# Functions in Python: Introduction

### Contents

## Functions

In [2]:
```
print('My name is Abdulaziz')
print('I live in Riyadh')
print('I am a data analyst and I love what I do!')
```
```
My name is Abdulaziz
I live in Riyadh
I am a data analyst and I love what I do!
```

In [29]:
```
# How to write a function
# first, start with "def" and write the name of the fuction ending with "():"
# second, write your code after 4 spaces or your code will not work
def function_name():
    #write your code here
    print('The right way to create a function')
```

In [3]:
```
# function to print
def my_introduction():
    print('My name is Abdulaziz')
    print('I live in Riyadh')
    print('I am a data analyst and I love what I do!')
```

In [8]:
```
# the way to execute a function
my_introduction()
```
```
My name is Abdulaziz
I live in Riyadh
I am a data analyst and I love what I do!
```

In [10]:
```
# hello_world function will not be executed, only the outside print will execute
def hello_world():
    print('Welcome to Paython')
print('This is outside the function')
```
```
This is outside the function
```

In [11]:
```
hello_world()
```
```
Welcome to Paython
```

In [13]:
```
# functions not defined will make errors
# NameError: name 'my_new_function' is not defined

my_new_function()
```
```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-13-47dd35cc22f3> in <module>
      2 # NameError: name 'my_new_function' is not defined
      3 
----> 4 my_new_function()

NameError: name 'my_new_function' is not defined
```

In [15]:
```
# wrong code to write a function
# IndentationError: expected an indented block
def will_not_work_function():
print('This identation is all wrong')
```
```
  File "<ipython-input-15-60c0c1dead2c>", line 4
    print('this identation is all wrong')
        ^
IndentationError: expected an indented block
```

In [18]:
```
def will_work_function():
    print('This identation is all right')
    # condition is right so the if statement will work
    if 10 > 5:
        print('Well 10 is greater than 5')
```

In [20]:
```
will_work_function()
```
```
This identation is all right
Well 10 is greater than 5
```

In [23]:
```
# for loop function
def a_more_complicated_function():
    for i in range(10):
        print('i is now:',i)
```

In [24]:
```
a_more_complicated_function()
```
```
i is now: 0
i is now: 1
i is now: 2
i is now: 3
i is now: 4
i is now: 5
i is now: 6
i is now: 7
i is now: 8
i is now: 9
```

In [25]:
```
def _Functions_Can_Be_NamedLikeThis_123():
    print('This works!')

_Functions_Can_Be_NamedLikeThis_123()
```
```
This works!
```

In [30]:
```
# functions cannot started with numbers
# SyntaxError: invalid syntax
def 123this_does_not_work():
    print('This does not work')
```
```
  File "<ipython-input-30-2c54b69ee67e>", line 3
    def 123this_does_not_work():
          ^
SyntaxError: invalid syntax
```

In [32]:
```
# documnetation is very important with the function is complex
# in this way you can right many lines
def documented_function():
    """This function does something that is will documented"""
    print('hello')
```

In [34]:
```
# running a function without () will return metadata about the function
## __main__ prefix that tell you this is a function
documented_function
```
Out[34]:
```
<function __main__.documented_function()>
```

In [36]:
```
# __doc__ this attirbute allows you to see the documentation of the function
documented_function.__doc__
```
Out[36]:
```
'This function does something that is will documented'
```

In [37]:
```
# functions are objects, which means you can assign functions to a variable
another_function = documented_function
```

In [39]:
```
# you can see here this variable has a referance from the original object (documented_function)
another_function
```
Out[39]:
```
<function __main__.documented_function()>
```

In [41]:
```
# you can use it as documented_function function
another_function()
```
```
hello
```

In [42]:
```
documented_function()
```
```
hello
```

In [50]:
```
name = 'Abdulaziz'
city = 'Riyadh'
```

```
In [45]: # variables defined outside a function can be used inside the function because the Global scope in python
         def introduction():
             print('My name is: ',name)
             print('I live in: ',city)
```

```
In [51]: introduction()

         My name is:  Abdulaziz
         I live in:  Riyadh
```

```
In [57]: # If you update variables, the functions will take the new updates
         name = 'Mohammed'
         city = 'Al Quwaiiyah'
```

```
In [58]: introduction()

         My name is:  Mohammed
         I live in:  Al Quwaiiyah
```

```
In [62]: # adding a string to an integer will result in an error
         # TypeError: can only concatenate str (not "int") to str
         'a' + 2

         ---------------------------------------------------------------------------
         TypeError                                 Traceback (most recent call last)
         <ipython-input-62-20c5c68a54f9> in <module>
               1 # adding a string to an integer will result in an error
               2 # TypeError: can only concatenate str (not "int") to str
         ----> 3 'a' + 2

         TypeError: can only concatenate str (not "int") to str
```

```
In [63]: # you can make operations in the same type
         'a' + 'b'

Out[63]: 'ab'
```

```
In [64]: 1 + 1

Out[64]: 2
```

