

Lab 7 – Loops I

Chapter:	5. Loops
Time:	80 Minutes

Lab 7

Objectives

- To write programs for executing statements repeatedly by using a *while* loop.
- To develop loops following the loop design strategy.
- To control a loop with the user's confirmation.
- To control a loop with a sentinel value.
- To control a loop with a counter.
- To learn loops from a variety of examples.

Current Lab Learning Outcomes (LLO)

By completion of the lab the students should be able to

- Use *while* loops.
- Write counter-controlled loops.
- Write sentinel-controlled loops.
- Avoid infinite loops.

Lab Requirements

- PyCharm (IDE).




Practice Activities with Lab Instructor (20 minutes)

Problem 1


Programming Exercises (5.1)

Write a program that reads an unspecified number of integers, determines how many positive and negative values have been read, and computes the total and average of the input values (**not counting zeros**). Your program ends with the input 0. Display the average as a floating-point number.

Here are some sample runs:



```
Enter an integer, the input ends if it is 0: 5 <enter>
Enter an integer, the input ends if it is 0: 8 <enter>
Enter an integer, the input ends if it is 0: -1 <enter>
Enter an integer, the input ends if it is 0: 2 <enter>
Enter an integer, the input ends if it is 0: 0 <enter>
The number of positives is 3
The number of negatives is 1
The total is 14
The average is 3.5
```



```
Enter an integer, the input ends if it is 0: 0 <enter>
No numbers are entered except 0
```

Solution

Phase 1: Problem-Solving Phase:

- 1- Define two variables to be counters for positive and negative numbers (**countPositive**, **countNegative**).
 - Initialize the counters to zero.
 - `countPositive = 0`
 - `countNegative = 0`
- 2- Define a variable to be a counter for all numbers regardless of their signs (**count**).
 - Initialize the counter to zero.
 - We need this counter in the average calculation (**average = total / count**).
 - `count = 0`
- 3- Define a variable to store the sum of all entered numbers (**total**).
 - Initialize the variable to zero.
 - We need this counter in the average calculation (**average = total / count**).
 - `total = 0`

- 4- Ask the user to enter a number (`number`).
 - `number = eval(input("message..."))`
- 5- Define a while loop and set its condition to `number != 0` (while `number` is not equal to 0):
 - Check the number (`number`) to determine if its positive or negative:
 - If (`number > 0`) Then:
 - `number` is positive, so increase `countPositive` by one.
 - `countPositive += 1`
 - Else If (`number < 0`) Then:
 - `number` is negative, so increase `countNegative` by one.
 - `countNegative += 1`
 - Add `number` to `total`.
 - `total += number`
 - Increase `count` by one.
 - `count += 1`
 - Ask the user to enter the next number (`number`).
 - `number = eval(input("message..."))`
 - If the user entered "0", the loop condition will be `False`, and the loop statement will end after the current iteration. Then the next step (Step #6) will be executed.
 - Otherwise (if the user entered any number rather than 0), the loop will continue looping.
- 6- Check `count` to determine if the user entered any number except 0, and then display the result:
 - If (`count == 0`) Then:
 - Print "No numbers are entered except 0"
 - Else:
 - Calculate the average (`average`).
 - `average = total / count`
 - Display the result (`countPositive`, `countNegative`, `total`, `average`)

Phase 2: Implementation Phase:

1. Create a new project and name it "Lab 7".
2. Create a new file and name it "activity_1.py".
3. Write the following code in the file:

activity_1.py

```
1  # Define two variables to be counters for positive and negative numbers
2  countPositive = 0
3  countNegative = 0
4  # Define a variable to be a counter for all numbers regardless of +,-
5  count = 0
6  # Define a variable to store the sum of all entered numbers
7  total = 0
8
9  # Ask the user to enter a number
10 num = eval(input(
11     "Enter an integer, the input ends if it is 0: "))
12
13 # Define a while loop and set its condition to number != 0
14 # (while number is not equal to 0)
15 while num != 0:
16     # Check the number to determine if its positive or negative
17     if num > 0:
18         countPositive += 1
19     elif num < 0:
20         countNegative += 1
21
22     # Add number to total
23     total += num
24     # Increase count by one
25     count += 1
26
27     # Ask the user to enter the next number
28     num = eval(input(
29         "Enter an integer, the input ends if it is 0: "))
30
31 # Check count to determine if the user entered any number except 0,
32 # and then display the result
33 if count == 0:
34     print("No numbers are entered except 0")
35 else:
36     # Calculate the average
37     average = total / count
38
39     print("The number of positives is", countPositive)
40     print("The number of negatives is", countNegative)
41     print("The total is", total)
42     print("The average is", average)
```

Problem 2

Programming Exercises (5.5)

Write a program that displays the following two tables side by side (note that 1 kilogram is 2.2 pounds and that 1 pound is 0.45 kilograms):

kilograms	pounds		pounds	kilograms
1	2.2		20	9.09
3	6.6		25	11.36
5	11.0		30	13.64
7	15.4		35	15.91
...				
193	424.6		500	227.27
195	429.0		505	229.55
197	433.4		510	231.82
199	437.8		515	234.09

Solution

Phase 1: Problem-Solving Phase:

1- Print the table header.

