**Programming for Data Science**

**Practical 01 – Intro to Python programming**

**What you will learn / do in this lab**

1. Perform input and output
2. Work with basic data types like integers, floats and strings
3. Use assignment, arithmetic, comparison and logical operators
4. Write if/else statements in Python
5. Write for/ while loops in Python
6. Write Python functions
7. Use Python lists

**Table of Contents**

[1. Overview 3](#_Toc67646864)

[A. What you will do for this lab 3](#_Toc67646865)

[B. Intro to Python 3](#_Toc67646866)

[2. Inputs, Outputs and Variables in Python 4](#_Toc67646867)

[Task 2-1: Basic Inputs and Outputs in Python 4](#_Toc67646868)

[Task 2-2: Formatting outputs 4](#_Toc67646869)

[Task 2-3: Average of electricity bills (SUBMISSION REQUIRED) 5](#_Toc67646870)

[3. Using import in Python 6](#_Toc67646871)

[Task 3-1: Timer 6](#_Toc67646872)

[Task 3-2: Reaction game 7](#_Toc67646873)

[4. Working with Python numeric types 8](#_Toc67646874)

[Task 4-1: Perform basic mathematical calculations 8](#_Toc67646875)

[Task 4-2: Mathematical functions in Python 9](#_Toc67646876)

[5. Working with Python string types 10](#_Toc67646877)

[Task 5-1: Perform simple string operations 10](#_Toc67646878)

[Task 5-2: Perform simple string operations (SUBMISSION REQUIRED) 11](#_Toc67646879)

[6. if-else statements 12](#_Toc67646880)

[Task 6-1: Odd or even 12](#_Toc67646881)

[Task 6-2: Spdonalds (SUBMISSION REQUIRED) 13](#_Toc67646882)

[7. for loop 14](#_Toc67646883)

[Task 7-1: Repeat a user-specified message 14](#_Toc67646884)

[Task 7-2: Calculate sum of numbers within a range (SUBMISSION REQUIRED) 14](#_Toc67646885)

[8. while loop (optional) 16](#_Toc67646886)

[Task 8-1: Count down timer 16](#_Toc67646887)

[Task 8-2: High-low number game 17](#_Toc67646888)

[9. Python Lists 18](#_Toc67646889)

[Task 9-1: Lists (Elements less than 5) 18](#_Toc67646890)

[Task 9-2: Lists (max, min, sorted, reverse) 18](#_Toc67646891)

[Task 9-3: List Slicing (SUBMISSION REQUIRED) 19](#_Toc67646892)

[10. Python functions 20](#_Toc67646893)

[Task 10-1: Scissors, Paper, Stone 20](#_Toc67646894)

[Task 10-2: Odd and Even (SUBMISSION REQUIRED) 21](#_Toc67646895)

# Overview

## What you will do for this lab

In this lab, you will learn how to write a few simple Python programs to get you familiar with the syntax of the Python programming language.

You will **NOT** be learning any data science-specific Python commands yet. Those will be featured in later topics, so please wait patiently for them 😊

## Intro to Python

Python has become one of the de facto standard language and platform for data analysis and data science.

Python is a very versatile programming language that that you can do many things with, from performing simple mathematical operations to data visualization and even machine learning and predictive analytics.

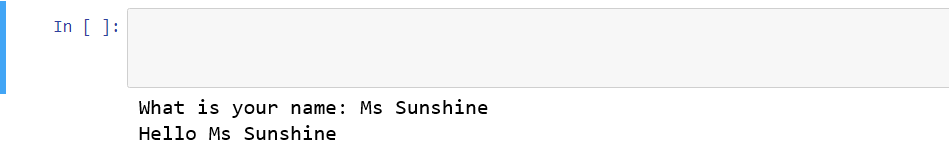
Regardless of the task at hand, make sure you are familiar with the basic concepts of the Python programming language first before delving into the more complicated stuff!

|  |  |
| --- | --- |
| **Python can be used to perform simple mathematical operations like this one.** | **Python can also be used to plot complex visualizations like this one.** |
| |  | | --- | | x = 5  y = 3  z = x\*y + y/x | | image |

# Inputs, Outputs and Variables in Python

### Task 2-1: Basic Inputs and Outputs in Python

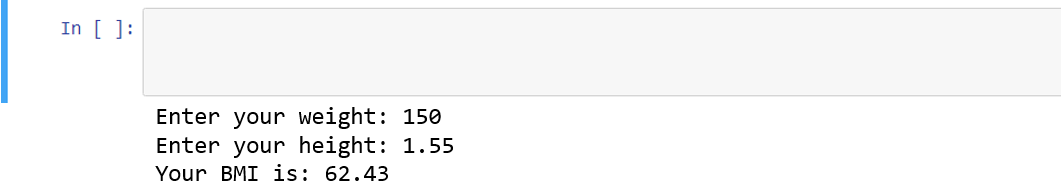
Write a Python program that prompts the user for his name, and outputs a greeting to him with his name as shown below.



### Task 2-2: Formatting outputs

Write a Python program that prompts the user to enter his height and weight, then calculates his BMI to a precision of 2 decimal place as shown below.

