# CSC148 Lab 5: Looping and Conditionals, Range().

First Name: Jeffrey

Last Name: Williams

AU ID: 5030143

Total: 25pts+3pts(Bonus)

From this lab session onwards, we will actually use the lab to “learn” new concepts and syntax we haven’t talked in class. They are usually relatively easy for you to figure out by simply doing the exercises and get feedback. You can absolutely ask questions and discuss with your TA and me. I hope this “active learning” will complement with our lectures. However, it is expected that you turn on your own original code.

NO COPY FROM THE INTERNET

The purpose of this lab is to get you comfortable with solving simple problems with “While” loop, for loop and explore the *range( )* functions. Please continue to practice with function definitions and how to call a function.

1. (10pts) The purpose of this exerci­se is to practice with *range( )* functions and learn more about *for* loops by observing its behavior.

*range( )* function is a very useful built-in function. *range(n)* will generate a list of numbers between 0 and n-1. The formal definition of *range( )* can be found here:

<https://docs.python.org/2/library/functions.html#range>

range(start, end, step). By default, start = 0, end = n-1, step =1.

*range(4,10)* 🡪 [4 5 6 7 8 9]

Please write down (without using your computer!!) the outcome of the following codes. Then check your answer with your computer.

|  |  |  |  |
| --- | --- | --- | --- |
| Expression | Expected value | Computer value | Reason |
| range(1,10) | [1 2 3 4 5 6 7 8 9] | [1 2 3 4 5 6 7 8 9] | Expression asks to count to 10 starting from 1, which is 9 because python always starts at 0 since range() is index-based |
| range(10,1,-2) | [1 2 3 4 5 6 7 8 9 7] | [10 8 6 4 2] | Expression means to go from 10 to 1, with the extra parameter meaning to subtract by 2 each time |
| range(0, 30,5) | [0 5 10 15 20 25 30] | [0 5 10 15 20 25] | Go from 0 to 30, increasing by 5 (third parameter) |
| range(1,0) | [0] | Nothing | There is no number to count from 1, so nothing prints |
| range(-5,0) | Nothing | [-5 -4 -3 -2 -1] | Python starts at a negative number (less than zero) and increases to zero. |
| range(5, -1, -1) | [5 4 3 2 1 0] | [5 4 3 2 1 0] | Starts at 5 and decreases by 1 to index -1. |
| range(-1,-6,-2) | [-1 -3 -5] | [-1 -3 -5] | Starts at -1 and decreases to -6 (which is -5 index) by intervals of 2 |
| range(0,6) | [1 2 3 4 5] | [0 1 2 3 4 5] | Starts at minimum range value 0 and increases to 6, which is 5 because it starts at 0. |

Let *s = “heLLo WorLd”*

Now fill in the table below, as usual. Notice the word above is a mixed-case word.

­­­

|  |  |  |
| --- | --- | --- |
| Expression | Expected Value | Calculated Value |
| s[1] | e | e |
| s[15] | An error | An error |
| s[1:5] | eLLo | eLLo |
| s[:5] | Error | heLLo |
| s[4:] | WorLd | o WorLd |
| ‘e’ in s | True | True |
| ‘x’ in s | False | False |

|  |  |  |
| --- | --- | --- |
| Expression | Expected Value | Calculated Value |
| s.index(‘e’) | 1 | 1 |
| s.find(‘e’) | 1 | 1 |
| s.index(‘x’) | 0 | Error |
| s.find(‘x’) | Error | -1 |
| s.count(‘o’) | 2 | 2 |
| s.index(‘L’,5)  #This means, “Look starting from index 5” | 8 | 9 |

Print out the outcome of the following code. Please first write down your results and then check with your computer. The best thing to do is to write what values of ‘i” would be for each iteration.

For example:

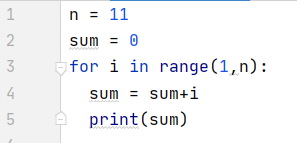
i =1 sum = 1

i = 2 sum = 3

i = 3 sum = 6

etc…

(a)



i = 1, sum = 1

i = 2, sum = 3

i = 3, sum = 6

i = 4, sum = 10

i = 5, sum = 15

i = 6, sum = 21

i = 7, sum = 28

i = 8, sum = 36

i = 9, sum = 45

i = 10, sum = 55

A list can include any type number, string, etc. You use [ ] to represent list. Here is an example of using list to loop through its elements.