- Use `format(value, format_specification)` to format the output.
- Example: `format("Text", ">10s")` means the string (Text) is formatted within a width of 10 and right-justified.
- `print(format("kilograms", "<10s"), format("pounds", ">10s"),
" | ", format("pounds", "<10s"), format("kilograms",
">10s"))`

2- Print the line below the table header.

- `print("-----")`

3- Define a variable to be a counter for the loop iterations (count).

- Initialize the counter to zero.
- We need this counter in the loop continuation condition to check if the loop iterates 100 times.
- `count = 0`

4- Define a variable to store the initial value of the kilograms that should be converted first to pounds (kilograms).

- `kilograms = 1`

5- Define a variable to store the initial value of the pounds that should be converted first to kilograms (pounds).

- `pounds = 1`

6- While `count` is less than 0:

- Convert `kilograms` to pounds (`kilograms_to_pounds`):
 - `kilograms_to_pounds = kilograms * 2.2`
- Convert `pounds` to kilograms (`pounds_to_kilograms`):
 - `pounds_to_kilograms = pounds / 2.2`
- Print a new row and format the output to look like a table.
 - ```
print(format(kilograms, "<10d"),
 format(kilograms_to_pounds, "10.1f"),
 "|",
 format(pounds, "<10d"),
 format(pounds_to_kilograms, "10.2f"))
```
- Increment `kilograms` by 2:
  - `kilograms = kilograms + 2`
- Increment `pounds` by 5:
  - `pounds = pounds + 5`
- Increment `count` by 1:
  - `count = count + 1`

## Phase 2: Implementation Phase:

1. Open the project "Lab 7" if it was not opened or create it if it was not existing.
2. Create a new file and name it "activity\_2.py".
3. Write the following code in the file:

### activity\_2.py

```
1 # Table header
2 print(format("kilograms", "<10s"),
3 format("pounds", ">10s"),
4 " | ",
5 format("pounds", "<10s"),
6 format("kilograms", ">10s"))
7 print("-----")
8
9 # Define the counters
10 kilograms = 1
11 pounds = 20
12 count = 0
13
14 # Table body
15 while count < 100:
16 # Convert units
17 kilograms_to_pounds = kilograms * 2.2
18 pounds_to_kilograms = pounds / 2.2
19 # Print a new formatted row
20 print(format(kilograms, "<10d"),
21 format(kilograms_to_pounds, "10.1f"),
22 " | ",
23 format(pounds, "<10d"),
24 format(pounds_to_kilograms, "10.2f"))
25 # Update the counters
26 kilograms += 2
27 pounds += 5
28 count += 1
```



## Individual Activities (60 minutes)

### Problem 3

Programming Exercises (5.6)

Write a program that displays the following two tables side by side (note that 1 mile is 1.609 kilometers and that 1 kilometer is 0.621 mile):

| Miles | Kilometers |  | Kilometers | Miles  |
|-------|------------|--|------------|--------|
| 1     | 1.609      |  | 20         | 12.430 |
| 2     | 3.218      |  | 25         | 15.538 |
| 3     | 4.827      |  | 30         | 18.645 |
| 4     | 6.436      |  | 35         | 21.753 |
| 5     | 8.045      |  | 40         | 24.860 |
| 6     | 9.654      |  | 45         | 27.968 |
| 7     | 11.263     |  | 50         | 31.075 |
| 8     | 12.872     |  | 55         | 34.183 |
| 9     | 14.481     |  | 60         | 37.290 |
| 10    | 16.090     |  | 65         | 40.398 |




#### Problem 4

Programming Exercises (5.41)

Write a program that reads integers, finds the largest of them, and counts its occurrences. Assume that the input ends with number 0. Suppose that you entered 3 5 2 5 5 5 0; the program finds that the largest number is 5 and the occurrence count for 5 is 4.

**Hint:** Maintain two variables, `max` and `count`. The variable `max` stores the current maximum number, and `count` stores its occurrences. Initially, assign the first number to `max` and 1 to `count`. Compare each subsequent number with `max`. If the number is greater than `max`, assign it to `max` and reset `count` to 1. If the number is equal to `max`, increment count by 1.

Here is a sample run:



```
Enter a number (0: for end of input): 3 <enter>
Enter a number (0: for end of input): 5 <enter>
Enter a number (0: for end of input): 2 <enter>
Enter a number (0: for end of input): 5 <enter>
Enter a number (0: for end of input): 5 <enter>
Enter a number (0: for end of input): 5 <enter>
Enter a number (0: for end of input): 0 <enter>
The largest number is 5
The occurrence count of the largest number is 4
```

## Extra Exercises (**Homework**)

### From the Textbook

- Programming Exercises:
  - 5.2
  - 5.7
  - 5.8
  - 5.9
  - 5.12
  - 5.13
  - 5.14
  - 5.15

### From MyProgrammingLab (<https://pearson.turingscraft.com>)

- 5.2
  - 51180
  - 51181
  - 51187
  - 51251
  - 51252
  - 51253

### Upload Your Solutions



Upload your solutions of the lab activities to Blackboard.