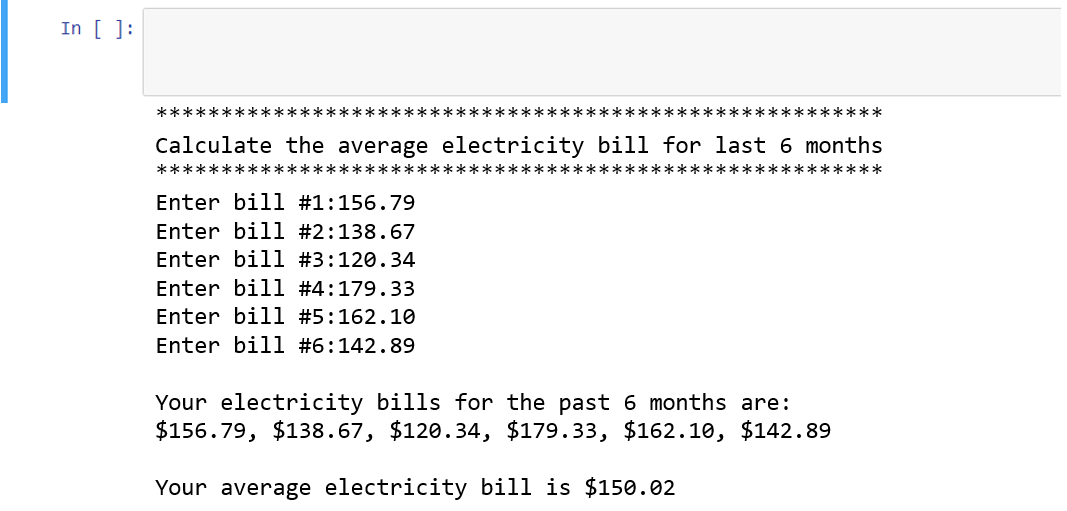
Hint -- > Remember to convert your inputs to datatype float before performing calculations!



### Task 2-3: Average of electricity bills (SUBMISSION REQUIRED)

Write a Python program that prompts the user to enter his electricity bill for the last 6 months and prints the average of his electricity bills as shown below.

Hint -- > Remember to convert your inputs to datatype float before performing calculations!



# Using import in Python

As mentioned earlier, Python programs can do *almost* anything.

Why is Python is so “powerful”?

The reason is simple – there are many other Python programmers out there, and they have written a lot of Python code which you can just re-use by “importing” their code, also known as “modules” into your own Python program.

In this section, you will learn how to import Python modules that can help you complete certain tasks you may need to do.

For example, you may need your Python program to wait for 2 seconds before executing the next action. In that case, you can import the ***time*** module, which will allow you to make your Python programme “sleep” for a specified number of seconds.

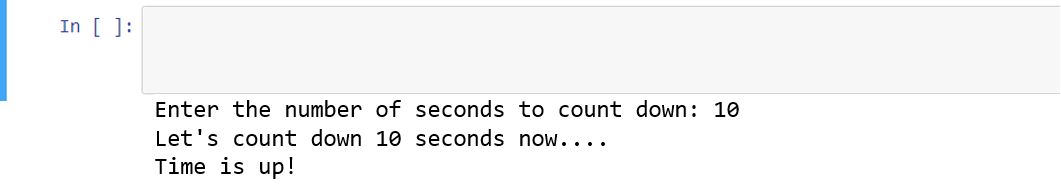
Perhaps you need to generate a list of 10 random numbers in the range of 1 to 100. To do that, you can import the ***random*** module, which will allow you to generate random numbers of your preferred range.

### Task 3-1: Timer

Write a Python program that acts like a count-down timer.

Your program should:

1. Prompt the user to enter the number of seconds he wishes to count down and store it in a variable called *time\_to\_wait*
2. Use the sleep function to wait for the specified number of seconds
3. Display a message “Time is up!” when the specified number of seconds has elapsed

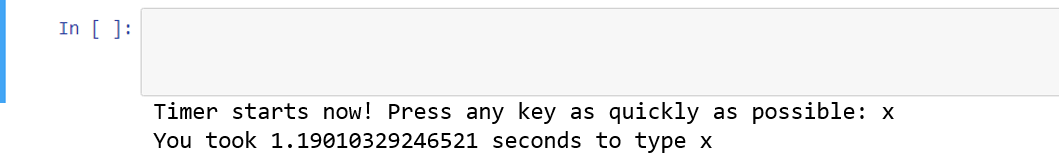


### Task 3-2: Reaction game

Write a Python ‘game’ that tests how fast a user can respond.

Your program should:

1. Store the current time in a variable *time\_now*
2. Prompt the user with a message “Timer starts now! Press any key as quickly as possible:” and wait for the user to key in his input in a variable called *user\_input*
3. Record and store the time at which the user’s input was captured in a variable *time\_entered*
4. Calculate the time that has elapsed between time\_now and time\_entered, which is the time the user took to respond. Store this in a variable *time\_response*
5. Display *time\_response* as shown below



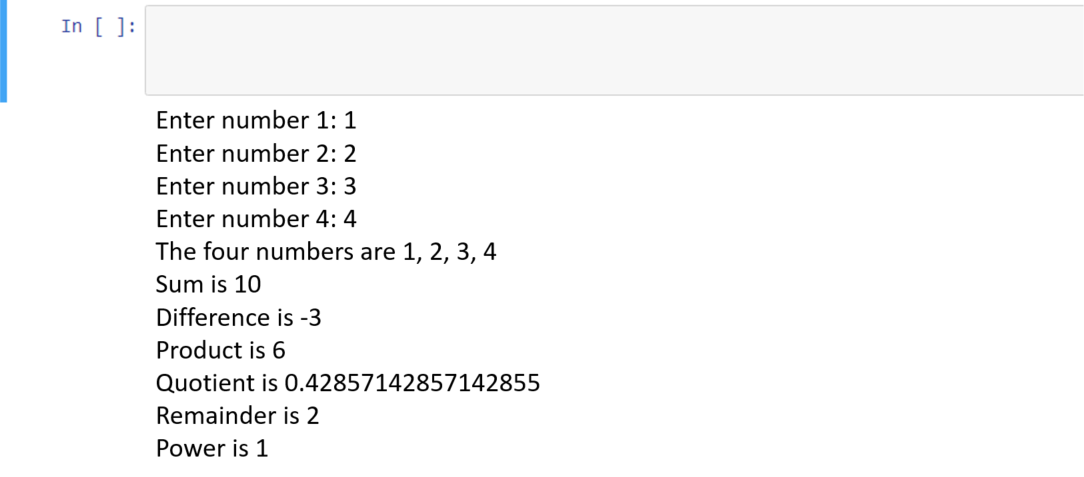
# Working with Python numeric types

### Task 4-1: Perform basic mathematical calculations

Write a Python program that performs the fundamental operations in mathematics: addition, subtraction, multiplication and division.