## Input Arguments

```
In [71]: # this is a function with input argument
         def my_introduction_2(name):
             print('My name is',name)
```

```
In [72]: # to execute my_introduction_2 function you must fill the argument inside parentheses
         my_introduction_2('Abdulaziz')

         My name is Abdulaziz
```

```
In [73]: # the same code above but with 2 input arguments
         def my_introduction_3(name,city):
             print('My name is',name)
             print('I live in',city )
```

```
In [75]: # be careful when input arguments, it must be in the same order
         my_introduction_3('Abdulaziz','Riyadh')

         My name is Abdulaziz
         I live in Riyadh
```

```
In [82]: # input argument in a function that does not have an argument will result in an error
         #TypeError: my_introduction() takes 0 positional arguments but 1 was given

         my_introduction('Abdulaziz')

         ---------------------------------------------------------------------------
         TypeError                                 Traceback (most recent call last)
         <ipython-input-82-0383d4cc6679> in <module>
               2 #TypeError: my_introduction() takes 0 positional arguments but 1 was given
               3
         ----> 4 my_introduction('Abdulaziz')

         TypeError: my_introduction() takes 0 positional arguments but 1 was given
```

```
In [83]: # the same when input more argument than the specified in the function
         # TypeError: my_introduction_2() takes 1 positional argument but 2 were given

         my_introduction_2('Abdulaziz','Riyadh')

         ---------------------------------------------------------------------------
         TypeError                                 Traceback (most recent call last)
         <ipython-input-83-93417a0fa301> in <module>
               2 # TypeError: my_introduction_2() takes 1 positional argument but 2 were given
               3
         ----> 4 my_introduction_2('Abdulaziz','Riyadh')

         TypeError: my_introduction_2() takes 1 positional argument but 2 were given
```

```
 In [1]: # function to square a number (integer or float)
         # multiplication does not work with string
         def square(x):
             print('The square of',x,'is',x*x)
```

```
 In [2]: square(2)

         The square of 2 is 4
```

```
 In [3]: square(2.2)

         The square of 2.2 is 4.840000000000001
```

```
 In [4]: square('Abdulaziz')

         ---------------------------------------------------------------------------
         TypeError                                 Traceback (most recent call last)
         <ipython-input-4-38df5b7f393b> in <module>
         ----> 1 square('Abdulaziz')

         <ipython-input-1-3ac69ae09e30> in square(x)
               1 def square(x):
         ----> 2     print('The square of',x,'is',x*x)

         TypeError: can't multiply sequence by non-int of type 'str'
```

```
 In [5]: num = 25
```

```
 In [7]: # you can pass variables that had integer or float data type
         square(num)

         The square of 25 is 625
```

```
In [10]: # this call square function twice
         another_num = 100

         square(num)
         square(another_num)

         The square of 25 is 625
         The square of 100 is 10000
```

```
In [11]: # This is an example of bad function
         salary = 3000
         expense = 900

         def my_savings(a,b):
             print('My total savings:', salary - expense)
```

```
In [12]: my_savings(3000,900)

         My total savings: 2100
```

```
In [13]: # why the answer not 500 ?
         # the problem is we assign salary and expense variables inside the function, let's correct the function next
         my_savings(1000,500)

         My total savings: 2100
```

```
In [16]: # the function should contain the arguments variables, not outside variables
         def my_actual_savings(a,b):
             print('My total savings:', a - b)
```

```
In [17]: my_actual_savings(1000,500)

         My total savings: 500
```

```
In [18]: # note that when the arguments' names have the same variables names outside the function, the arguments' names will be used
         def calculate_savings(salary,expense):
             print('My total savings:', salary - expense)
```

In [20]:
```python
# here you can see, calculate_savings function did not use the salary and expense outside the function
calculate_savings()
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-20-b923a2a0d72c> in <module>
      1 # here you can see, calculate_savings function did not use the salary and expense outside the function
----> 2 calculate_savings()

TypeError: calculate_savings() missing 2 required positional arguments: 'salary' and 'expense'
```

In [22]:
```python
# do not be confused, this will use the global variables that specified before
calculate_savings(salary,expense)
```

```
My total savings: 2100
```

In [24]:
```python
# here it is the same my_actual_savings function but with different arguments names
calculate_savings(2000,1000)
```

```
My total savings: 1000
```

In [25]:
```python
# this function to print something for many times you specify
def print_many_times(string, times):
    for i in range(times):
        print(string)
```

In [27]:
```python
print_many_times('I love Data Analysis',3)
```

```
I love Data Analysis
I love Data Analysis
I love Data Analysis
```

