Graded lab 1B - 18:15-19:45 (8 points)

In this task you will be implementing a class for representing credit cards and performing various operations on them. You need to create the CreditCard.h and CreditCard.cpp files from scratch. Each part contains a description of what needs to be implemented as well as example usages of this particular functionality in the main.cpp file. The main.cpp file cannot be modified (apart from uncommenting the code) and the functions you implement should fit the example usages in it. Given the same input your code should produce the same output as provided in the output.txt file.

After finishing the task make sure to upload your solution to Teams as a single .zip file. The .zip file should contain only .h and .cpp files.

Part 1 (1 point)

Create the CreditCard.h file and define the CreditCard class inside of it. The class should contain private fields representing the following properties:

- card number (array of characters; you can assume that all of the numbers contain exactly 16 digits)
- expiration date month (int)
- expiration date year (int)
- current balance (int)
- the name of the last bought item (array of characters; you can assume that items are no longer than 30 characters)

You should also create a CreditCard.cpp file where you will put all of the implementations of functions defined in further parts of the task.

Part 2 (2 points)

Implement two constructors for the CreditCard class:

- Both take the card number and its balance as arguments (balance has a default value of 100). The card number is represented by a string that contains 16 digits with possible other characters in-between. Both constructors should copy only the digits from the argument to the array inside of the object.
- The first one also takes and expiration date written in the format "MM/YYYY" and sets the last bought item to "".
- The second one copies the expiration date and last bought item form the card given as an argument.

Part 3 (1.5 points)

Implements the output and input operators:

- The output operator's result should match the format of the output from the output.txt file.
- The input operator should read two integers from the user representing the month and year of a new expiration date. If the date is later than the current expiration date it overrides the current one (otherwise nothing happens).

Part 4 (1 point)

Implement the less than (<) and greater or equal (>=) operators for credit cards. The operators should compare the cards by their expiration date.

Part 5 (1.5 points)

Implement a public function for buying items with the card. The function should have two versions:

- The first one takes an item as an argument, in a form of a const char* and buys it using the card. The price of an item is determined by the length of the name. Remember to update the last item bought.
- The second version takes an integer i as an argument and buys the last bought item i times.

Part 6 (1 point)

Implement two versions of the += operator for credit cards.

- The first one takes an integer i as an argument and moves the expiration date i months forward.
- The second one takes a card as an argument and copies its expiration date provided that it's later than the current one.