Your program should:

1. Prompt the user to enter four numbers and store them in the variables *n1*,*n2*,*n3* and *n4*
2. Add up the four numbers and store the calculated value in the variable *sum*
3. Subtract *n4* from *n1* and store the calculated value in the variable *difference*
4. Multiply *n2* by *n3* and store the calculated value in the variable *product*
5. Divide the sum of *n1* and *n2* by the sum of *n3* and *n4*, and store the calculated value in the variable *quotient*
6. Find the remainder of the product of *n1* and *n2*, divided by the product of *n3* and *n4* and store in the value *remainder*
7. Calculate *n1* to the power of n2 and store in the variable *powerI*
8. Print the outputs as shown below



### Task 4-2: Mathematical functions in Python

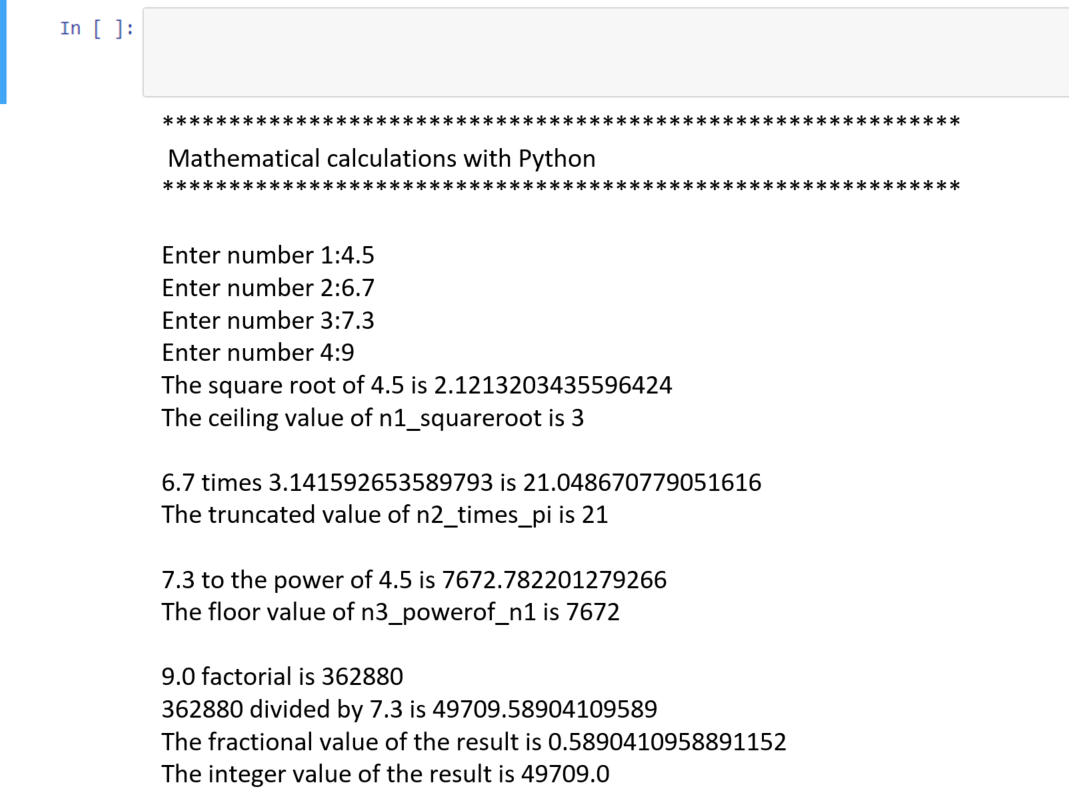
Write a Python program that performs mathematical functions in Python like sqrt, factorial, ceil, floor and so on.

Your program should:

1. Prompt the user to enter four numbers and store them in the variables n1,n2,n3 and n4
2. Perform the following operations on each variable as specified below.
3. Calculate the square root of n1 and store the value of the result to a variable n1\_squareroot. Print out the **ceiling** value of n1\_squareroot.
4. Multiply n2 by the mathematical value of **pi** and store the result to a variable n2\_times\_pi. . Print out the **truncated** value of n2\_times\_pi.
5. Calculate the value of n3 to the power of n1 and store the result to n3\_powerof\_n1. Print out the **floor** value of n3\_powerof\_n1.
6. Compute the factorial of n4 and store the result to a variable n4\_factorial. Divide n4\_factorial by n3 and store the result to n4n3. Compute the fractional and integer parts of n4n3 and print out their values.

Hint -- > Remember to convert your inputs to datatype float before performing calculations!

Your program should produce outputs similar to that shown below.



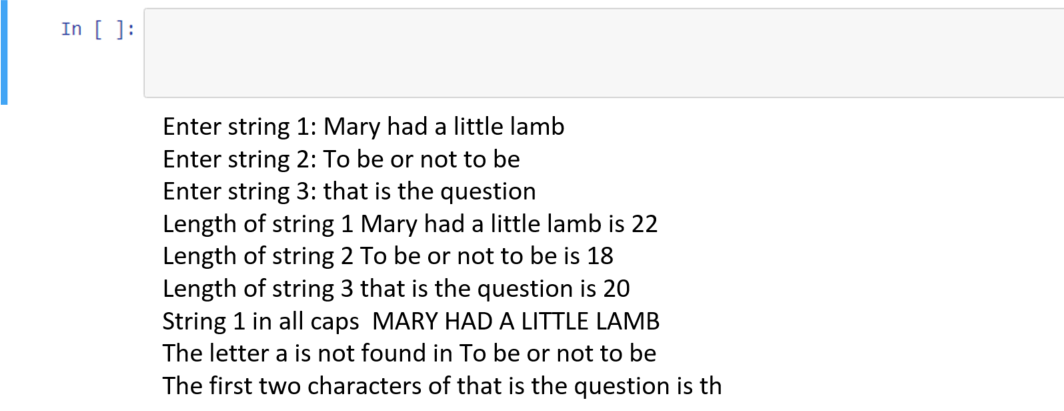
# Working with Python string types

### Task 5-1: Perform simple string operations

Write a Python program that performs simple string operations using the string class

Your program should:

