

Unit 3: Conditions, If and While
Skill Builder 2: Flag Waving

In this lesson, you will learn about *counters* and *accumulators* by calculating the number and the total of numbers entered and determine the average of the numbers.

Objectives:

- Counter statement
- Accumulator statement
- **While** loop with a flag
- Calculating an average (mean)

You will write a program that will count and add up a set of numbers, but you do not know in advance how many numbers there will be. This lesson introduces the concept of a counter statement, an accumulator statement, a “flag” value to end a loop, and the calculation of the average of the entered numbers.

1. Begin a new Python file and name it **COUNTER**.

Make two variables and set them both equal to zero:

count = 0

total = 0

count keeps track of how many numbers are entered and

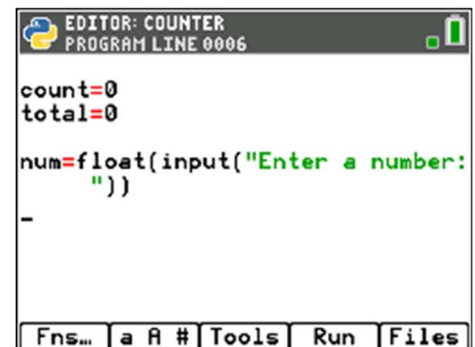
total keeps a running total of the numbers that have been entered.

A third variable, **num**, stores each number entered one at a time.

Rather than setting it equal to 0, use an **input** statement to get the first number from the user:

num=float(input("Enter a number: "))

Remember that most of the Python programming tools are found under <Fns...>.



2. Now start a **while** loop that will end when a certain number is entered. Some options here are 0, -999 or -1. Keep in mind that this unique number cannot be one of the numbers you are trying to process. We will use -999 as our “flag” value to signal that we have finished entering numbers.

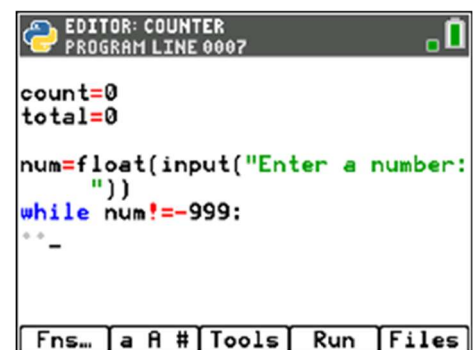
while num != -999:

*Recall that **while** is found on <Fns...> Ctl and **!=** is on all the places you can find **==**. Simplest is the [test] menu.*

3. The **while** loop **block** will:
 - a) Count the number of numbers entered.
 - b) Add them up.
 - c) Then, ask for another number.

Counting is performed with the statement

count = count + 1





10 Minutes of Code: Python

UNIT 3: SKILL BUILDER 2

TI-84 PLUS CE PYTHON

STUDENT ACTIVITY

which adds 1 to the variable **count** each time it is processed.

Adding them up, or totaling, is accomplished with the statement:

total = total + num

“Ask for another number” requires an **input** statement like the first one:

num=float(input(“Enter a number (-999 to end): ”))

but here we add a message telling the user how to end the loop.

*Tip: Copy the first **input** statement and paste it here, then edit it.*

Be sure all three of these statements are properly indented.

*The **input** statement comes **last** in the loop body to control the loop.*

```

EDITOR: COUNTER
PROGRAM LINE 0010
count=0
total=0

num=float(input("Enter a number:
"))
while num!=-999:
    count=count+1
    total=total+num
    num=float(input("Enter a numbe
r(-999 to end):"))

```

- Test your program so far since the **while** loop is complete. Enter any number and enter -999 to end the program.

Question: Is -999 included in the **total**?

Answer: No!

*Tip: If your program does not end when you enter -999, you can press **[on]** to break it, go into the Editor and check your code.*

```

PYTHON SHELL

>>> # Shell Reinitialized
>>> # Running COUNTER
>>> from COUNTER import *
Enter a number:50
Enter a number(-999 to end):98
Enter a number(-999 to end):-999
>>> |

```

- We have finished with the loop. Skip a line or two and erase (backspace) the indent spaces. *Note the cursor position in this image.*

Now is the time to process the inputted numbers. We kept track of the **count** and the **total** so now (after the loop ends) we can calculate the average.

Try it yourself before looking at the next step.

```

EDITOR: COUNTER
PROGRAM LINE 0012
num=float(input("Enter a number:
"))
while num!=-999:
    count=count+1
    total=total+num
    num=float(input("Enter a numbe
r(-999 to end):"))

```

- The statement

avg = total / count

calculates the average and stores the result in the variable **avg**.

Ponder this: Why do we use / here and not // as in the last lesson?



TI-84 PLUS CE PYTHON

STUDENT ACTIVITY

Finally, write the **print** statement(s) that report the count, total and average to the user. You may wish to use more than one print statement. Our code is shown in the next step.

- We used three print statements at the end of the program to report the results:

```
print("count    = ", count)
print("total    = ", total)
print("average = ", avg)
```

A sample run of the completed program is shown.

Sometimes you might notice the average has a lot of trailing zeros followed by a 1. Python's method of storing and computing with binary floating point numbers then converting them to decimal approximations can lead to stray decimals in some calculations. This anomaly is common with Python on virtually any computing platform and has nothing to do with the TI-84 Plus CE Python graphing calculator.

```
EDITOR: COUNTER
PROGRAM LINE 0014
"))
while num!=-999:
**count=count+1
**total=total+num
**num=float(input("Enter a number
r(-999 to end):"))
**
**
avg=total/count
```

```
EDITOR: COUNTER
PROGRAM LINE 0018
**num=float(input("Enter a number
r(-999 to end):"))
**
**
avg=total/count

print("count=",count)
print("total=",total)
print("average=",avg)
```

```
PYTHON SHELL
>>> # Running COUNTER
>>> from COUNTER import *
Enter a number: 65
Enter a number(-999 to end):83
Enter a number(-999 to end):7
Enter a number(-999 to end):2
Enter a number(-999 to end):-999
count= 4
total= 157.0
average= 39.25
>>> |
```

```
PYTHON SHELL
>>> # Running COUNTER
>>> from COUNTER import *
Enter a number:56
Enter a number(-999 to end):98
Enter a number(-999 to end):45
Enter a number(-999 to end):78
Enter a number(-999 to end):-999
count= 4
total= 277.0
average= 69.25000000000001
>>> |
```