In [29]:
```python
# you will get an error if you input wrong data type
print_many_times(3,'I love Data Analysis')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-29-cc683421738f> in <module>
      1 # you will get an error if you input wrong data type
----> 2 print_many_times(3,'I love Data Analysis')

<ipython-input-25-cc4c57d1c0f0> in print_many_times(string, times)
      1 # this function to print something for many times you specify
      2 def print_many_times(string, times):
----> 3     for i in range(times):
      4         print(string)

TypeError: 'str' object cannot be interpreted as an integer
```

In [35]:
```python
# it is very recommended to write documentation to explain the function
def print_many_times_with_doc(string, times):
    """
    This function to print something for many times you specify

    First argument takes a string data type
    Second argument takes an integer data type (float data type do not work)

    """
    for i in range(times):
        print(string)
```

In [36]:
```python
print_many_times_with_doc('I love Jupyter',3)
```

```
I love Jupyter
I love Jupyter
I love Jupyter
```

In [37]:
```python
# to read the documentation it is not good to read it this was
print_many_times_with_doc.__doc__
```

Out[37]:
```
'\n    This function to print something for many times you specify \n    \n    First argument takes a string data type\n    Second argument takes an integer data type (float data type do not work)\n    \n    '
```

In [38]:
```python
# you should print the documentation
print(print_many_times_with_doc.__doc__)
```

```

    This function to print something for many times you specify

    First argument takes a string data type
    Second argument takes an integer data type (float data type do not work)

```

In [39]:
```python
# function to print higher number

def print_higher_number(a,b):
    if a > b:
        print('Higher number is',a)
    else:
        print('Higher number is',b)
```

In [40]:
```python
print_higher_number(10,5)
```

```
Higher number is 10
```

In [43]:
```python
print_higher_number(5,7)
```

```
Higher number is 7
```

In [44]:
```python
# function to print higher number with error

def print_higher_number_with_error(a,b):
    if a > b:
        print('Higher number is',a)
    else:
        print('Higher number is',b)

        result = b + 'a'
```

In [45]:
```python
print_higher_number_with_error(50,20)
```

```
Higher number is 50
```

In [47]:
```python
# testing your code parts
print_higher_number_with_error(20,40)
```

```
Higher number is 40

---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-47-03aa7b3c0f7b> in <module>
      1 # testing your code parts
----> 2 print_higher_number_with_error(20,40)

<ipython-input-44-55eba588f6cb> in print_higher_number_with_error(a, b)
      7         print('Higher number is',b)
      8 
----> 9         result = b + 'a'
     10 

TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

In [51]:
```python
# function that multiply 3 numbers
def multiply(num_1, num_2, num_3):
    print('Multiplication result:',num_1 * num_2 * num_3)
```

In [52]:
```python
# in multiplication you can multiply integers with floats
multiply(2,5.3,8)
```

```
Multiplication result: 84.8
```

In [54]:
```python
# you can not pass 4 arguments because the function only specified 3 arguments
multiply(2,5.3,7,20)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-54-77eb29d9fff0> in <module>
      1 # you can not pass 4 arguments because the function only specified 3 arguments
----> 2 multiply(2,5.3,7,20)

TypeError: multiply() takes 3 positional arguments but 4 were given
```

In [56]:
```python
# function calculate the length of a list
def length(some_list):
    count = 0
    for element in some_list:
        count += 1

    print('The length of the list is',count)
```

In [57]:
```python
num_list = [4,8,12,20,25,30,45]

length(num_list)
```

```
The length of the list is 7
```

```
In [25]: teams_list = ['Juventus','Milan','Napoli','Roma']

         length(team_list)

---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-25-9a4f4a29b3ad> in <module>
      1 teams_list = ['Juventus','Milan','Napoli','Roma']
      2
----> 3 length(team_list)

NameError: name 'length' is not defined
```

```
In [61]: # you can specify the list in the argument

         length([5.2,8,16,7])

The length of the list is 4
```

```
In [87]: # len return only the number of the list
         num_teams = len(teams_list)

         num_teams

Out[87]: 4
```

```
In [92]: # this return the function
         num_teams = length(teams_list)

The length of the list is 4
```

```
In [93]: num_teams
```

```
In [86]: # what happens? it returns nothing
         print(num_teams)

None
```

## Return Values

```
In [69]: def subtract(num_1, num_2):

             result = num_1 - num_2
```

```
In [70]: # nothing happend
         subtract(10, 7.7)
```

```
In [74]: # assign the function to variable and nothing happend
         r = subtract(100,50)
```

```
In [75]: # None is a special value in Python that indicates no value or nothing
         print(r)

None
```

```
In [76]: # the problem is the function we had not to specify a return value
         # be default it rerutn none-type
         type(r)

Out[76]: NoneType
```

```
In [78]: def subtract_returns_none_be_default(num_1, num_2):

             result = num_1 - num_2
             # "return" is a keyword that return a value from a function
             return None
```

```
In [80]: r = subtract_returns_none_be_default(100,50)
         print(r)