1. Prompt the user to enter three string variables and store them in s1, s2 and s3
2. Print out the length of all the three strings
3. Print out s1 in all capital letters
4. Locate letter a in s2 and print out the index if found or print “The letter a is not found in s2” if not found
5. Extract the first 2 characters from s3 and print them out



### Task 5-2: Perform simple string operations (SUBMISSION REQUIRED)

Write a Python program to achieve the following:

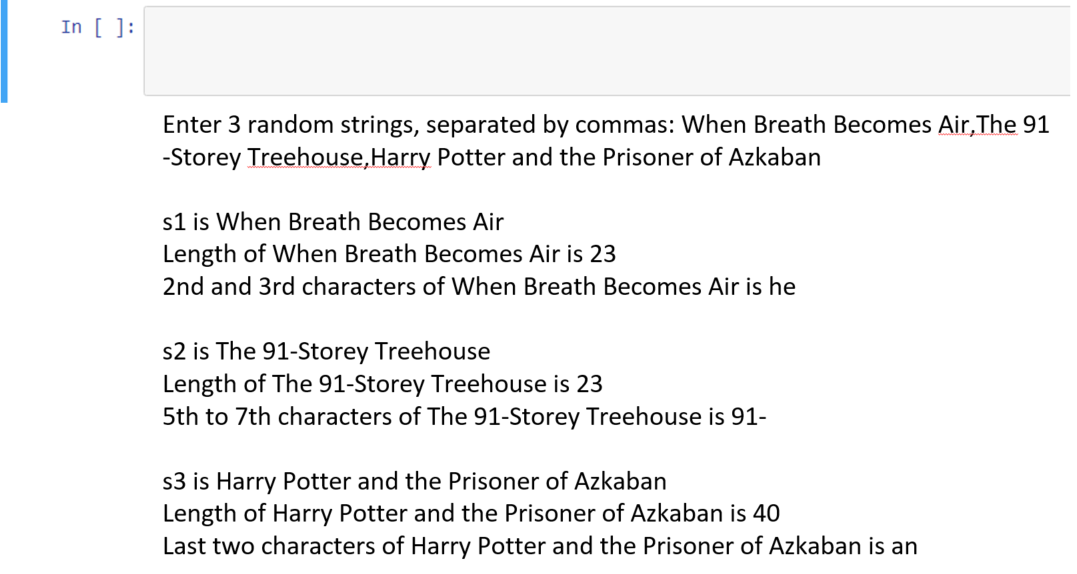
1. prompt the user to enter 3 random strings of at least 8 characters long, each separated by a comma

A sample of what the user might enter is given below

*When Breath Becomes Air,The 91-Storey Treehouse,Harry Potter and the Prisoner of Azkaban*

1. use the **split()** function in Python to separate the strings and store the separated strings into three variables, s1, s2 and s3
2. Perform the following operations on each string:

* Calculate the length of each string
* Extract the 2nd and third characters of s1, fifth to seven characters of s2 and last two characters of s3
* Your output should look similar to that shown below



# if-else statements

### Task 6-1: Odd or even

Ask the user for a number. If the input is not numeric, display an error message.

Hint: use the isnumeric() function.

If the input is numeric, check if the number is even or odd and print out an appropriate message to the user.

Hint: Use the % modulus operator which gives the remainder when a number x is divided by another number y.

Your program should display output similar to that shown below.

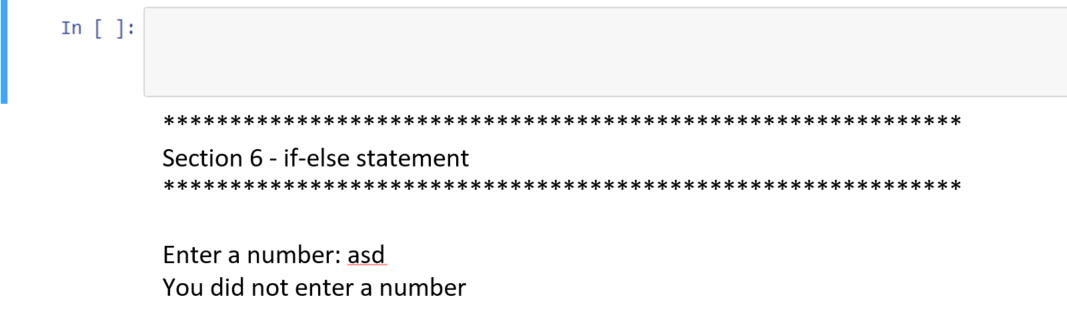


Figure 6-1: Error message when user entes a non-numeric input

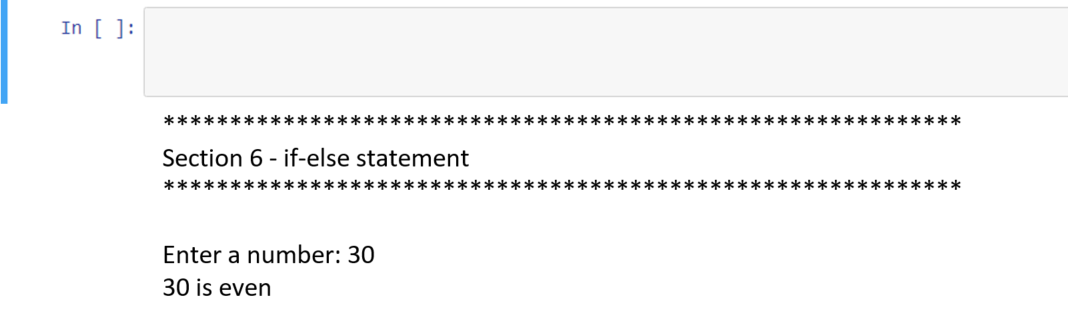


Figure 6-2: Message displayed when user enters a valid number that is even



Figure 6-3: Message displayed when user enters a valid number that is odd

### Task 6-2: Spdonalds (SUBMISSION REQUIRED)

Write a Python program that displays a breakfast menu to the user, asks him for the food he wants to buy and calculates the amount he must pay for the food

Your program should:

