CIS 246 – C++ Programming

Program:	**Week 04**
Points:	**20**
Chapter(s):	**4 & 5**
File(s) to Submit:	**Product.h, ProductDriver.cpp (within a zip)**

Description

Summary: Write a C++ application that calculates total retail values for a series of sales transactions input by the user.

Description: A mail-order house sells five different products whose retail prices are as follows:

Product 1	$2.98
Product 2	$4.50
Product 3	$9.98
Product 4	$4.29
Product 5	$8.67

Your application must read a product number and quantity sold for each transaction from stdin, compute the total value of that sale, and add the value of the sale to a grand total for that product.

After all data has been input, display the total value of each of the five inventory items.

You must write **a driver and a class** using separate files for full credit.

Your program **must use a switch structure** to determine which product's sales to update.

Your program **must use a sentinel-controlled loop** to determine when the program should stop looping and display the results.

Learning Objectives

In this assignment, you will practice:

- Using switch selection control structures
- Using a sentinel-controlled repetition structure
- Creating a driver program and a class
- Calling member functions.
- Displaying neatly formatted output to the screen using C++ syntax

Requirements for the Product Class

Data Members

1. product price (in dollars)
2. total sales for the product (in dollars)

Be sure to:

- use the appropriate data type for each data member
- use the appropriate access modifier for each data member
- initialize each data member using an in-class initializer

Member Functions

1. **One 1-argument constructor** – The constructor uses one parameter representing the product price. Set the value of the data member using this argument.
2. **Getters and setters for each data member**
3. **updateTotalSales function**
   a) This function uses one parameter (representing the quantity sold) and does not return a value. Be sure to use the appropriate data type for the parameter and the appropriate access modifier for the function.
   b) It computes the total value of the current sale by multiplying the argument passed in by the product's price. Call the get function for the price to retrieve the product's price.
   c) It updates the product's totalSales data member by adding the value calculated in step b to the product's existing totalSales. Call the getter for totalSales to get the existing value, and the setter for totalSales to set the new value.
   d) You must use the getter and setter functions to access the data members. Do not access the data members directly.

Requirements for the ProductDriver Program

main Function

1. Instantiate five separate objects of the class Product, passing the appropriate argument to each constructor. Remember, the Product class constructor requires the price of that instance of the product. The prices can be hard-coded numbers. You do not have to ask the user to type them in.
2. Using a **sentinel-controlled loop**:
   a. Read the product number and quantity sold for one transaction.
   b. Use a **switch statement** to determine which Product object's sales to update
   c. Call the appropriate Product object's updateTotalSales function.
   d. Continue the loop until the user types a **0** for the product number.

3. After the loop is complete, display each Product object's total sales, including the dollar sign, to two decimal places with a field width of 10.

Sample Run

```
Product number: 1
Quantity sold: 5000
Product number: 2
Quantity sold: 33
Product number: 1
Quantity sold: 100
Product number: 5
Quantity sold: 10
Product number: 5
Quantity sold: 65
Product number: 3
Quantity sold: 28
Product number: 4
Quantity sold: 56
Product number: 1
Quantity sold: 10
Product number: 0

Total Sales
===========
Product 1: $  15227.80
Product 2: $    148.50
Product 3: $    279.44
Product 4: $    240.24
Product 5: $    650.25
```

Requirements for Full Credit on This Project

SUBMIT YOUR OWN WORK – Plagiarism is not tolerated in this course. Please review the section on the Academic Honor Code in the syllabus. I will not hesitate to drop you from this class if you submit a program that is plagiarized.

COMPLETE AND ACCURATE – Your program must compile, execute, and give accurate output.

FOLLOW ALL REQUIREMENTS ACCORDING TO THE INSTRUCTIONS – Follow the instructions as written for completing this project, even if you [think you] know a "better" way to do something.

COMMENTS – Include comments in your code. There must be a comment at the top of each source code or header file that includes your name, the assignment number, and a description of the code in that file. There must be comments at each important step in your algorithm that describes that step.

FOLLOW BEST PRACTICES – Follow best practices in C++ programming as discussed in class, including, but not limited to, appropriate use of private/public, appropriate use of classes and/or header files, sets & gets, white space, alignment, meaningful variable names, naming conventions, using statements, etc. Points will be deducted for sloppy code that is hard to read, even if it works, so pay attention to these details.

SUBMIT ALL FILES BEFORE THE DUE DATE – Submit a .zip of ONLY source code files to the dropbox for this assignment on Canvas before the due date. Do not submit anything except .cpp and/or .h, within a zip. Do not submit .exe files. Do not submit a folder structure. Do not submit project files from an IDE.