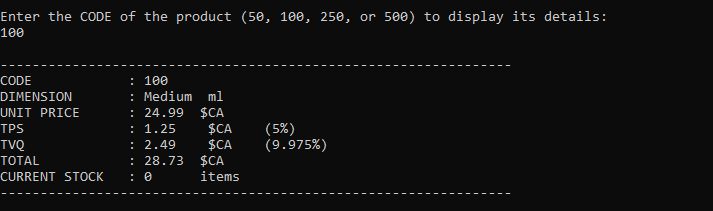
1. Option 1. Validate the product code. Check that the program accepts only 50, 100, 250, and 500.

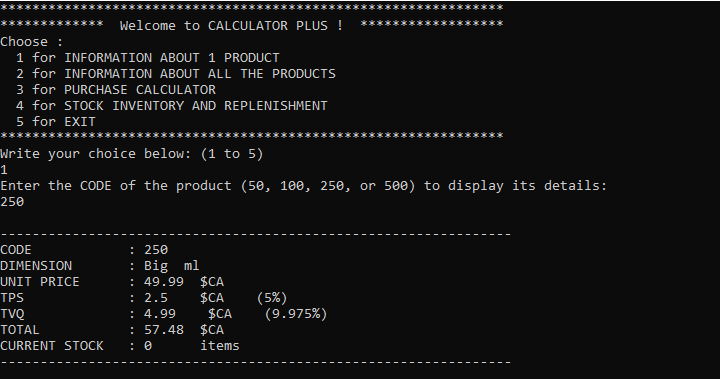


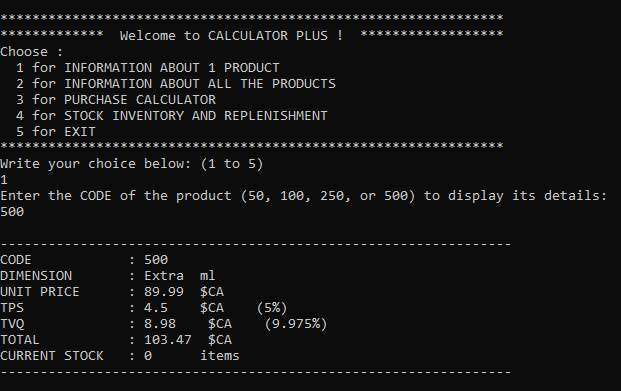
1. Option 1. Check the result for product code 50.
2. Option 1. Check the result for product code 100.
3. Option 1. Check the result for product code 250.
4. Option 1. Check the result for product code 500.