1. Show the user a menu consisting of the breakfast menu choices as shown in Fig. 4 and 5
2. Prompt the user to enter his breakfast choice
3. Display and error message and terminate the program if the choice is invalid
4. If breakfast choice is valid, prompt him to enter the quantity he wants to buy
5. Calculate and display the total amount he must pay

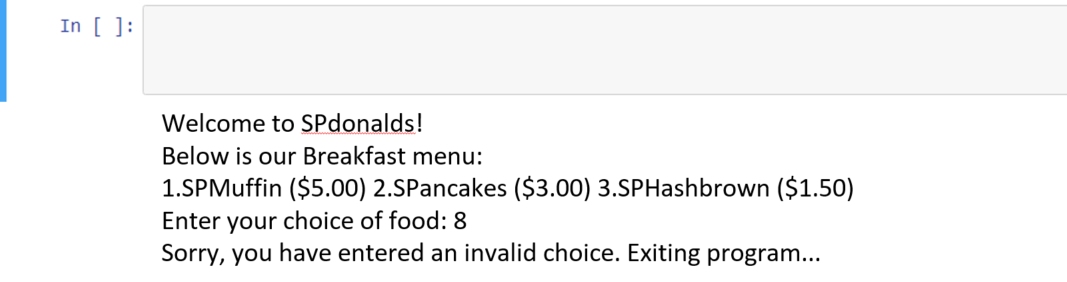


Figure 6-4: Invalid input

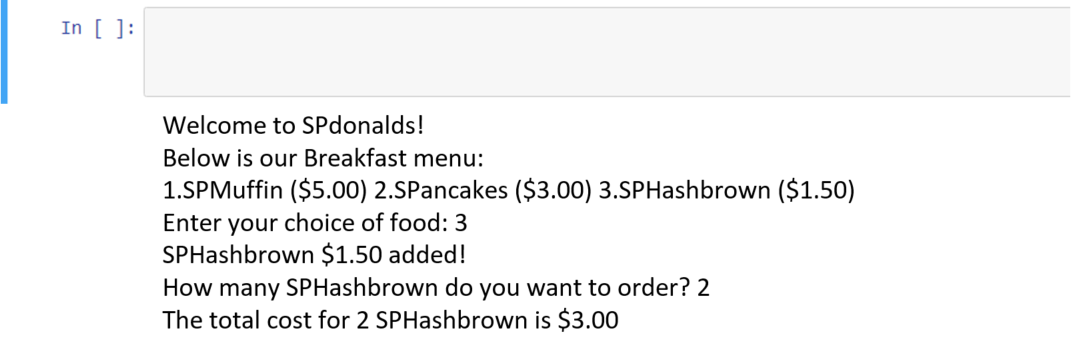
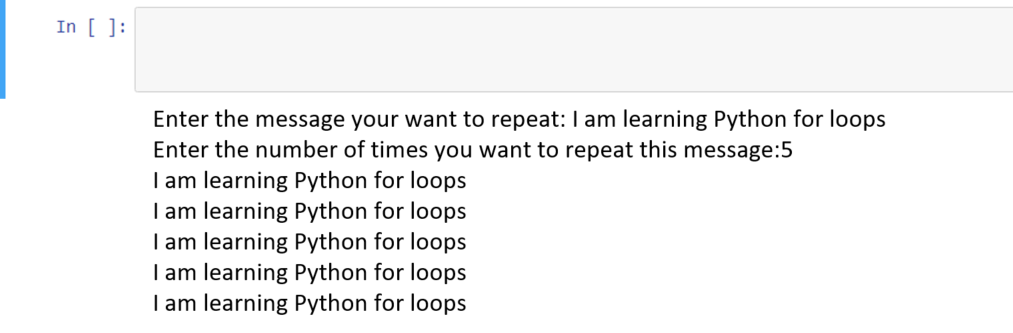


Figure 6-5: Valid input

# for loop

### Task 7-1: Repeat a user-specified message

Write a program that will ask the user for a message and the number of times they want that message displayed. Then output the message as specified.



### Task 7-2: Calculate sum of numbers within a range (SUBMISSION REQUIRED)

Write a Python program that calculates the sum of all the numbers from x to y, where x and y are numbers entered by the user.

Your program should:

1. Inform the user the purpose of the program as shown in the screenshot below
2. Prompt the user to enter the value for x and y.
3. check if 1) x and y are numeric 2) higher than zero 3) y is greater than x. If not, display an error message and terminate the program.
4. Use a **for** loop to calculate the sum of numbers from x to y and store the final value in the variable sum\_of\_numbers
5. Display sum\_of\_numbers

