I. Experimenting with methods.
In this lab you will be using 3 methods in several different ways.
I will often say to "comment out" some code. This means, change it to a comment so the computer will ignore it. This can be done several ways
- Just type # at the beginning of each line to be commented out
- In Thonny: Highlight the lines to be commented out. Under Edit select Comment out. (Useful for commenting out a whole chunk of code.) Notice there is also an option to Uncomment.
- In IDLE: Highlight the lines to be commented out. Under Format select Comment out region. (Useful for commenting out a whole chunk of code.) Notice there is also an option to Uncomment region.

A) In this section you will need to organize a program with some methods that I am giving you.
Download ExploringFunctions.txt to your Python Programs folder. Change the .txt to .py.
Add your name and section where the comment at the top indicates.

Run it. Call main(). (It does nothing yet, but it should run.)

Part B) Now we will be adding method calls.
Remove the ... at the top of main.

1) Add the method call
   \texttt{divAndMod(89, 5)}
   below /* comment 1 */ in main().
   Run the module. Call main().
   Results - the output:

   explanation:

2) Comment out the method call you just added.
   Below /* comment 2 */
   add the method call
   \texttt{rewriter('w')}
   Run the program. Call main().
   Results - the output:

   explanation:
3) **Comment out** the method call you just added. 
Below /* comment 3 */
**add** a method call to **triple**, providing one float literal **argument** (of your choice).
**Your method call:**

**Run** the program. Call main().
**Results - the output:** (Is there any?)
**explanation:**

If you did this part correctly, you only had **one active method call** each time you ran the program. This lets us concentrate on what that method call does. In the next several parts of this lab we will continue to have **only one active method call each time we run the program**.

4) **Comment out** the previous method call.
   (At this time there should be NO active method calls in main)

**Part C) Using the return value.**

1) Below /* comment 4 */ **add** the lines
   ```
   x = triple(1.1)
   print("x = {0:.2f}".format(x))
   ```
   (Here we are assigning the **return value** of **triple** to the variable **x**.  
   1.1 is an **argument** for **triple**. **1.1** is a literal - an actual value, not a variable.)
   **Run** the program and call main().
   **Results - the output:**
   **explanation:**

2) **Comment out** the two lines you just added.
   Below /* comment 5 */ **add** the lines
   ```
   a = rewriter('w')
   print("\na = ", a)
   ```
   **Run** the program. (Don't change rewriter!)
   **Results - the output:** (What happened?)
   **explanation:**
3) **Comment out** the two lines you just added.
Below /* comment 6 */,
- **add a call** to \texttt{divAndMod}, providing two literal arguments, and **assign the result (return value)** to a variable \( r \).
- **Print** \( r \) after the method call, with a label \( r = \)
**Run** the program.
**Results - the output:**

**explanation:**

4) **Comment out** the lines you just added.

**Part D) Using variables** instead of **literals** as **arguments**.

1) **Add** the lines
\[
\begin{align*}
y &= 2.1 \\
x &= \text{triple}(y) \\
\text{print}("x = \{0:.2f\}.format(x)) \\
\text{print}("y = \{0:.2f\}.format(y))
\end{align*}
\]
below /* comment 7 */
**Run** the program and call main().
**Results - the output:**

**explanation:**

2) **Comment out** the four lines you just added.
**Add** the lines
\[
\begin{align*}
n &= 23 \\
d &= 10 \\
r &= \text{divAndMod}(n,d) \\
\text{print}("r =", r) \\
\text{print}("n =", n, "d =", d)
\end{align*}
\]
below /* comment 8 */
**Run** the program and call main().
**Results - the output:**

**explanation:**
3) **Comment out** the five lines you just added.

Below /* comment 9 */ **add** lines that

- **assign** a string literal to a variable \( s \),
- use \( s \) as an **argument** to a call to rewriter, and
- add a **print** statement (to main) that prints the variable \( s \), labeling it with the variable name to which it is assigned. This should be printed to a new line, not the line with the output from rewriter.

**Run** the program and call main().

**Results - the output:**

**explanation:**

4) **Comment out** the three lines you just added.

**Part E** Putting it all together. Under the comment /* Part E */, **write code** to produce the following output. Use the three methods **as much as possible** to produce this output. Do **NOT** change the method definitions. (This program has no input from the user.) Some of what you see will be printed by main and some by the other three methods.

```
*****$$$$$NNNNN
Dividing 13 by 5:  The quotient is 2
                  The remainder is 3
Dividing 15 by 5:  The quotient is 3
                  The remainder is 0

+----------------+
When I triple some number I get 33.3.

   eeeeee
  ccccc
 sssss
```

**Print** out your **program ExploringFunctions.txt** and the **output** for part E to hand in with this lab packet.
II. Introduction to Scope.
A) Download and run Scoper.txt (change to .py)

In which function does \( a \) get its value from the statement
\[
a = 1
\]

In which function does \( a \) get its value from the statement
\[
a = 22
\]

Outside of the functions which value does \( a \) have, 1 or 22?

B) Uncomment the line
\[
#third()
\]

Run the program. What happens?

III. Submitting your work.
• Make sure your name is on this lab packet.
• Staple
  
  the program ExploringFunctions.txt and its output to the Lab 11 packet.
• Turn the lab in at the file cabinet.