Functions in Python

May 4, 2019

```
In [8]: # Function in Python
        def printDetails(): # Definition of  Function
            print("\nSANJEEV SHARMA")
            print("\nComputer Science(New)\nCLASS XII\nCode No. 083\n2019-20")
            print("Unit 1: Programming and Computational Thinking (PCT-2):\tFunctions")
        printDetails() # Call to Function
```

```
SANJEEV SHARMA

Computer Science(New)
CLASS XII
Code No. 083
2019-20
Unit 1: Programming and Computational Thinking (PCT-2):        Functions
```

```
In [1]: # Use of Function
        def sumOfDigits(): # Definition of  Function
            s=0
            n = int(input("\nEnter a Number\t"))
            while(n > 0) :
                    m= n%10
                    s=s+m
                    n= n//10
            print("\n The Sum of Digits of the above number is   ", s )

        sumOfDigits() # Call to Function
```

```
Enter a Number        123

 The Sum of Digits of the above number is    6
```

```
In [2]: # Use of Arguments in Function
        def texample(x ,y):
            print("\nInside texample Function")
```

```
        print("\n x = ",x, "\t y = ",y)

    x= int(input("\n Input the value of x"))
    y= int(input("\n Input the value of y"))
    print("\nOutside texample Function")
    print("\n x = ",x, "\t y = ",y)
    texample(x,y)
```

```
 Input the value of x10

 Input the value of y20

Outside texample Function

 x =  10          y =  20

Inside texample Function

 x =  10          y =  20
```

```
In [3]: # Use of Function Argument

    def reverseOfNumber(num): # Definition of  Function
        reverse=0
        while(num > 0) :
                m= num%10
                reverse=reverse * 10 + m
                num= num//10
        print("\n The Reverse of the Number is " , reverse)

    number= int(input("\nEnter a Number\t"))
    reverseOfNumber(number) # Call to Function with argument
```

```
Enter a Number        123

 The Reverse of the Number is  321
```

```
In [4]: # Use of Return statement in Function
    def texample(x ,y):
         return x+y
    x= int(input("\n Input the value of x"))
    y= int(input("\n Input the value of y"))
    print("\nOutside texample Function")
    print("\n x = ",x, "\t y = ",y)
```

```
        sum =texample(x,y)
        print("\n Sum is ",sum)
```

```
 Input the value of x10

 Input the value of y20

Outside texample Function

 x =  10          y =  20

 Sum is  30
```

```
In [7]: # Use of Function Argument and Return Statement
        def reverseOfNumber(num): # Definition of  Function
            r=0
            while(num > 0) :
                    m= num%10
                    r=r * 10 + m
                    num= num//10
            return r
        number= int(input("\nEnter a Number\t"))
        reverse = reverseOfNumber(number) # Call to Function with argument
        print("\n The Reverse of the Number is " , reverse) # Call to Function with argument
```

```
Enter a Number        123

 The Reverse of the Number is  321
```

```
In [8]: # Use of Default Argument
        def reverseOfNumber(num=789): # Definition of  Function and Default Argument
            r=0

            while(num > 0) :
                    m= num%10
                    r=r * 10 + m
                    num= num//10
            return r
        number= int(input("\nEnter a Number\t"))

        reverse = reverseOfNumber() # Call to Function with No argument
        print("\n The function will take its Default Argument 789 and calculate the reverse")
        reverse = reverseOfNumber() # Call to Function with No argument
        print("\n The Reverse of the Number is " , reverse)
```

```
        print("\n The function will take its Argument as entered by the user and calculate the
        reverse = reverseOfNumber(number) # Call to Function with With argument
        print("\n The Reverse of the Number is " , reverse)
```

```
Enter a Number        123

 The function will take its Default Argument 789 and calculate the reverse

 The Reverse of the Number is  987

 The function will take its Argument as entered by the user and calculate the reverse

 The Reverse of the Number is  321
```

```
In [9]: # Use of One Default and One actual Argument
        def Dfunction(x,y=20): # Default argument must follows non-default argument

           print("\n x = " , x , "y = ",y)

        print("\ncalled with Dfunction(10) ")
        Dfunction(10) # Called with Single Argument i.e x
        print("\ncalled with Dfunction(10,200) ")
        Dfunction(10,200)
```

```
called with Dfunction(10)

 x =  10 y =  20

called with Dfunction(10,200)

 x =  10 y =  200
```

```