None
```

```
In [81]: def subtract_return_a_result(num_1, num_2):

             result = num_1 - num_2

             return result
```

```
In [83]: r = subtract_return_a_result(100,50)
         print(r)

50
```

```
In [88]: # function calculate the length of a list with return
         def length_with_return(some_list):
             count = 0
             for element in some_list:
                 count += 1

             print('The length of the list is',count)

             return count
```

```
In [95]: num_teams = length_with_return(teams_list)

The length of the list is 4
```

```
In [98]: # now it works after we put return to our code
         num_teams

Out[98]: 4
```

```
In [99]: print(num_teams)

4
```

```
In [111]: # this function find the maximum number in a list
          def find_max_in_list(some_list):
              # variable to hold the maximum element in the list starting from the first element in the list "some_list[0]"
              max_element = some_list[0]
              # specify the length of the list
              length = len(some_list)
              # for loop to the entire list to check every element in the list
              for i in range(1,length):
                  # if statement to compare elements and hold the maximum element
                  if some_list[i] > max_element:
                      max_element = some_list[i]

              return max_element
```

```
In [112]: num_list_2 = [10,20,30,40,50,100]
```

```
In [113]: max_element = find_max_in_list(num_list_2)

          print(max_element)

100
```

```
In [114]: num_list_2.append(1897)
```

```
In [115]: max_element = find_max_in_list(num_list_2)

          print(max_element)

1897
```

```
In [116]: # if you do not specify anything in return it will returns none type
          def empty_return(x,y):
              total = x + y

              return
```

```
In [118]: print(empty_return(10,5))

None
```

```
In [119]: # we can return multiple values
          def add_sub(x,y):

              add_result = x+y
              sub_result = x-y

              return add_result,sub_result
```

```
In [121]: add_sub(5,4)

Out[121]: (9, 1)
```

```
In [126]: # python can assign results in different variables
          result_1 , result_2 = add_sub(5,4)
```

```
In [127]: result_1

Out[127]: 9
```

In [128]:
```
result_2
```
Out[128]:
```
1
```

In [135]:
```
# dash "_" indicates that it will ignore the second return from the function
result_1 , _ = add_sub(10,8)
```

In [136]:
```
result_1
```
Out[136]:
```
18
```

In [140]:
```
# dash "_" indicates that it will ignore the first return from the function
_ , result_2 = add_sub(10,8)
```

In [141]:
```
result_2
```
Out[141]:
```
2
```

In [142]:
```
def positive_or_negative(num):

    if num > 0 :
        return 'Positive!'
    else:
        return 'Zero or negative!'
```

In [143]:
```
positive_or_negative(10)
```
Out[143]:
```
'Positive!'
```

In [144]:
```
positive_or_negative(-10)
```
Out[144]:
```
'Zero or negative!'
```

In [145]:
```
def positive_negative_zero(num):

    if num > 0 :
        return 'Positive!'
    elif num < 0:
        return 'Negative!'
    else:
        return 'Zero'
```

In [147]:
```
positive_negative_zero(7)
```
Out[147]:
```
'Positive!'
```

In [148]:
```
positive_negative_zero(-7)
```
Out[148]:
```
'Negative!'
```

In [149]:
```
positive_negative_zero(0)
```
Out[149]:
```
'Zero'
```

In [150]:
```
def positive_negative_zero_forgotreturn(num):

    if num > 0 :
        return 'Positive!'
    elif num < 0:
        return 'Negative!'
```

In [152]:
```
# this return none type
# be careful
positive_negative_zero_forgotreturn(0)
```

In [154]:
```
# in our code above we specified that the function must have one item in the list
empty_list = []

find_max_in_list(empty_list)
```
```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-154-a9ca619792c8> in <module>
      2 empty_list = []
      3 
----> 4 find_max_in_list(empty_list)

<ipython-input-111-f6edf39554ff> in find_max_in_list(some_list)
      2 def find_max_in_list(some_list):
      3     # variable to hold the maximum element in the list starting from the first element in the list "some_list[0]"
----> 4     max_element = some_list[0]
      5     # specify the length of the list
      6     length = len(some_list)

IndexError: list index out of range
```

In [155]:
```
#, in this case, we can return none type from the beginning before executing the entire code
def find_max_in_list(some_list):

    if len(some_list) == 0:
        print('Zero element list!')

