PIC 10A Section 1 - Homework # 2 (due Monday, January 17, by 11:59 pm)

You should upload each .cpp (and .h file) separately and submit them to CCLE before the due date/time! Your work will otherwise not be considered for grading. Do not submit a zipped up folder or any other type of file besides .h and .cpp.

Be sure you upload files with the precise name that you used in your editor environment otherwise there may be linker and other errors when your homeworks are compiled on a different machine. Also be sure your code compiles and runs on Visual Studio 2022.

COUNTING CHANGE THE CANADIAN WAY

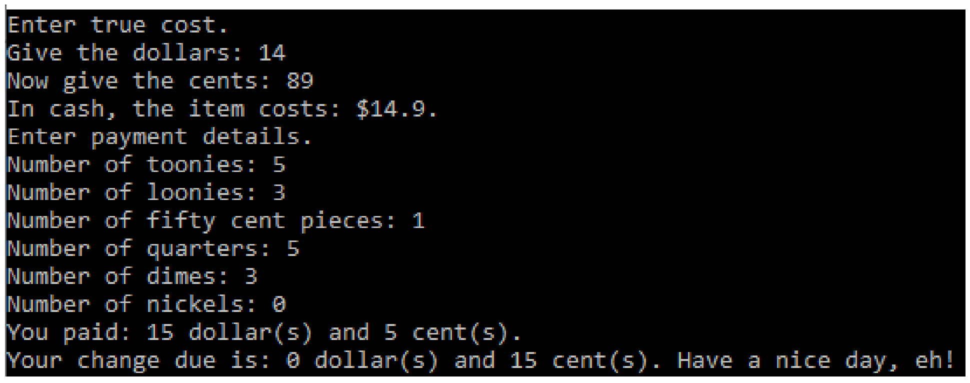
This homework will give some practice in working with variables and performing simple arithmetic. You should submit a single file **change.cpp**.

In this homework, we will write a friendly Canadian cash register program. It will ask a user for the cost of an item (dollars and cents) and then compute the cost in cash (in Canada, when one pays cash, costs are always rounded to the nearest nickel since we got rid of the penny). The program then prompts a user for their payment: the number of toonies (two-dollar coins), loonies (one-dollar coins), fifty-cent pieces, quarters, dimes, and nickels they have. Then it will print the total value they have paid, their change due, and finish with wishing them a nice day. See the screenshot for a demo.

You might not have seen a toonie before, but they are real, and anyone who possesses such a coin obtains magical powers. No one really uses fifty-cent pieces, but they still exist...

Please refer to the syllabus for how the work will be graded. The program should execute in the following manner. You must make sure the output matches the format below perfectly.

Enter true cost.
Give the dollars: [USER ENTERS NUMBER FOR DOLLARS OF ITEM]
Now give the cents: [USER ENTERS NUMBER FOR CENTS]
In cash, the item costs: $[PROPER COST AFTER ROUNDING TO THE NEAREST 5-CENTS].
Enter payment details.
Number of toonies: [USER ENTERS NUMBER]
Number of loonies: [USER ENTERS NUMBER]
Number of fifty cent pieces: [USER ENTERS NUMBER]
Number of quarters: [USER ENTERS NUMBER]
Number of dimes: [USER ENTERS NUMBER]
Number of nickels: [USER ENTERS NUMBER]
You paid: [CORRECT NUMBER OF DOLLARS] dollar(s) and [CORRECT NUMBER OF CENTS] cent(s).
Your change due is: [CORRECT NUMBER OF DOLLARS] dollar(s) and [CORRECT NUMBER OF CENTS] cent(s). Have a nice day, eh!



```
Enter true cost.
Give the dollars: 14
Now give the cents: 89
In cash, the item costs: $14.9.
Enter payment details.
Number of toonies: 5
Number of loonies: 3
Number of fifty cent pieces: 1
Number of quarters: 5
Number of dimes: 3
Number of nickels: 0
You paid: 15 dollar(s) and 5 cent(s).
Your change due is: 0 dollar(s) and 15 cent(s). Have a nice day, eh!
```

So just to be clear: the true cost of the item was $14.89. But as a cash transaction, it is treated as $14.90.

Important remarks: You may assume:

- the user always enters valid answers without any typos and
- they will always pay at least as much cash as the item cash value. Thus, their change due will always be bigger than or equal to $0.
- Also: don't worry about the singular vs plural for the dollars and cents. Note the parentheses. We haven't studied control flow yet so you are not expected to resolve those details.

Important hints:

- A large part of this course (and programming) amounts to splitting a project into multiple tiny, manageable chunks. So start with the basics:
  (i) can you write a program that asks for the number of dollars and cents and then prints those same values out, without any rounding?
  If you can do that, (ii) try to write a program to accept the number of toonies, loonies, etc., and see if you can correctly calculate the total amount in dollars and cents.
  Only once you can accomplish both (i) and (ii) above, you can start to deal with rounding the prices, calculating differences, etc. Do this one step at a time.

- To do the rounding to the nearest nickel: calculate the fractional number of nickels present and then round. For example, with $14.89, the 89 cents amounts to 17.8 nickels. And 17.8 can easily be rounded to 18 nickels, which is 90 cents.

- For most of your calculations, unless you want to worry about roundoff errors and add some extra steps to manage them, it is recommended to calculate prices and payments in whole number of cents so that $14.90 would be treated as 1490 cents, etc. Integer division and modular arithmetic will allow you to calculate dollars and cents easily from a value like 1490 cents.