(0) Download the Starter Kit for Lab2 (see our onCourse site). Unzip the downloaded file so that you have a new folder on your Desktop.

(1) Make **a new CONSOLE project**. You may use the Operating System (**OS**) and Integrated Development Environment (**IDE**) of your choice.

Part I: "Is it a Leap Year?"

(2) **Add to Project:** leap.cpp

(3) **Complete this program** to print a message indicating whether the year that is input from the keyboard is a **leap year** or not. The algorithm to determine whether a year is a leap year is shown below:

```
IF   (  (the year is divisible by 4   AND   the year is not divisible by 100)
                    OR
        (the year is divisible by 400)  )

                    it is a leap year

ELSE
                    it is not a leap year
```

Note: Use modulo division. In C++, `%` *is the modulo division operator. Google it to learn how to use it in your program.*

(4) Once you have your program working, **complete the following table:**

Year	**T/F divisible by 4**	**T/F not div. by 100**	**T/F divisible by 400**	**Yes/No Leap Year?**
200	true	false	false	NO
1492				
1998				

In the last row of the table, you find a year which succeeds for reasons *different* than 1492.

Call me over when you have completed this part of the lab.

Remove loop.cpp from your Project. Create a totally **new** (empty) .cpp file.

Part II: Practice with Loops

Welcome to the Hotel California
Such a lovely place
(Such a lovely place)

Such a lovely face

(5) Write a while loop to print "**Such a lovely place**" two (2) times, each time on its own line. After the loop finishes, print a blank line then print "Such a lovely face".

(6) Change the loop to print "Such a lovely place" ten (10) times.

(7) Prompt the user to enter an integer from the keyboard and then read in that integer. Change your loop to print "Such a lovely place" that many times.

Call me over to see this once you have this running.

Part III: Finding Averages

Her mind is Tiffany-twisted, she got the Mercedes bends
She got a lot of pretty, pretty boys she calls friends

(8) Hmmm, I wonder how many "pretty, pretty boys"? Write a (completely new) loop to continually prompt the user to enter an integer for "the number of pretty boys she calls friends" from the keyboard until the user (that'd be *you* ☺) enters a negative number and then at the very end print the number of times you entered a value, the sum of all the inputs, and the average of those values. The negative number should *not* be included in the average. Use the C language's **`printf()`** statement to format your output with two places *after* the decimal point, for example:

```
printf("The average is: %5.2f \n",  average);
```

Sample output (*italics are your program's output-prompts*, **bold is user input**)

```
Enter the number of pretty, pretty boys:   17
Enter the number of pretty, pretty boys:   21
Enter the number of pretty, pretty boys:    3
Enter the number of pretty, pretty boys:    5
Enter the number of pretty, pretty boys:   -1

The average is:  11.50
```

Call me over to see this once you have this running.

Part IV: Infinite Loops

Last thing I remember, I was
Running for the door
I had to find the passage back
To the place I was before
'Relax,' said the night man,
'We are programmed to receive.
You can check-out any time you like,
But you can never leave!'

(9) As you know, an infinite loop is a loop whose test never becomes false. Infinite loops are usually due to one (or all) of the following reasons:

a. you forgot the update (e.g., counter) inside your loop
b. your test which determines when to stop is insufficient
c. or as the song says:
"you can check out any time you like, but you can never leave"

(10) Remove your previous .cpp file from your Project. Add **loop.cpp**, to your project. **Without changing anything**, run this code. What happens?

(11) Read the comments in the code to see what I *really* want this loop to do and then answer the following questions. Write in your fix (**but the fix can NOT change the while loop test**; also continue to use the data type `float`) and list your expected output below.

(a) What will you do to fix this code? (*don't* change the test in the while loop)

(b) In the next step (but not yet!), you'll put in your fix to this code. Show me the output *that you expect* to see after you make this fix (only fill in left column**)**

Your expected OUTPUT: The real OUTPUT:

DO NOT GO BEYOND THIS POINT UNTIL YOU HAVE FILLED IN YOUR ANSWERS to the left column AND YOU HAVE TALKED WITH ME.

(12) Ok, now you can make your fix to **loop.cpp** (but do *not* change the while-loop test). Re-run the program. Write down the "real" output in the column above and continue to #15.

(13) Assuming that you *still* have an infinite loop, how is this possible? Isn't it true that:

$$0.1 + 0.1 + \ldots + 0.1 \text{ \textit{(twenty times)}} = \sum_{i=1}^{20} 0.1 = 20(0.1) == \texttt{2.0} \text{ ??}$$

So what's wrong?

(14) Time for the some "fancy" formatted output. Use a C printf statement to get formatted output to multiple places of precision. Insert this line at the inside-bottom of your while loop.

```
printf("sum is %.15f \n", sum);
```

(15) Ugh, what is wrong? *Before the loop*, try printing just **0.1** using the **`%.15f`** formatting. Get it now?

Write an English explanation as to what caused the original code to be an infinite loop.