        return None

    max_element = some_list[0]

    length = len(some_list)

    for i in range(1,length):

        if some_list[i] > max_element:
            max_element = some_list[i]

    return max_element
```

In [156]:
```
find_max_in_list(empty_list)
```
```
Zero element list!
```

In [160]:
```
def find_first_capital_letter(some_string):

    capital_letter = None

    for ch in some_string:
        if ch.upper() == ch and ch != " ":
            capital_letter = ch
            break

    if capital_letter is None:
        return 'No capital letters found'
    else:
        return 'First capital letter ' + capital_letter
```

In [163]:
```
find_first_capital_letter('how Are you')
```
Out[163]:
```
'First capital letter A'
```

In [164]:
```
find_first_capital_letter('how are you')
```
Out[164]:
```
'No capital letters found'
```

In [165]:
```
def create_dictionary_representation(name, age, occupation):

    dictionary = {
        'name': name,
        'age': age,
        'occupation': occupation
    }

    return dictionary
```

In [166]:
```
info_dictionary = create_dictionary_representation('Abdulaziz',27,'Data Analyst')
```

In [169]:
```
info_dictionary
```
Out[169]:
```
{'name': 'Abdulaziz', 'age': 27, 'occupation': 'Data Analyst'}
```

In [170]:
```
def generate_list(name,num_elements):

    return_list = []

    for i in range(num_elements):
        return_list.append(name)

    return return_list
```

In [173]:
```
some_list = generate_list('Abdulaziz',4)
```

In [174]:
```
some_list
```
Out[174]:
```
['Abdulaziz', 'Abdulaziz', 'Abdulaziz', 'Abdulaziz']
```

In [175]:
```
def generate_list(name,num_elements):

    print('Generate list using list comprehension')
    return_list = [name for i in range(num_elements)]

    return return_list
```

In [177]:
```
some_list = generate_list('Data',3)

some_list
```

```
Generate list using list comprehension
```

Out[177]:
```
['Data', 'Data', 'Data']
```

In [179]:
```
def generate_list(name,num_elements):
    # this work even we did ot assign to a variable
    return [name for i in range(num_elements)]
```

In [181]:
```
some_list = generate_list('Moonlight',4)

some_list
```

Out[181]:
```
['Moonlight', 'Moonlight', 'Moonlight', 'Moonlight']
```

In [182]:
```
def add(a,b):
    return a+b

def sub(a,b):
    return a-b

def mul(a,b):
    return a*b

def div(a,b):
    return a/b
```

In [183]:
```
def calculate(a, b, operator):

    if operator == 'add':
        return add(a,b)

    if operator == 'sub':
        return sub(a,b)

    if operator == 'mul':
        return mul(a,b)

    if operator == 'div':
        return div(a,b)
```

In [184]:
```
calculate(10,5,'add')
```

Out[184]:
```
15
```

In [185]:
```
calculate(10,5,'mul')
```

Out[185]:
```
50
```

In [187]:
```
calculate(10,5,'div')
```

Out[187]:
```
2.0
```

## Keywords Arguments

In [191]:
```
def total_score(math,database,network,programming):

    print('Math:',math,'Database:',database,'Network:',network,'Programming:',programming)
    return math + database + network + programming
```

In [195]:
```
total_score(95,85,85,95)
```

```
Math: 95 Database: 85 Network: 85 Programming: 95
```

Out[195]:
```
360
```

In [196]:
```
# keyword arguments allow you to specify input arguments by name while invoking function

total_score(math=95,database=85,network=85,programming=95)
```

```
Math: 95 Database: 85 Network: 85 Programming: 95
```

Out[196]:
```
360
```

In [198]:
```
# you can not put any arguments name, you must use arguments names of the function
total_score(math=95,database=85,network=85,program=95)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-198-9cfdfd7ada14> in <module>
      1 # you can not put any arguments name, you must use arguments names of the function
----> 2 total_score(math=95,database=85,network=85,program=95)

TypeError: total_score() got an unexpected keyword argument 'program'
```

In [202]:
```
# python knows the positions of the arguments even if you did not specify them
# but you must have start with postional argument then keyword argument
total_score(95,85,network=85,programming=95)
```

```
Math: 95 Database: 85 Network: 85 Programming: 95
```

Out[202]:
```
360
```

In [203]:
```
# python does not understand starting with keyword argument follows positional argument
# it must be started with positional argument then followed by keyword argument
total_score(math=95,85,network=85,programming=95)
```

```
  File "<ipython-input-203-281e42b1074d>", line 3
    total_score(math=95,85,network=85,programming=95)
                       ^
SyntaxError: positional argument follows keyword argument
```

In [205]:
```
# you can change the positions with keyword arguments
total_score(programming=95,database=85,network=85,math=95)
```

```
Math: 95 Database: 85 Network: 85 Programming: 95
```

Out[205]:
```
360
```

In [207]:
```
# in this case, there are multiple values for argument 'math'
total_score(95,database=85,network=85,math=95)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-207-b438f1ae3bcb> in <module>
      1 # in this case, there are multiple values for argument 'math'
----> 2 total_score(95,database=85,network=85,math=95)

TypeError: total_score() got multiple values for argument 'math'
```

In [209]:
```
# in this case, keyword argument repeated
total_score(math=95,database=85,network=85,math=95)
```

