CS118 Exercises
Loops

In the following exercises, do not use any special commands like `break` or `continue`.

1. Write a Python program that collects three integers from the user without loops. Add the three values together and report the sum.

2. Write a Python program that collects three integers from the user using a FOR loop. Maintain a *running total* through the loop. Report the sum after three values have been totaled.

3. Write a Python program that collects three integers from the user using a WHILE loop. Maintain a *running total*. Report the sum after three values have been totaled.

4. Make a copy of #2 and modify it to collect 10 integers from the user.

5. Make a copy of #3 and modify it to collect 10 integers from the user.

6. Make a copy of #2 and modify it to ask the user how many values will be collected before beginning the loop, and then have the loop collect that many values and sum them.

7. Make a copy of #3 and modify it to ask the user how many values will be collected before beginning the loop, and then have the loop collect that many values and sum them.

8. Make a copy of #3 and modify it to have the loop collect values only until the user enters -99 for a value.

9. Write a Python program that will collect two integers from the user, repeating until only one of them is negative.

10. Write a Python program that collects an integer, `N`, from the user. Have a loop collect `N` integers from the user, maintaining a running total. Report the final result to the shell window.

11. Write a Python program that implements this operation:

$$p = \sum_{k=0}^{N} \frac{(-3)^{-k}\sqrt{12}}{2k+1}$$

where `N` is a positive integer (less than 25) obtained from the user. The value produced should be familiar. Using `print()`, display `p` during each iteration to 12 decimal places. Which is more appropriate – a `FOR` loop, or a `WHILE` loop?

12. In this assignment you will need to *nest* WHILE loops:

Write a Python program that will collect a positive integer from the user. Collect another positive integer and add the product of the two integers to a running total. Keep collecting a new second integer and multiplying by the same first integer and adding to the running total until the user enters 0 for the second integer. When that occurs, go back to collecting the first and second integers (i.e. start again), repeating until the first integer is negative or zero. When the first integer becomes negative or zero, do not collect the second integer nor add to the running total - simply report the running total. Your output should look like the following – be careful to get it exactly right!

```
Integer #1: 3
Integer #2: 2
Integer #2: 5
Integer #2: 7
Integer #2: 4
Integer #2: -4
Integer #2: 2
Integer #2: 0
Integer #1: 5
Integer #2: 1
Integer #2: 10
Integer #2: 0
Integer #1: -3
The running total is: 103
```