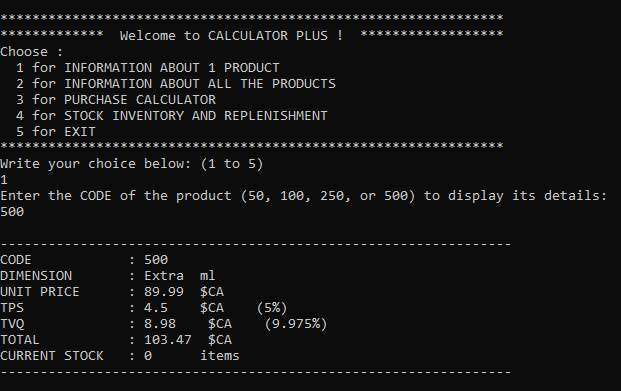


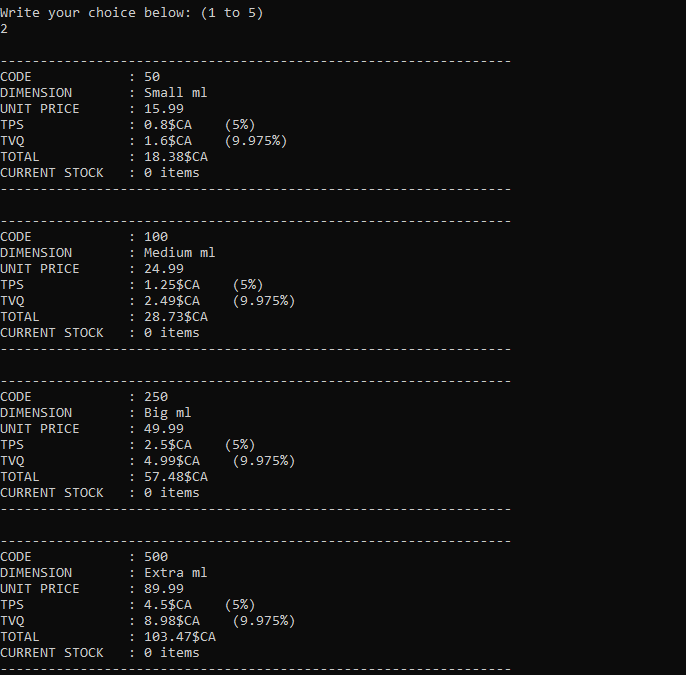




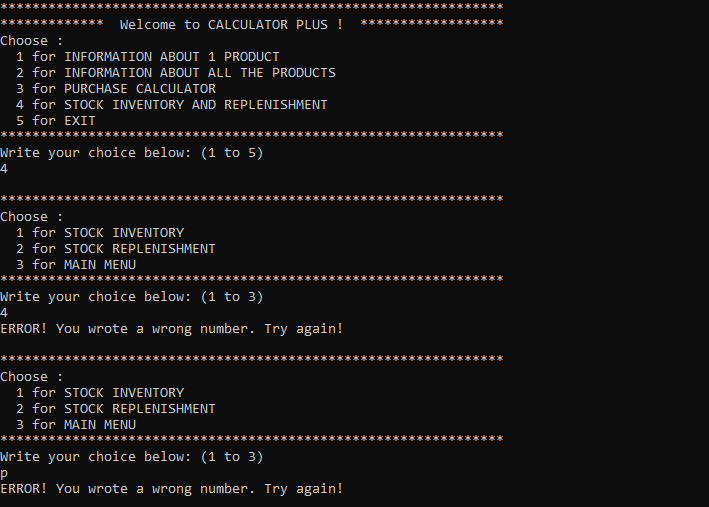


1. Option 2. Check the result.

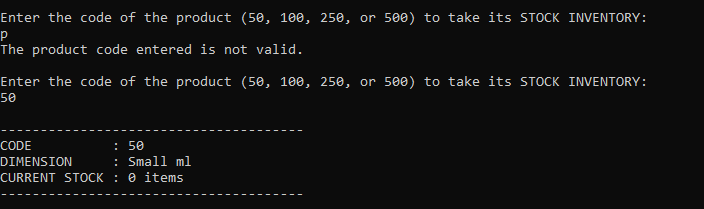




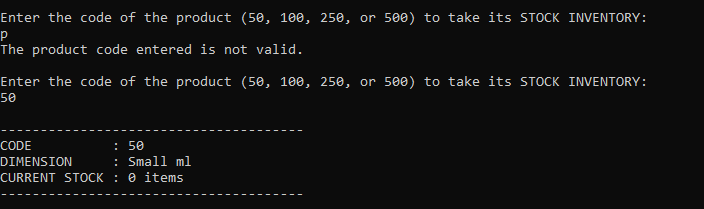
1. Option 4. Validate the sub-menu choice. Check that the program accepts only 1, 2, and 3.

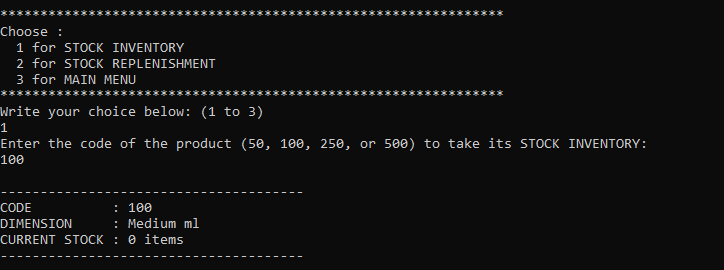


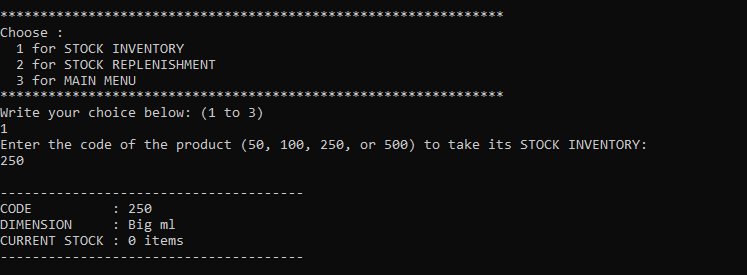
1. Option 4. Submenu Option 1. Validate the product code. Check that the program accepts only 50, 100, 250, and 500.

