Homework 1: I/O, arithmetic expressions, formatting, etc. **Due Date:** 9/11/20

Suppose you have amounts of Mexican pesos and Euros that you would like to convert to U.S. Dollars to find out how much money they are worth in total. Write the algorithm (steps in pseudocode) and the corresponding program to convert Mexican Pesos and Euros to U.S. Dollars and display their equivalent and the total on the screen.

The program must prompt the user to enter the amount of Mexican Pesos and Euros (**double** precision real numbers) to be converted and then display the result in the format shown below. The amounts in U.S. Dollars (**double** precision real numbers) must be **rounded off** to the **cents** (second decimal digit).

Assume that the amounts in Mexican Pesos and Euros will be entered with no more than two decimal digits (cents).

For the conversion assume that:

```
1 U.S. Dollar = 18.62 Mexican Pesos
1 U.S. Dollar =  0.92 Euros
```

Solution requirements:

1) Declare named constants **DOL2MEX** (18.62) and **DOL2EURO** (0.92) for working with these values in your program. Declare variables **pesos** and **euros** to store the amounts in Mexican pesos and Euros entered by the user. Be sure of choosing the appropriate data types for them.

2) Use the **floor()** function to round the values.

After rounding the values in **U.S. Dollars** to the cents you must add them to get the total and then **convert** each of them to **two separate whole numbers**, one corresponding to the whole part (bills) and the other corresponding to the decimal digits (coins). The numbers MUST be **converted** to whole numbers, not just displayed as whole numbers. Since you work with real numbers you should round them to get the correct amount in cents. Use the following identifiers for the variables that will hold the separated amounts:

- **wholedollarsp**: stores the whole dollars of the amount obtained from converting the pesos
- **centdollarsp**: stores the cents of the amount obtained from converting the pesos
- **wholedollarse** stores the whole dollars of the amount obtained from converting the euros
- **centdollarse**: stores the cents of the amount obtained from converting the euros
- **wholetotal**: stores the whole dollars of the total amount
- **centtotal**: stores the cents of the total amount

Note: you can declare other variables if necessary.

3) You **can** use ONLY the material learned and used in the first 6 lab assignments.

4) Your program must pass the 4 tests shown at the end of this handout.

5) Format the output to display **real** numbers in fixed format with 2 decimal digits. Pay attention to the format of the output (especially the alignment of columns).

6) After asking for the input your program must clear the screen and show the output. Pay attention to the titles. For Windows use system("cls"), for Mac use system("clear").

7) Since system("pause") works only on Windows, use the lines of code shown below to pause your program:

```
// This is to pause the execution of the program
cout << "Press Enter to continue ...";
cin.sync();
cin.ignore();
cin.get();
```

The program must compile without errors or warnings.

Create a project if necessary (or use one that is already open) and **add the existing item** named **hw1_TXX.cpp** (provided with this handout).

Enter your algorithm (as comments) and implement it in C++.

Note:

- Do NOT remove or modify the statements that I use to test certain things in your program.
- Run my sample solution to know how your program must behave. Pay attention to the input and the output formats. Your solution must behave exactly like mine.
- Carefully analyze the sample run shown below and use it as a reference to ensure you do the right things.
- Try the values for pesos and euros specified in the four tests and check if you get the right result (compare with my solution). If you get an error message on the output, read the comment on the line specified in the message to find out what is wrong.

Sample run of the program

```
Mexican Peso and Euro to U.S. Dollar converter

Please enter the amount of Pesos: 1345.13
Please enter the amount of Euros: 187.56
```

Input screen

After clearing the screen:

```
Mexican Peso and Euro to U.S. Dollar converter


  1345.13 pesos:        72 US dollars with 24 cents

   187.56 euros:       203 US dollars with 87 cents

          Total:       276 US dollars with 11 cents
```

Output screen

Review the examples discussed in class, the lab assignments done so far, and the textbook to get an idea of what you need to do. The **algorithm** must be written in **pseudocode** and should look like my lab handouts. **Include your algorithm in the source code as comments.**

Do not hesitate to use the corresponding topic in Discussions to post your questions/doubts about this assignment. I will reply as soon as I can.

IMPORTANT:

You must submit ONLY ONE solution per team.

Your program must be well commented, use meaningful identifiers, use named constants, and use indentation as shown in the textbook.

Your program must have the following comments at the top:

```
//*********************************************************************
// Team #            CSCI/CMPE 1370          Spring 2020          Homework # 1
// First and Last Name
// First and Last Name
//  Using your own words, write here a description of what the program does.
//
//*********************************************************************
```

Include the names of both teammates only if both participated in the solution, otherwise just enter your name.

Please name your source code file **hw1_TXX.cpp** (where XX are two digits indicating your team number as in hw1_T03.cpp).

When done, submit your solution through Blackboard using the "Assignments" tool. Do NOT email it.