```
  File "<ipython-input-209-28662af26e67>", line 2
    total_score(math=95,database=85,network=85,math=95)
                                              ^
SyntaxError: keyword argument repeated
```

In [213]:
```
def print_student_detail(name,university,math,database,network,programming):

    total = math + database + network + programming

    print('Name:',name)
    print('University:',university)
    print('Score:',total)
```

In [212]:
```
print_student_detail('Abdulaziz','SEU',
                    math=95,database=85,network=85,programming=95)
```

```
Name:  Abdulaziz
University:  SEU
Score:  360
```

In [214]:
```
print_student_detail('SEU','Abdulaziz',
                    math=95,database=85,network=85,programming=95)
```

```
Name: SEU
University: Abdulaziz
Score: 360
```

In [216]:
```
print_student_detail(university='SEU',name='Abdulaziz',
                    math=95,database=85,network=85,programming=95)
```

```
Name: Abdulaziz
University: SEU
Score: 360
```

In [221]:
```
# sorted function is used to sort a list
sorted(num_list)
```

Out[221]:
```
[4, 8, 12, 20, 25, 30, 45]
```

```
In [222]: sorted(num_list,reverse=False)

Out[222]: [4, 8, 12, 20, 25, 30, 45]
```

```
In [223]: sorted(num_list,reverse=True)

Out[223]: [45, 30, 25, 20, 12, 8, 4]
```

```
In [224]: print('Abdulaziz','Mohammed','Abdullah')

          Abdulaziz Mohammed Abdullah
```

```
In [230]: print('Abdulaziz','Mohammed','Abdullah',sep='|')

          Abdulaziz|Mohammed|Abdullah
```

```
In [235]: print('Abdulaziz','Mohammed','Abdullah',sep='|',end='*****')

          Abdulaziz|Mohammed|Abdullah*****
```

## Default Arguments

```
In [236]: def print_student_detail(name,university,
                                   math,database,network,programming,
                                  enrolled):

              total = math + database + network + programming

              print('Name:',name)
              print('University:',university , 'Enrolled',enrolled)
              print('Score:',total)
```

```
In [237]: print_student_detail('Abdulaziz','SEU',
                              math=95,database=85,network=85,programming=95,
                              enrolled=True)

          Name: Abdulaziz
          University: SEU Enrolled True
          Score: 360
```

```
In [238]: print_student_detail('Abdulaziz','SEU',
                              math=95,database=85,network=85,programming=95,)

          ---------------------------------------------------------------------------
          TypeError                                 Traceback (most recent call last)
          <ipython-input-238-d6c2bd1908a8> in <module>
                1 print_student_detail('Abdulaziz','SEU',
          ----> 2                       math=95,database=85,network=85,programming=95,)

          TypeError: print_student_detail() missing 1 required positional argument: 'enrolled'
```

```
In [239]: def print_student_details(name,university,
                                   math,database,network,programming,
                                  enrolled=False):

              total = math + database + network + programming

              print('Name:',name)
              print('University:',university , 'Enrolled',enrolled)
              print('Score:',total)
```

```
In [240]: print_student_details('Abdulaziz','SEU',
                              math=95,database=85,network=85,programming=95,)

          Name: Abdulaziz
          University: SEU Enrolled False
          Score: 360
```

```
In [251]: # programming is by default 50. If you do not specify the score, by default it will be 50
          def print_student_details(name,university,
                                   math,database,network,programming=50,
                                  enrolled=False):

              total = math + database + network + programming

              print('Name:',name)
              print('University:',university , 'Enrolled',enrolled)
              print('Programming:',programming)
              print('Score:',total)
```

```
In [252]: print_student_details('Abdulaziz','SEU',
                              math=95,database=85,network=85,programming=95)

          Name: Abdulaziz
          University: SEU Enrolled False
          Programming: 95
          Score: 360
```

```
In [253]: print_student_details('Abdulaziz','SEU',
                              math=95,database=85,network=85)

          Name: Abdulaziz
          University: SEU Enrolled False
          Programming: 50
          Score: 315
```

```
In [255]: # if there is no default value to an argument it will make an error
          print_student_details('Abdulaziz','SEU',
                              math=95,database=85)

          ---------------------------------------------------------------------------
          TypeError                                 Traceback (most recent call last)
          <ipython-input-255-c64091716a1d> in <module>
                1 # if there is no default value to an argument it will make an error
                2 print_student_details('Abdulaziz','SEU',
          ----> 3                       math=95,database=85)

          TypeError: print_student_details() missing 1 required positional argument: 'network'
```

```
In [257]: #you can not specify default argument then followed it non-default argument
          def print_student_details(name,university,
                                   math=50,database,network,programming=50,
                                  enrolled=False):

              total = math + database + network + programming

              print('Name:',name)
              print('University:',university , 'Enrolled',enrolled)
              print('Programming:',programming)
              print('Score:',total)

            File "<ipython-input-257-1aaec4a0de31>", line 2
              def print_student_details(name,university,
                                       ^
          SyntaxError: non-default argument follows default argument
```