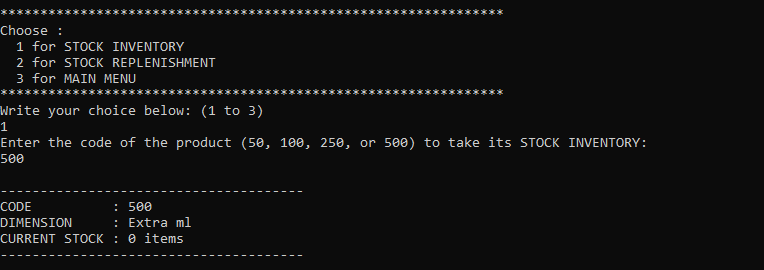


1. Option 4. Submenu Option 1. Check the result for product code 50.
2. Option 4. Submenu Option 1. Check the result for product code 100.
3. Option 4. Submenu Option 1. Check the result for product code 250.
4. Option 4. Submenu Option 1. Check the result for product code 500.

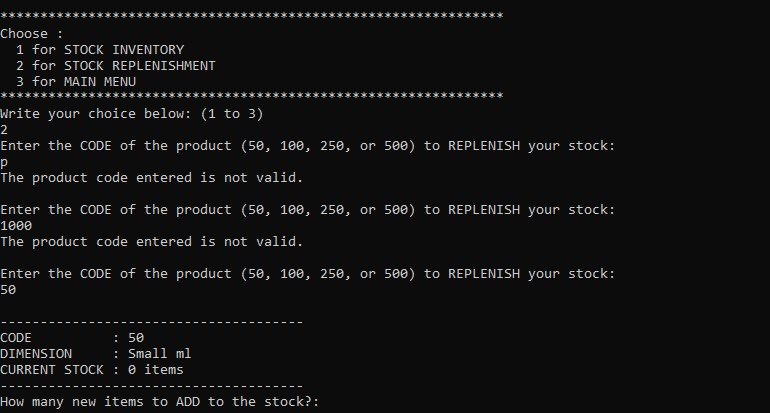




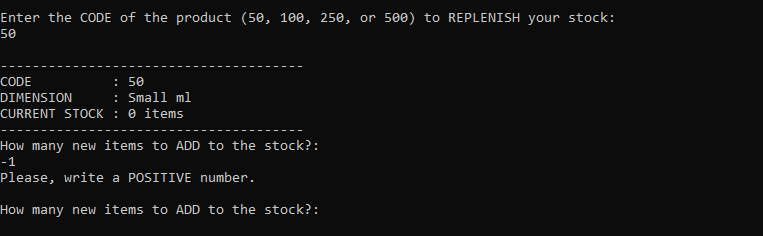




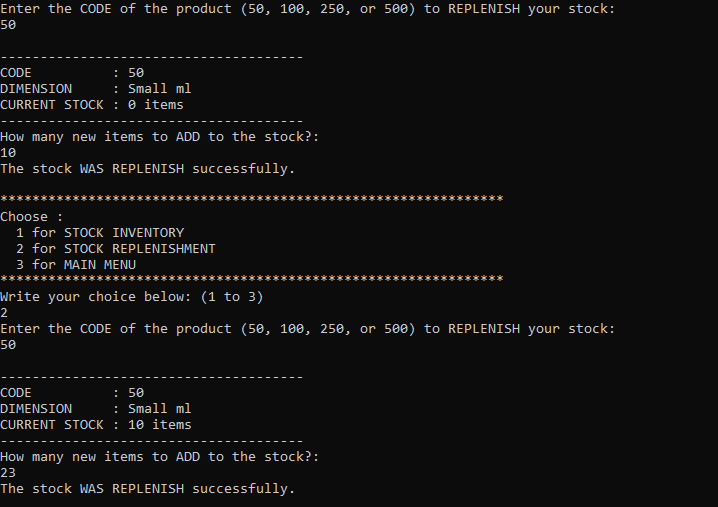
1. Option 4. Submenu Option 2. Validate the product code. Check that the program accepts only 50, 100, 250, and 500.