Grading criteria

You start with 100 points and then lose points as you don't do something that is required.

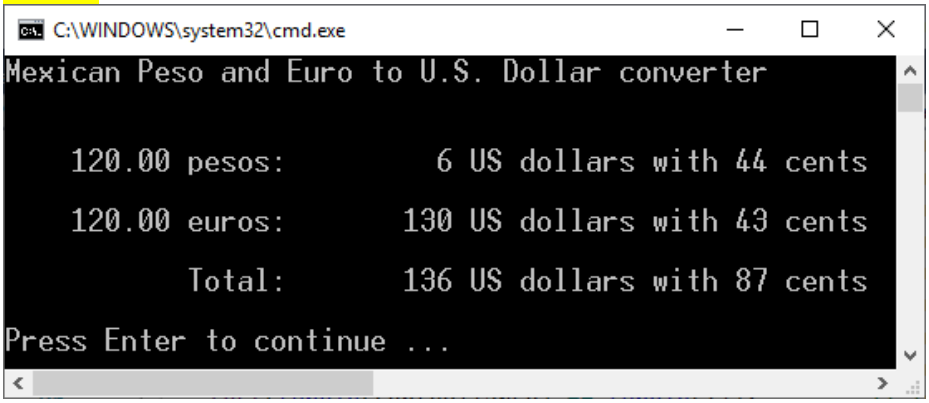
```
-10: Too few/no comment (no description of what the program does).
-10: Didn't use named constants.
-10: Data type is not correct.
-10: Mixed data types in expressions.
-10: Incorrect output format.
-20: Didn't round off.
-10: Incorrect rounding.
-10: Did not convert the amounts to whole values.
-10: Incorrect expression used to calculate the cents of U.S. Dollars
-10: Did not pass test. (each)
-30: Missing/incorrect algorithm.
-15: Poor quality algorithm.
 -5: Incorrect file name (should be hw1_TXX.cpp)
-50: Incomplete program.
-50: Program does not compile.
-100: No submission.
-10: Late submission.
```

Important: more points may be lost for other reasons not specified here.

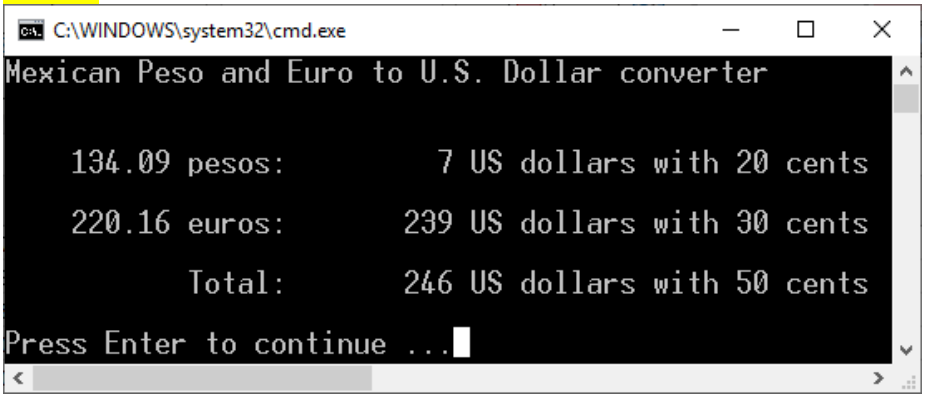
Sample runs of the program:

Note: I am showing just the output generated (you can see the values that were inputted from it)

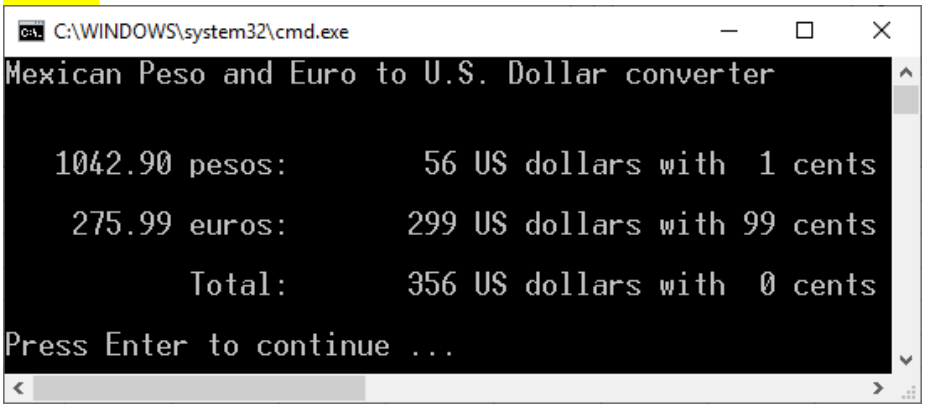
Test 1



Test 2



Test 3



Test 4