Lab 1

Objectives

- Introduction to writing and running a C program in Jupyter Hub
- Introduction to print statement and escape sequences
- Introduction to variables, function definitions, function calls

During your registered lab time…

1. Create a new file called `lab1.c` in Jupyter Hub and work through the exercises following the instructions in the Exercises section below.
2. Download your `lab1.c` from Jupyter Hub (File->Download) and submit this file through the Lab 1 link under Assignments on the CSC111 BrightSpace page.
   Submissions after the deadline (Friday at midnight) will not be accepted and will receive a zero grade.

NOTE: grading will consider both correctness and quality of your code.

Exercises

1. Practice generating output:
   a. Write, compile, and run a program that prints `"Welcome to CSC 111"`.

      Recall, to compile: `gcc -Wall -Werror -pedantic -std=c18 -o lab1 lab1.c`
      Recall, to run: `./lab1`

   b. Add the following lines of code to your `main` function, below the `printf` statement you just wrote.

      ```
      int n = 111;
      double divisor = 1000.0;
      double result = n/divisor;
      ```

   c. Add code to print out the values of `n`, `divisor`, and `result` such that the result matches the output shown below. Think carefully about what format specifiers you will need to use.

      ```
      The value of n is 111 and the divisor is 1000.00
      The result is 0.111
      ```

   d. Modify the code you just wrote to produce the output below.

      ```
      The value of n is "111" and the divisor is "1000.00"
      The result is "0.111"
      ```

   e. Questions to think about (no need to hand in answers):
      i. What is the value of `result` if the type of `divisor` is changed to `int` instead of `double`? Why?
      ii. What happens if a floating-point value is accidentally printed with the wrong format specifier? (ie. if %d is used to print a variable of type `double`)

2. Defining and calling functions:

   Recall from physics, if an object is travelling with initial velocity *v* and acceleration *a* for *t* seconds, the total distance travelled *d* is given by the formula:

$$d = vt + \frac{at^2}{2}$$

   a. Design a function named `calculate_distances` You should have both a function prototype and the function definition should be after your main function.
   The function should calculate the distance for the values: `a=2`, `t=2` and `v=3` using the formula provided above and print the values of `a`, `t` and `v` and the calculated distance.

   b. Save and compile your program. If you see any error messages and fix your code and recompile until no errors appear. Read the error messages as they will give you hints about where and what the problem might be.
   If you run your program you will notice it does not generate the output you expect. This is because we DEFINED the function `calculate_distances`, but we did not CALL it!

   c. **Call** your `calculate_distances` function in your `main` after the code you wrote in Exercise 1. Recompile. Fix any errors and recompile until no errors appear. Run your program. You should see the following output:

```
Using a = 2.000000, t = 2.000000 and v = 3.000000
d = 10.000000
```

   d. Add code to your function so that it calculates and prints the distances for all of the following values `a`, `t` and `v`

      `a=2`, `t=2` and `v=3` (you should have this one already)
      `a=6`, `t=10` and `v=17`
      `a=0.5`, `t=1.06` and `v=11.1`

   e. Compile and run your program. You should now see the following output:

```
Using a = 2.000000, t = 2.000000 and v = 3.000000
d = 10.000000
Using a = 6.000000, t = 10.000000 and v = 17.000000
d = 470.000000
Using a = 0.500000, t = 1.060000 and v = 11.100000
d = 12.046900
```

   f. Questions to think about (no need to hand in answers):

      i. What types did you end up making `a`, `t` and `v`. What would the output be if you declared them to all be integers? Why?

      ii. What happens if you remove the function prototype and try to recompile your program?