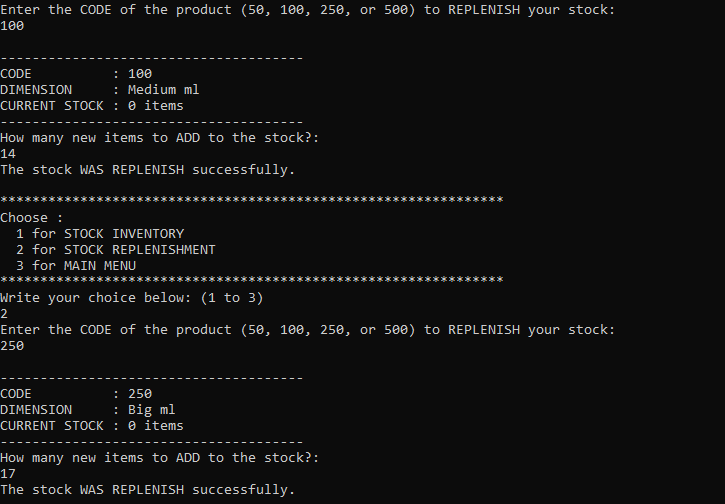


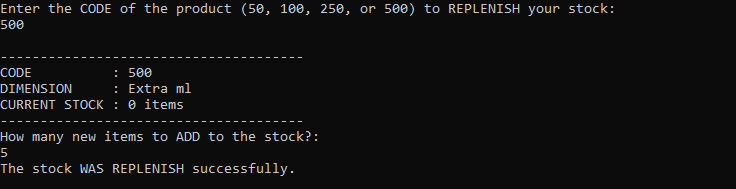
1. Option 4. Submenu Option 2. Validate the quantity added the stock. Check that the program accepts only positive numbers.



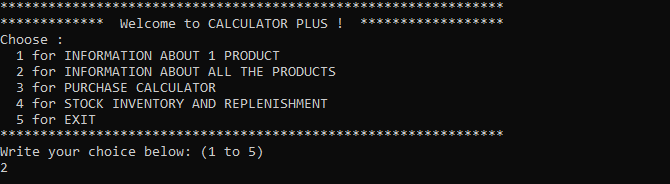
1. Option 4. Submenu Option 2. Add 10 products 50.
2. Option 4. Submenu Option 2. Add 23 products 50.
3. Option 4. Submenu Option 2. Add 14 products 100.
4. Option 4. Submenu Option 2. Add 17 products 250.
5. Option 4. Submenu Option 2. Add 5 products 500.

