CS118
Command-Line Interface Output

For each exercise, provide a separate .PY file that performs the specified task.
For each of the programs provided, use the command: `print('\x1b[2J')` to clear the console first.
Italicized questions should be answered in the code as a comment.

Once you have completed all programs, place them in a ZIP file and submit the ZIP file.

1. Write a Python3 program that uses `print()` to put the phrase "Hello, World!" (without the quotes) in the console.

2. Create a duplicate of the program from #1 and modify it to **print <u>three</u> blank lines** in the console BEFORE the words appear, and **two blank lines** AFTER the words – use the newline character (`\n`) to achieve this.

3. Create a duplicate of the program from #1, but modify it to **print two TAB characters** between "Hello," and "World". Use the tab escape sequence (`\t`) to achieve this.

4. Put the following code in a Python3 program and run it to see what it looks like:

```
FL1 = 3.141592658
IN1 = 7
ST1 = "My friend"
FL2 = 333.141592658
IN2 = 337
ST2 = "My super-duper friend"
print('String\t\tFloat\t\tInteger');
print("%s\t\t%f\t\t%d\n" % (ST1, FL1, IN1))
print("%s\t\t%f\t\t%d\n" % (ST2, FL2, IN2))
```

Modify the program's `print()` lines by adding *width modifiers* (to all placeholders) and *left-justification modifiers* (as needed) so that all columns align neatly as shown below. **Make the width modifiers be the same for each row in a column – for example, if you choose the first column to be 25 characters wide, use the same width modifier for each placeholder in that column.**

For this to work correctly, you'll need to add placeholders to the first (header) `print()` line and use the constant strings (aka "literals") in place of replacement variables just for this header `print()` line.

```
String                          Float                 Integer
My friend                    3.141593                       7
My super-duper friend      333.141593                     337
```

5. Write a Python3 program that will create the table below **without using any spaces** in the format string (`\t` and `\n` are fine – just no spaces). All words and numbers must be stored in variables. The hypens (`-`) and dollar signs (`$`) can be hardcoded in the format string.

Use width modifiers on each placeholder – normally these will be the same value in each row, although there are occasionally exceptions: for example, because the `$` in columns 2 and 3 takes up one space of the column, the width modifier for that column will be one unit smaller in those rows that have the `$` (compared to rows that don't have the `$`).

The placeholder may change in a row (for example, the first row – the header – will be strings, but the third row will contain numbers) but the width modifier should normally be the same.

I recommend hardcoding the words and numbers for this exercise, although in most programs they might come from user input. **Be sure that the columns align even if you change the values in the variables.**

Learning this technique will allow you to quickly prepare tabular output and will pay off handsomely on test day. The technique is called the "Rapid Table Construction" technique and a complete description is available in the Canvas files Rapid Table Construction folder.

```
Item                Input Value        Adjusted Value
----                ---------------    ---------------
Crust               $         0.0000   $           1.50
Sauce               $         0.0000   $           0.75
Cheese              $         1.2043   $           1.30
Pepperoni           $         2.9013   $           3.00
Anchovies           $         3.0123   $           3.10
                      -------------     --------------
Total                                  $           9.65
Overhead & Profit                      $          12.15


Sale Price                             $          21.80
```