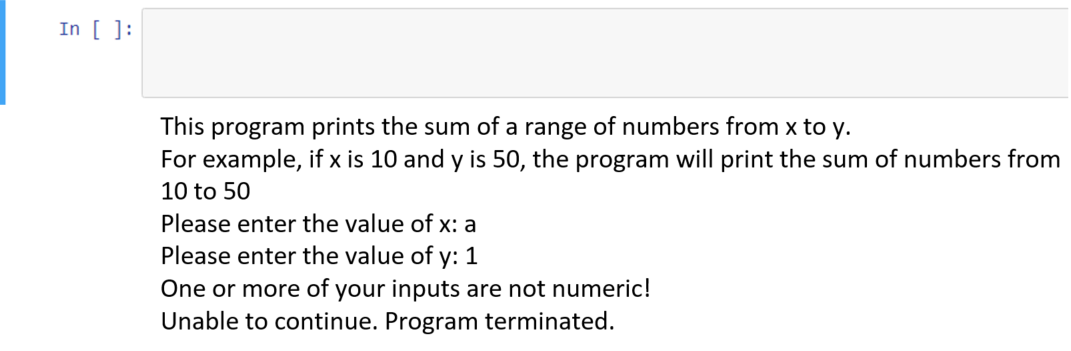


Figure 7-1: User did not enter numbers for x and y

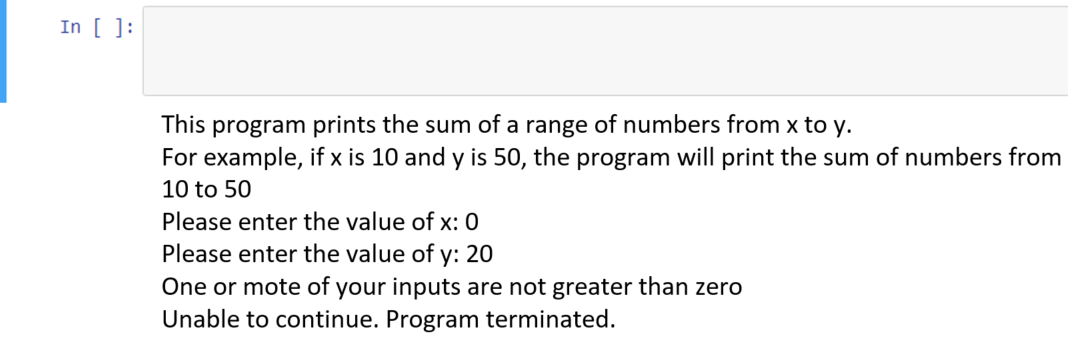


Figure 7-2: User did not enter numbers greater than zero for x

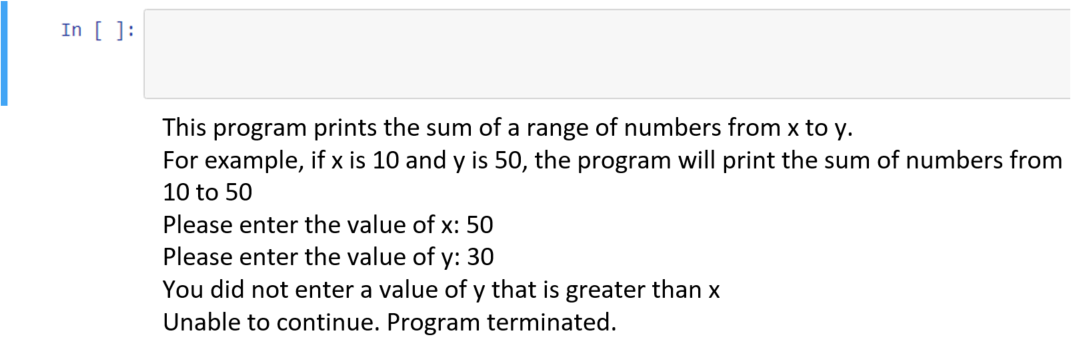


Figure 7-3: User did not enter y greater than x

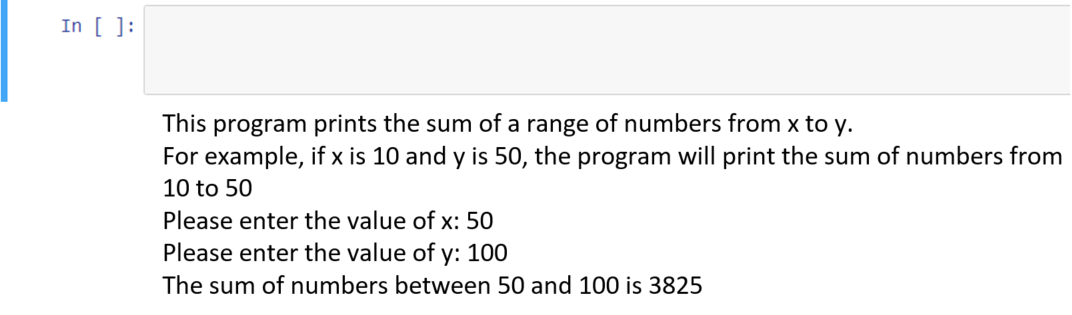
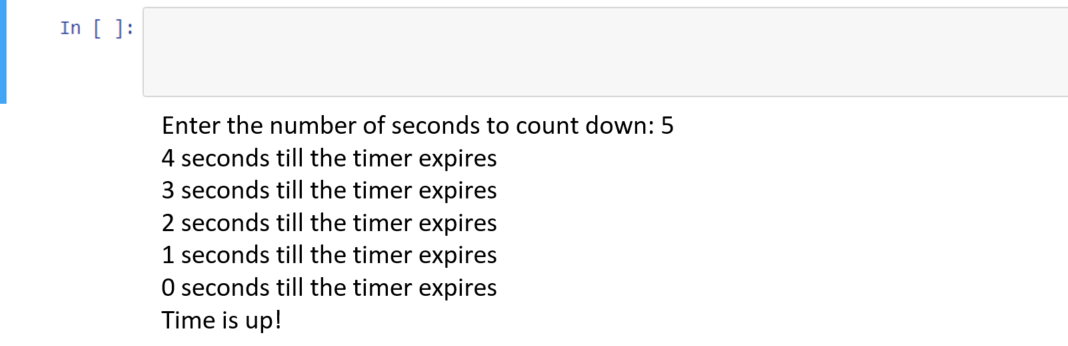


Figure 7-4: Print sum of numbers to user

# while loop (optional)

### Task 8-1: Count down timer

Create a program using **while** loop which will prompt the user to enter the number of seconds he wishes to count down, then indicate how long there is to go before time runs outs. When the time runs out it should display “Time is up!”