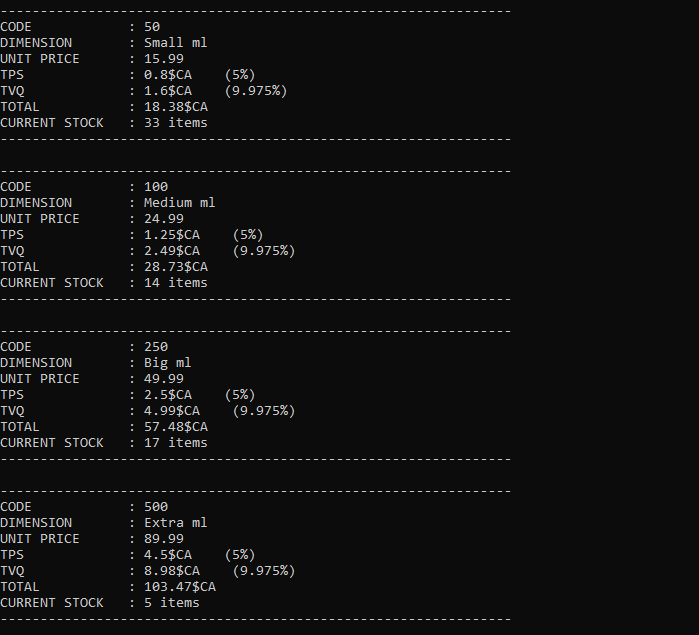




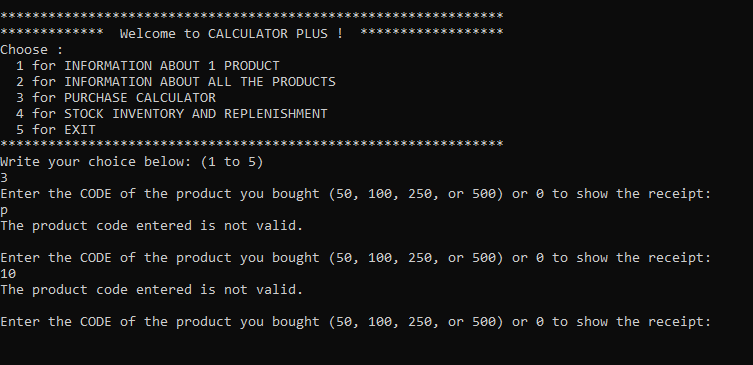


1. Option 2. Result. Check that the stock was effectively replenish with 33 products 50, 14 products 100, 17 products 250, and 5 products 500.

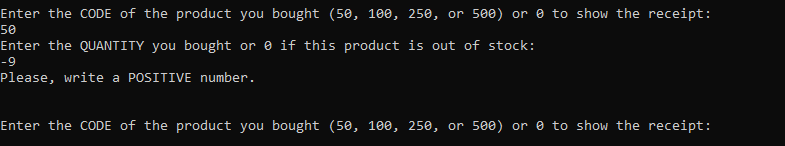




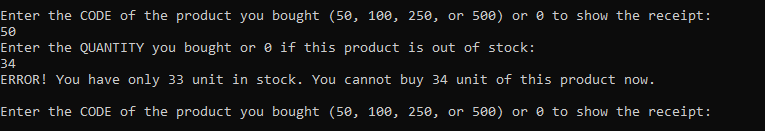
1. Option 3. Check that the program accepts only 50, 100, 250, and 500.



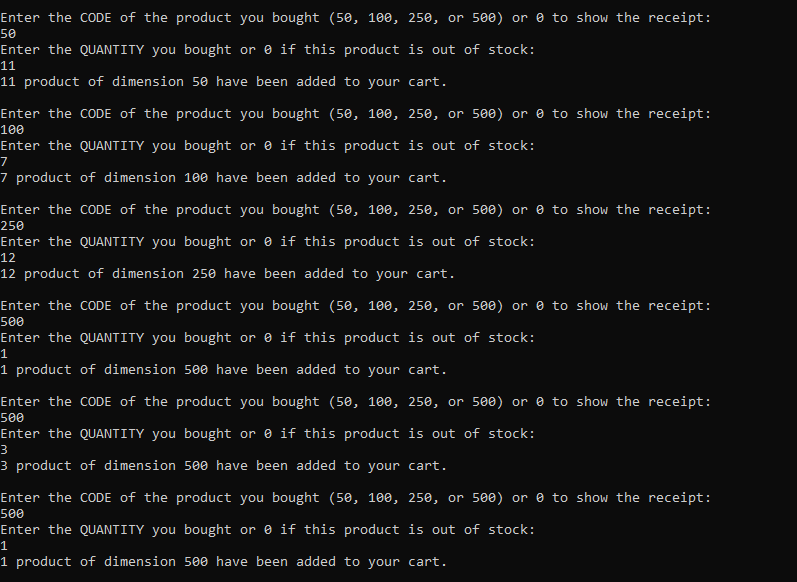
1. Option 3. Validate the quantity of items bought. Check that the program accepts only positive numbers.