Here is a tutorial of list:

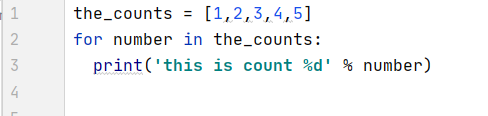
<http://www.tutorialspoint.com/python/python_lists.htm>

(b)

We also use a bit of formatted printing in this exercise. To learn more about this, see here:

<http://learnpythonthehardway.org/book/ex8.html>

Basically, *%d* means integer, and *%s* means string. You use % to connect the string with the number.



this is count 1

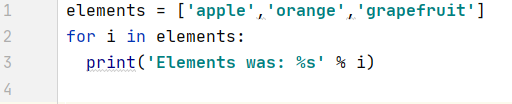
this is count 2

this is count 3

this is count 4

this is count 5

(c)

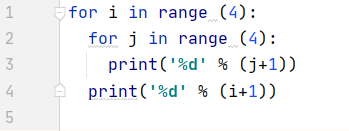


Elements was: apple

Elements was: orange

Elements was: grapefruit

(d)



j = 1

j = 2

j = 3

j = 4

i = 1

j = 1

j = 2

j = 3

j = 4

i = 2

j = 1

j = 2

j = 3

j = 4

i = 3

j = 1

j = 2

j = 3

j = 4

i = 4

2. (5pts) This question tested nested for loops.

Write a function called ***plot(s)*** in which ***s*** is a string representing a social security number written as contiguous digits (for instance, 509435456). It then prints each digit on a separate line followed by a colon and the digit printed the number of times equal to its value. Thus the output for 509435456 would be

5:55555

0:

9:999999999

4:4444

3:333

5:55555

4:4444

5:55555

6:666666

You must use a nested loop to do this. Note that a **for** loop, for instance, beginning with **for j in range(0)** is executed zero times. Design your program so it will work for a string of any length.

3. (5pts) You will practice While loop and Conditionals in this exercise.

I would suggest you try this out by first write down the flow chart. What are the conditions for enter a loop? What is the condition for *if/elif* statement.

Write a program to ask the user to input 10 integers, then prints the largest odd number. If no odd number is entered, it should print a message to that effect.

Here are some outputs:

AU60185:Lab5 bxiao$ Python Lab5.py

enter a number 1-10: 1

enter a number 1-10: 3

enter a number 1-10: 5

enter a number 1-10: 6

enter a number 1-10: 7

enter a number 1-10: 8

enter a number 1-10: 9

enter a number 1-10: 13

enter a number 1-10: 10

enter a number 1-10: 132

maximum odd number: 13

Hint: here are some code to get your started

count = 0

oddNumberCount = 0

maximum = 0

while count < 10:

count = count +1

4. (5pts) Each new term in the Fibonacci sequence is generated by adding the previous two terms. By starting with 1 and 1, the first 10 terms will be:

1, 1, **2,** 3, 5, **8,** 13, 21, **34**, 55, 89, ...

By considering the terms in the Fibonacci sequence whose values do not exceed **4 million**, find the sum of the even-valued terms.

Time your computation. How much time elapsed until you get your results?

Hint :

Please first observe the Fibonacci sequence. When is the number even?

When the number is very large, the program can be slow.

Make sure to consider the computational efficiency when designing your algorithm. You can use time module.

Import time

Start = time.time()

#your code

End = time.time()

print(end-start)

report the time that your code needs to solve this question.

5. (3pts Bonus) Smallest fraction number.

Ask the user for a few fraction numbers, such as 12/5 or 94/37, find the smallest fraction number that they entered.

Please enter a fraction number: 12/5

Do you have more? Yes

Please enter a fraction number: 94/37

Do you have more? Y

Please enter a fraction number: 25/3

Do you have more? yes

Please enter a fraction number: 90/57

Do you have more? no

Your program should return

The smallest fraction is 90/57.

Hint: s = “12/5”

s[0]-> 12

first value = int(s[0]) # this returns the 12 as an integer