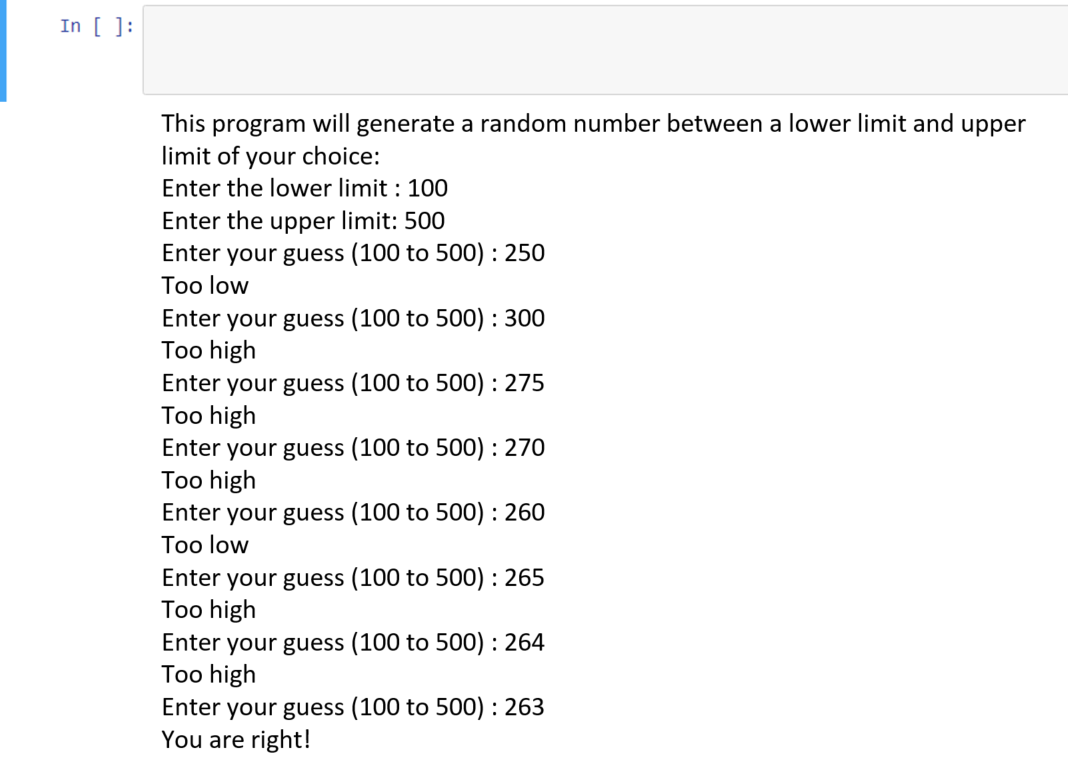


### Task 8-2: High-low number game

Write a Python ‘game’ that generates a random number and repeatedly allows the user to guess the number by giving him hints until he gets it right.

Your program should:

1. Ask the user to select the lower limit and the upper limit that the random number should be generated within
2. After obtaining the lower and upper limits, generate a random number between that range, and store it in a variable named secret\_number
3. Write a while loop that repeatedly performs the following actions until the user guesses the random number correctly
4. Prompt the user to guess the value of the number
5. If his guess is higher than the actual random number, print “Too high”
6. If his guess is lower than the actual random number, print “Too low”
7. If his guess is equal to the random number, print “You are right!” and exit the loop

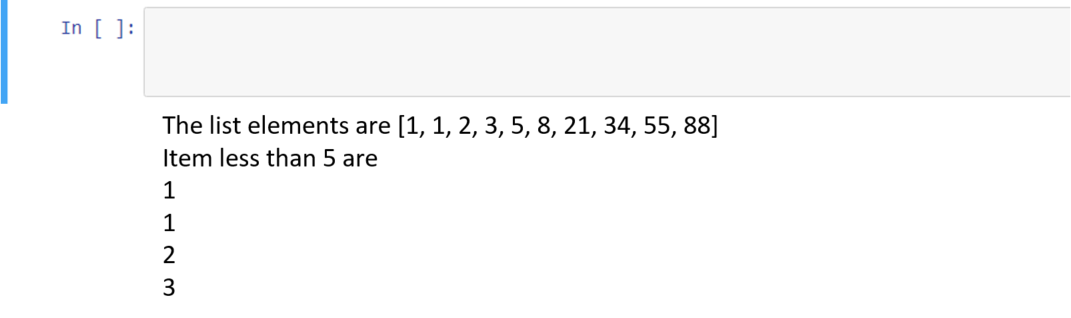


# Python Lists

### Task 9-1: Lists (Elements less than 5)

Given the following Python list, write a program that prints all the elements of the list that are less than 5.

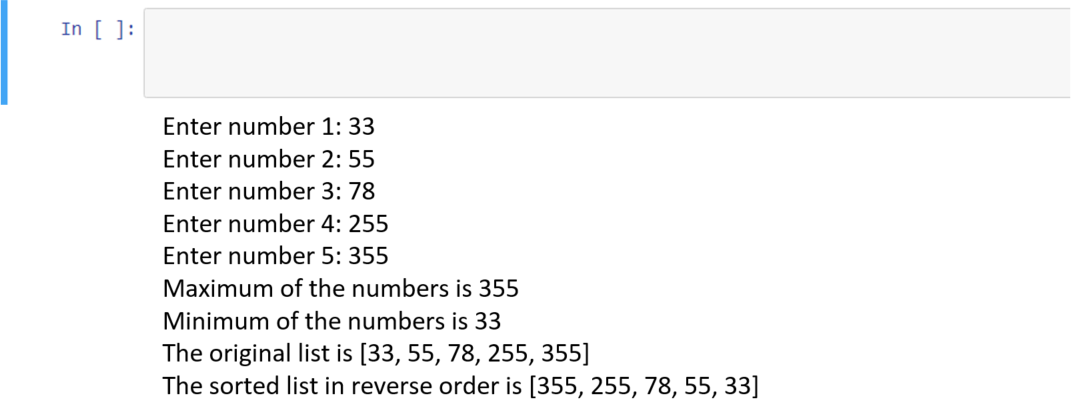
aList = [1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89]



### Task 9-2: Lists (max, min, sorted, reverse)

Write a Python program that prompts the user to enter a list of 5 numbers and prints the maximum and minimum values that were entered. In addition, print the list sorted in reverse order (highest numbers first).

Your output should look similar to that below.