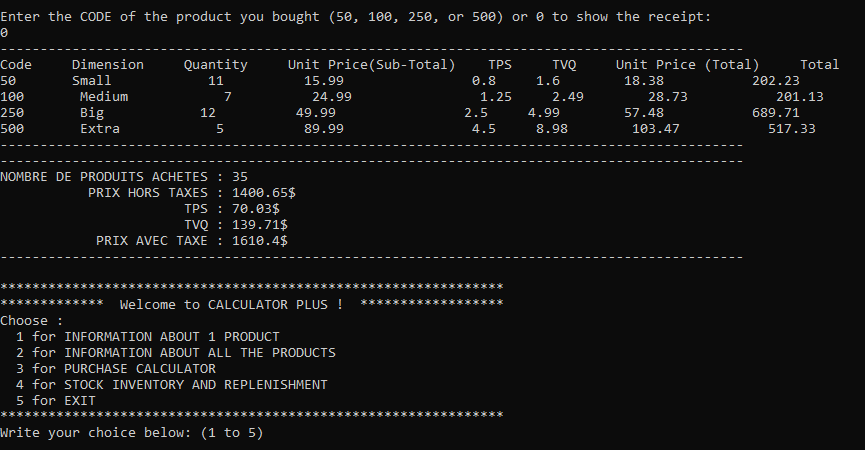
1. Option 3. Validate the quantity of items bought. Check that the program accepts only positive numbers that are lower than or equal to the number of items included within the stock.



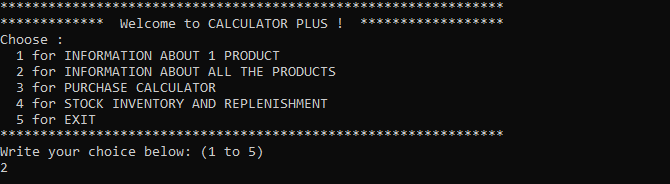
1. Option 3. Buy 11 products 50.
2. Option 3. Buy 7 products 100.
3. Option 3. Buy 12 products 250.
4. Option 3. Buy 1 product 500.
5. Option 3. Buy 3 products 500.
6. Option 3. Buy 1 product 500.

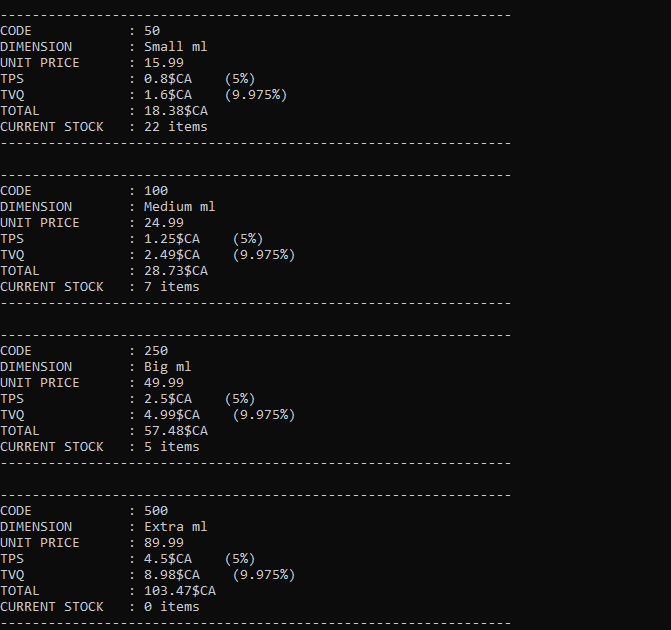


1. Option 3. Display the result.



1. Option 2. Result. Check that the stock was update and include now 22 products 50, 7 products 100, 5 products 250, and 0 products 500.





**EVALUATION GRID –**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Aspect considered | Answer Quality and Weighting | | | **TOTAL** |
| *Excellent*  *or*  *Complete* | *Acceptable*  *or*  *Incomplete* | *Absent*  *or*  *Incorrect* |
| Variables (quantity, name, type, declaration and initialization) and appropriate use of the notions learned | 16% | 15% to 9% | 0% | **16%** |
| 32 Functionalities | 64% | 63% to 2% | 0% | **64%** |
| Communication (level of relevance of the new messages displayed or correct use of the messages provided) | 10% | 9% to 6% | 0% | **10%** |
| Code structure (indents, comments, line of code, no empty space, switch, list... ) | 10% | 9% to 6% | 0% | **10%** |
| TOTAL | | | | **100%** |

