03_Functions

January 19, 2018

1 Functions

A fucntion is a named sequence of statements that performs a computation. A function could be either already defined (built-in) in a programming language (e.g. print()), or you can define your own function and

1.0.1 Examples of Built-in Functions

```
In [1]: min(30, 5, 7)
```

```
Out[1]: 5
```

```
In [2]: max("This is a Python workshop")
```

```
Out[2]: 'y'
```

```
In [3]: len("Python")
```

```
Out[3]: 6
```

1.0.2 Random Numbers

```
In [4]: import random # import the library random to generate random numbers

        x = random.random()  # returns a random float between 0 and 1
        print(x)
```

```
0.9420782336639401
```

```
In [5]: # Another way of defining libraries

        import random as rdm # giving the library an alias

        x = rdm.random()
        print(x)
```

```
0.7608862622892321
```

```
In [6]: # randint() takes two parameters "low" and "high" and returns an integer
        # between "low" and "high"

        rdm.randint(3, 12)
        # or
        random.randint(3,12)
```

```
Out[6]: 8
```

```
In [7]: # we can choose an element from a given sequence at random

        # define the sequence
        t = [1,5, 8, 10, 20]
        rdm.choice(t)
```

```
Out[7]: 8
```

1.0.3 Creating New Functions

```
In [8]: # A function definition specifies the name of the function followed by a sequence of
        # statements that execute when the function is called.

        def print_me():
            print("My name is Adele")
```

```
In [9]: # call the function print_me()

        print_me()
```

```
My name is Adele
```

```
In [10]: # Update the print_me() function to print a string given as an argument

         def print_me(toprint): #toprint is an argument
             print(toprint)
```

```
In [11]: print_me("My name is Beyonce")
```

```
My name is Beyonce
```

1.0.4 Creating Functions that returns a value

The previous function print_me() is called a "void" function because it is not return a value. It is simply performing an action (printing), but not returning a value.

Suppose that we want to create a new function that takes two arguments "a", and "b", and returns the double of their sum i.e. 2*(a+b). Let's call this function "bing()".

```
In [12]: def bing(a, b):
             result = 2*(a+b)
             return result

         # if we don't have the return statemnet, the value of the result
         # will not be returned after the function is done computations

In [13]: bing(2, 3)

Out[13]: 10

In [14]: # Now we can save the returned value in a variable

         x = bing(4,5)
         print(x)

18
```

1.0.5 Exercises

Rewrite your pay computation with time-and-a-half for overtime and create a function called "computepay" which takes two parameters (hours and rate).

```
In [15]: import sys

         try:
             hours = float(input("Enter Hours: "))
             rate = float(input("Enter Rate: "))
         except:
             print("Please enter valid input...")
             sys.exit(1)


         def computepay(hours, rate):
             if hours > 40:
                 overtime_hours = hours - 40  # hours over 40
                 hours -= overtime_hours  # regular rate hours
                 overtime_pay = overtime_hours * rate * 1.5
                 return (hours * rate) + overtime_pay
             else:
                 return (hours * rate)

         pay = computepay(hours, rate)

         print("Pay: %.2f" % pay)

         # OR
         print("Pay: " + str(pay))
```

```
        # OR
        print("Pay: {}".format(pay))
```

```
Enter Hours: 3
Enter Rate: 15
Pay: 45.00
Pay: 45.0
Pay: 45.0
```