### Task 9-3: List Slicing (SUBMISSION REQUIRED)

Given the three Python lists below, provide the code that will retrieve the required elements in each list.

list\_1 = [300,50,80,90,199,800,74,33]

list\_2 = ['Apple','Banana','Durian','Grapes','Papaya','Watermelon']

list\_3 = [0.0, 1.1, 2.2, 3.3, 4.4, 5.5, 6.6, 7.7, 8.8,9.9]

1. The numbers 800, 74 and 33 in list\_1
2. The last two elements in list\_2
3. All the elements in list\_3 except the first and the last

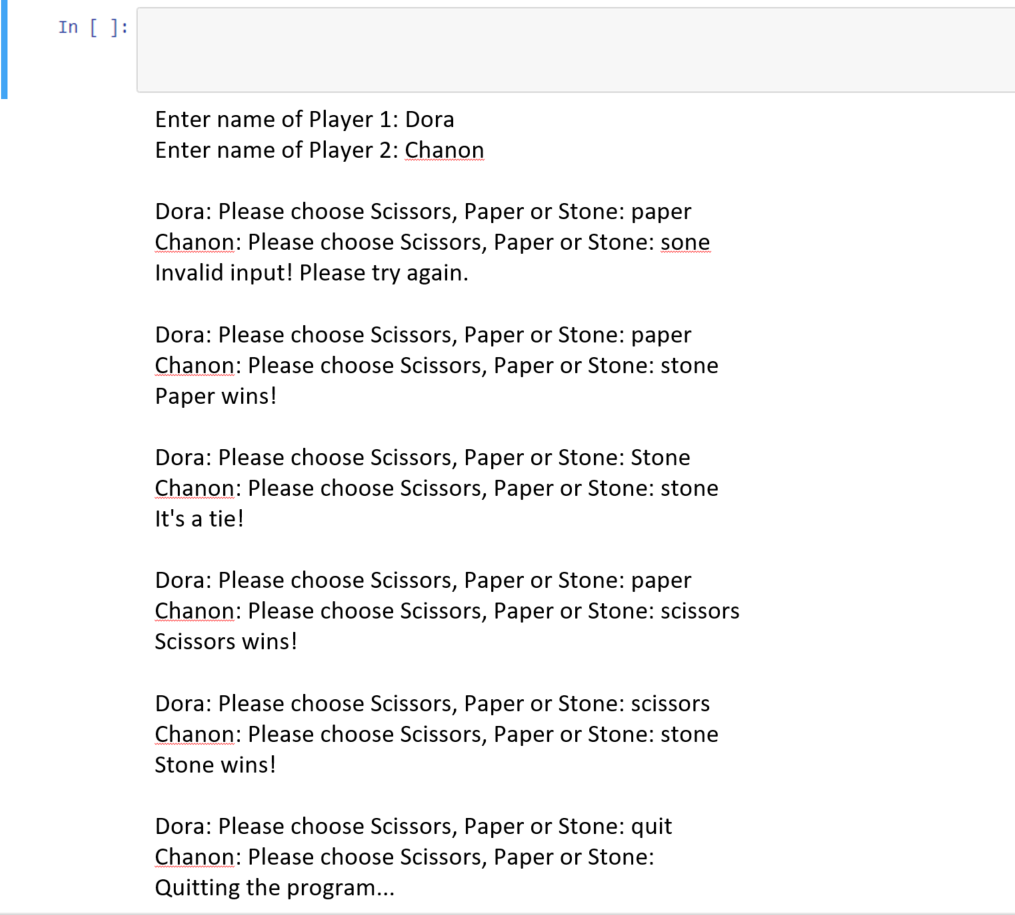
# Python functions

### Task 10-1: Scissors, Paper, Stone

Write a Python function compare(input1, input2) that takes in two string inputs which can be one of these values: Scissors, Paper or Stone. Your function should return two outputs: a message and a True or False value as follows:

|  |  |
| --- | --- |
| * If the two inputs are the same, e.g. both are “Scissors”, return “It’s a tie”, False * If input1 is “Scissors” and input2 is “Paper”, return “Scissors wins!”, False * If input1 is “Scissors” and input2 is “Stone”, return “Stone wins!”, False * If input1 is “Paper” and input2 is “Scissors”, return “Scissors wins!”, False * If input1 is “Paper” and input2 is “Stone”, return “Paper wins!”, False | * If input1 is “Stone” and input2 is “Scissors”, return “Stone wins!”, False * If input1 is “Stone” and input2 is “Paper”, return “Paper wins!”, False * If input1 is “Quit” or input2 is “Quit”, return “Quitting the program”, True * If either of input1 or input2 is not one of these values: Scissors, Paper, Stone, Quit, then return “Invalid input. Please try again”, False |

Using this function, create a Python program that simulates a two-player Scissors-Paper-Stone game that keeps repeating until any one of the user keys in a “quit” as his choice. A sample run is shown below.



### Task 10-2: Odd and Even (SUBMISSION REQUIRED)

Write a Python function that takes in a list of numbers, and outputs them into two lists, one consisting of even numbers, and the other odd numbers.

Your program should:

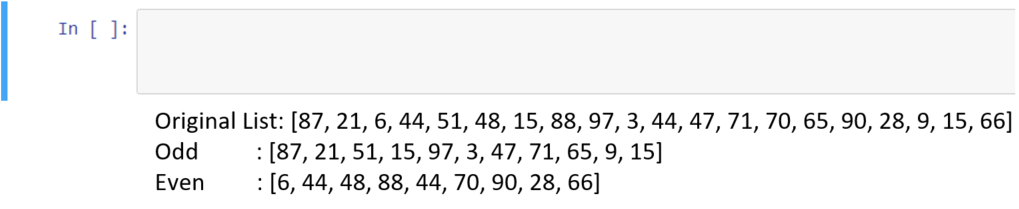
1. Generate a list with 20 random numbers in the range of 1 to 100 and store them in the variable original\_list. Hint: You can generate a list of numbers using list comprehension and the random.randint() function

Example:

# Generate 1000 numbers from 1 to 10

numbers = [random.randint(1,10) for x in range(0,1000)]

1. Define a function called ***oddandeven(numbers\_list)*** that takes in one parameter of type list containing a list of numbers and returns two outputs of two lists, one that contains even numbers, and the other that contains odd numbers
2. After writing your function, call the ***oddandeven(numbers\_list)*** function, passing in original\_list as the parameter, and storing the outputs of the function as even\_numbers and odd\_numbers respectively
3. Print the contents of even\_numbers and odd\_numbers as shown below



**-- End of Lab --**