*\*\* This exercise will count on 75% of the final project grade.*

COMPLETE AND SUBMIT THE FOLLOWING SECTION

BEFORE THE ORAL PRESENTATION OF YOUR PROJECT

**Course Title and ID :**

420-P16-AS Structured Programming

**FINAL PROJECT –**

EXERCISE 2 – C++

Arrays–User-defined functions–Pointers–Structures

**Team Member 1**

First Name: Robert

Last Name: Littlejohn

**Team Member 2**

First Name: Skander

Last Name: Ben Hassen

**Team Member 3**

First Name: Finley James

Last Name: Mitchell

**TEST SCENARIOS**

Execute your program with the data given for each scenario below, and add at least 1 screenshot including both the data input and the data output to demonstrate the way your program responds.

1. Entry. Validate the menu choice. Check that the program accepts only 1, 2, 3, 4, and 5.
2. Option 1. Validate the product code. Check that the program accepts only 50, 100, 250, and 500.
3. Option 1. Check the result for product code 50.
4. Option 1. Check the result for product code 100.
5. Option 1. Check the result for product code 250.
6. Option 1. Check the result for product code 500.
7. Option 2. Check the result.
8. Option 4. Validate the sub-menu choice. Check that the program accepts only 1, 2, and 3.
9. Option 4. Submenu Option 1. Validate the product code. Check that the program accepts only 50, 100, 250, and 500.
10. Option 4. Submenu Option 1. Check the result for product code 50.
11. Option 4. Submenu Option 1. Check the result for product code 100.
12. Option 4. Submenu Option 1. Check the result for product code 250.
13. Option 4. Submenu Option 1. Check the result for product code 500.
14. Option 4. Submenu Option 2. Validate the product code. Check that the program accepts only 50, 100, 250, and 500.
15. Option 4. Submenu Option 2. Validate the quantity added the stock. Check that the program accepts only positive numbers.
16. Option 4. Submenu Option 2. Add 10 products 50.
17. Option 4. Submenu Option 2. Add 23 products 50.
18. Option 4. Submenu Option 2. Add 14 products 100.
19. Option 4. Submenu Option 2. Add 17 products 250.
20. Option 4. Submenu Option 2. Add 5 products 500.
21. Option 2. Result. Check that the stock was effectively replenish with 33 products 50, 14 products 100, 17 products 250, and 5 products 500.
22. Option 3. Check that the program accepts only 50, 100, 250, and 500.
23. Option 3. Validate the quantity of items bought. Check that the program accepts only positive numbers.
24. Option 3. Validate the quantity of items bought. Check that the program accepts only positive numbers that are lower than or equal to the number of items included within the stock.
25. Option 3. Buy 11 products 50.
26. Option 3. Buy 7 products 100.
27. Option 3. Buy 12 products 250.
28. Option 3. Buy 1 product 500.
29. Option 3. Buy 3 products 500.
30. Option 3. Buy 1 product 500.
31. Option 3. Display the result.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Code | Dimension | Quantity | Unit Price (Sub-Total) | TPS (5%) | TVQ (9.975%) | Unit Price (Total) | Total |
| 50 | Small | 11 | 15.99 | 0.80 | 1.60 | 18.38 | 202.23 |
| 100 | Medium | 7 | 24.99 | 1.25 | 2.49 | 28.73 | 201.13 |
| 250 | Big | 12 | 49.99 | 2.50 | 4.99 | 57.48 | 689.71 |
| 500 | Extra | 5 | 89.99 | 4.50 | 8.98 | 103.47 | 517.33 |
|  |  |  |  |  |  |  |  |
| NOMBRE DE PRODUITS ACHETES | | | 35 |  |  |  |  |
| PRIX HORS TAXES | | | 1400.65 |  |  |  |  |
| TPS (5%) | | | 70.03 |  |  |  |  |
| TVQ (9.975%) | | | 139.71 |  |  |  |  |
| PRIX AVEC TAXES | | | 1610.40 |  |  |  |  |

1. Option 2. Result. Check that the stock was update and includes now 22 products 50, 7 products 100, 5 products 250, and 0 products 500.