```
In [258]: # the same error in the above example
          def print_student_details(name,university='SEU',
                                   math,database,network,programming=50,
                                  enrolled=False):

              total = math + database + network + programming

              print('Name:',name)
              print('University:',university , 'Enrolled',enrolled)
              print('Programming:',programming)
              print('Score:',total)

            File "<ipython-input-258-4c896bfba47e>", line 2
              def print_student_details(name,university='SEU',
                                       ^
          SyntaxError: non-default argument follows default argument
```

```
In [259]: def print_student_details(name='Abdulaziz',university='SEU',
                                   math=50,database=50,network=50,programming=50,
                                  enrolled=False):

              total = math + database + network + programming

              print('Name:',name)
              print('University:',university , 'Enrolled',enrolled)
              print('Math:',math)
              print('Database:',database)
              print('Network:',network)
              print('Programming:',programming)
              print('Score:',total)
```

In [261]:
```
# if all arguments have default values the function will work fine
print_student_details()
```
```
Name: Abdulaziz
University: SEU Enrolled False
Math: 50
Database: 50
Network: 50
Programming: 50
Score: 200
```

## Variable Length Arguments

In [1]:
```
# empty argument print nothing
print()
```

In [2]:
```
print('Abdulaziz')
```
```
Abdulaziz
```

In [3]:
```
print('Abdulaziz','Mohammed')
```
```
Abdulaziz Mohammed
```

In [4]:
```
print('Abdulaziz','Mohammed','Abdullah')
```
```
Abdulaziz Mohammed Abdullah
```

In [7]:
```
# print function accepts variable length arguments
print('Abdulaziz','Mohammed','Abdullah','Abdulrahman')
```
```
Abdulaziz Mohammed Abdullah Abdulrahman
```

In [8]:
```
def print_fn(string_1):

    print(string_1)
```

In [9]:
```
print_fn()
```
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-9-a2814f842039> in <module>
----> 1 print_fn()

TypeError: print_fn() missing 1 required positional argument: 'string_1'
```

In [12]:
```
# with default argument the function will work if you do not input an argument
def print_fn(string_1='\n'):

    print(string_1)
```

In [13]:
```
print_fn()
```

In [14]:
```
# if you input more than the specified argument in the function you will get an error
print_fn('One','Two')
```
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-14-81d06a5a1b54> in <module>
----> 1 print_fn('One','Two')

TypeError: print_fn() takes from 0 to 1 positional arguments but 2 were given
```

In [15]:
```
def print_fn(string_1='\n',string_2=''):

    print(string_1,string_2)
```

In [16]:
```
print_fn('One')
```
```
One
```

In [17]:
```
print_fn('One','two')
```
```
One two
```

In [19]:
```
# trying input three arguments and the function only specify two, this will not work
print_fn('One','two','Three')
```
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-19-e7d10162caa1> in <module>
      1 # trying input three arguments and the function only specify two, this will not work
----> 2 print_fn('One','two','Three')

TypeError: print_fn() takes from 0 to 2 positional arguments but 3 were given
```

In [21]:
```
# * before argument name indicate that the function can be invoked with any number of arguments
# *args function receives the variable numbers of arguments as a tuple
def print_fn(*args):

    args_type = type(args)

    print(args_type)
    print(args)
```

In [22]:
```
print_fn()
```
```
<class 'tuple'>
()
```

In [23]:
```
print_fn('One')
```
```
<class 'tuple'>
('One',)
```

In [24]:
```
print_fn('One','Two')
```
```
<class 'tuple'>
('One', 'Two')
```

In [26]:
```
# python is consider entire list as a single element
teams_list = ['Juventus','Milan','Napoli','Roma']

print_fn(teams_list)
```
```
<class 'tuple'>
(['Juventus', 'Milan', 'Napoli', 'Roma'],)
```

In [27]:
```
# what if you want unpack elements?
# use * before the name of the list
print_fn(*teams_list)
```
```
<class 'tuple'>
('Juventus', 'Milan', 'Napoli', 'Roma')
```

In [29]:
```
# two requires arguments and one variable length argument
def students_in_college(college, city, *students):

    print('College: ',college)
    print('City: ',city)
    print('Students: ',students)
```

In [34]:
```
# two arguments must be specified college and city.
students_in_college()
```
```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-34-d4ab75a99cee> in <module>
      1 # two arguments must be specified college and city.
----> 2 students_in_college()

TypeError: students_in_college() missing 2 required positional arguments: 'college' and 'city'
```

In [35]:
```
# students can be empty or more
students_in_college('Computer and Informatic','Riyadh')
```
```
College:  Computer and Informatic
City:  Riyadh
Students:  ()
```

In [36]:
```
students_in_college('Computer and Informatic','Riyadh','Abdulaziz')
```
```
College:  Computer and Informatic
City:  Riyadh
Students:  ('Abdulaziz',)
```

In [37]:
```
students_in_college('Computer and Informatic','Riyadh','Abdulaziz','Mohemmed')
```
```
College:  Computer and Informatic
City:  Riyadh
Students:  ('Abdulaziz', 'Mohemmed')
```

In [39]:
```
# start with keyword argument and follows it with positional argument make an error
students_in_college(college='Computer and Informatic',city='Riyadh','Abdulaziz','Mohemmed')
```

```
  File "<ipython-input-39-52f753f89ea5>", line 2
    students_in_college(college='Computer and Informatic',city='Riyadh','Abdulaziz','Mohemmed')
                                                                       ^
SyntaxError: positional argument follows keyword argument
```

In [40]:
```
def students_in_college(*students, city, college):

    print('College: ',college)
    print('City: ',city)
    print('Students: ',students)
```

In [42]:
```
# python thinks these are students names and waits for keyword arguments college and city
students_in_college('Computer and Informatic','Riyadh','Abdulaziz')
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-42-b882f6260c18> in <module>
      1 # python thinks these are students names and waits for keyword arguments college and city
----> 2 students_in_college('Computer and Informatic','Riyadh','Abdulaziz')

TypeError: students_in_college() missing 2 required keyword-only arguments: 'city' and 'college'
```

In [48]:
```
# always start with positional argument follows keyword argument
students_in_college('Abdulaziz',city='Riyadh',college='Computer and Informatic')
```

```
College:  Computer and Informatic
City:  Riyadh
Students:  ('Abdulaziz',)
```

In [49]:
```
students_in_college('Abdulaziz','Mohammed',city='Riyadh',college='Computer and Informatic')
```

```
College:  Computer and Informatic
City:  Riyadh
Students:  ('Abdulaziz', 'Mohammed')
```

In [50]:
```
# ** before argument name this pack to variable length arguments into a dictionary not a tuple

def student_details(**kwargs):

    print(type(kwargs))
    print(kwargs)
```

In [52]:
```
student_details()
```

```
<class 'dict'>
{}
```

In [53]:
```
student_details(name='Abdulaziz')
```

```
<class 'dict'>
{'name': 'Abdulaziz'}
```

In [54]:
```
student_details(name='Abdulaziz', age=27)
```

```
<class 'dict'>
{'name': 'Abdulaziz', 'age': 27}
```

In [55]:
```
student_details(name='Abdulaziz', age=27, college='Computer and Informatic')
```

```
<class 'dict'>
{'name': 'Abdulaziz', 'age': 27, 'college': 'Computer and Informatic'}
```

In [59]:
```
# because it is a dictionary within a function, you can iterate over the items in details argument using a for loop
def student_details(**details):

    for key, value in details.items():
        print(key, value)
```

In [58]:
```
student_details(name='Abdulaziz', age=27, college='Computer and Informatic')
```

```
name Abdulaziz
age 27
college Computer and Informatic
```

In [62]:
```
details_dictionary = {'name':'Mohammed','age': 22,'college':'Computer and Informatic'}
```

In [63]:
```
# this will not work becouse it accpecting keyword argument
student_details(details_dictionary)
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-63-017c6d0eb84f> in <module>
----> 1 student_details(details_dictionary)

TypeError: student_details() takes 0 positional arguments but 1 was given
```

In [66]:
```
# but if you use ** before the dictionary. this unpack the dictionary and it works
student_details(**details_dictionary)
```

```
name Mohammed
age 22
college Computer and Informatic
```

In [78]:
```
# function checks if a particular key is present in the input dictionary. if it exists, print in the screen
def student_details(**details):

    if 'name' in details:
        print('Name: ',details['name'])

    if 'age' in details:
        print('Age: ', details['age'])

    if 'college' in details:
        print('College: ', details['college'])

    #print(details)
```

In [79]:
```
student_details(name='Abdulaziz')
```

```
Name:  Abdulaziz
```

In [80]:
```
student_details(name='Abdulaziz',college='Computer and Informatic',age=27)
```

```
Name:  Abdulaziz
Age:  27
College:  Computer and Informatic
```

In [82]:
```
# Level will not be printed but it saved in the dictionary.
student_details(name='Abdulaziz',college='Computer and Informatic',age=27, level=8)
```

```
Name:  Abdulaziz
Age:  27
College:  Computer and Informatic
```

In [86]:
```
# function print student names in tuple and college details in a dictionary
def students_in_college(*student_name,**college_details):
    print('Students--')
    for i in student_name:
        print(i)

    print()

    print('College Details')
    for key, value in college_details.items():
        print(key, value)
```

In [87]:
```
students_in_college('Abdulaziz','Mohammed', name='Saudi Electronic University', city='Riyadh')
```

```
Students--
Abdulaziz
Mohammed

College Details
name Saudi Electronic University
city Riyadh
```

Reference: Instructor at skillsoft is Janani Ravi