In [10]: # Use of List Arguments in Function
         def texample(two):
             two[0]=100   # Updating the Value of List
             print("\nInside texample Function\n")
             for i in two:
              print(i ,"\t")

         list=[1,2,3,4]
         print("\nOrginal List \n",list)
         print("\nCalled with texample(list)\n")
         texample(list) # Passing list as argument
         print("\nUpdated List \n",list)
```

```
Orginal List
 [1, 2, 3, 4]

Called with texample(list)


Inside texample Function

100
2
3
4

Updated List
 [100, 2, 3, 4]
```

```
In [11]: # Use of List Arguments in Function
         def calculateSum(*args):
             sumOfNumbers = 0
             for elem in args:
                 sumOfNumbers += elem
             return sumOfNumbers

         list1 = [1,2,3,4,5,6,7,8]
         sum = calculateSum( *list1)
         print("\nSum of List Elements is = " , sum)
```

```
Sum of List Elements is =  36
```

```
In [12]: # Use of and tuple Arguments in Function
         def texample(two):
             two=(5,6,7,8)   # Updating the Tuple
             print("\nInside texample Function\n")
             for i in two:
              print(i ,"\t")
         tuple=(1,2,3,4)
         print("\nOrginal Tuple \n",tuple)
         print("\nCalled with texample(tuple)\n")
         texample(tuple) # Passing tuple as a argument
         print("\nNo Updation Reflected \n",tuple)
```

```
Orginal Tuple
 (1, 2, 3, 4)
```

```
Called with texample(tuple)


Inside texample Function

5
6
7
8

No Updation Reflected
 (1, 2, 3, 4)
```

```
In [13]: # Use of Tuple Arguments in Function
         def calculateSum(*args):
             sumOfNumbers = 0
             for elem in args:
                 sumOfNumbers += elem
             return sumOfNumbers

         tuple = (1,2,3,4,5)
         sum = calculateSum( *tuple)
         print("\nSum of List Elements is = " , sum)
```

```
Sum of List Elements is =  15
```

```
In [14]: # Use of Tuple Argument and List Arguments as a Tuple in Function
         def texample(*two):
             for i in two:
              print(i ,"\t")
         t=(1,2,3) # Tuple
         list=[1,2,3,4]
         print("\nCalled with texample(t)\n")
         texample(t) # Passing tuple as a single argument
         print("\nCalled with texample(*t)\n")
         texample(*t) # Passing tuple as a variable argument
         print("\nCalled with texample([1,2,3,4])\n")
         texample(list) # Passing list as single argument
         print("\nCalled with texample(*t,[1,2,3,4])\n")
         texample(*t,list) # Passing tuple as variable argument and list as a single argument
         print("\nCalled with texample(t,[1,2,3,4])\n")
         texample(t,list) # Passing tuple and list as a single argument
```

```
Called with texample(t)
```

```
(1, 2, 3)

Called with texample(*t)

1
2
3

Called with texample([1,2,3,4])

[1, 2, 3, 4]

Called with texample(*t,[1,2,3,4])

1
2
3
[1, 2, 3, 4]

Called with texample(t,[1,2,3,4])

(1, 2, 3)
[1, 2, 3, 4]
```

```
In [15]: # Use of List Argument as a Tuple in Function
         def texample(*two):
             print("\nInside texample Function\n")
             for i in two:
              print(i ,"\t")

         list=[1,2,3,4]
         print("\nCalled with texample(list) - As a Single Value\n")
         texample(list) # Passing list as a single argument
         print("\nCalled with texample(*tlist) - As a Tuple of Values \n")
         texample(*list) # Passing list  as a tuple argument
         ##print("\nCalled with texample([1,2,3,4])\n")
```

```
Called with texample(list) - As a Single Value


Inside texample Function

[1, 2, 3, 4]

Called with texample(*tlist) - As a Tuple of Values
```

```
Inside texample Function

1
2
3
4
```

```
In [16]: # Use of Dictionary argument in Function

         def texample(**two):
             print("Inside texample function\n",two)

         two={'Name' : 'Radha' , 'Age':30}
         print("Initial Dictionary is ",two)
         texample(**two)
         print("Outside texample function \n",two)
```

```
Initial Dictionary is  {'Name': 'Radha', 'Age': 30}
Inside texample function
 {'Name': 'Radha', 'Age': 30}
Outside texample function
 {'Name': 'Radha', 'Age': 30}
```

```
In [17]: # Use of Dictionary argument as a Tuple in Function

         def texample(**two):
             two['Name'] ="Karan"  # Change of Name key value
             print("Inside texample function\n",two)

         two={'Name' : 'Radha' , 'Age':30}
         print("Initial Dictionary is ",two)
         texample(**two)
         print("Outside texample function - No Change reflected \n",two)
```

```
Initial Dictionary is  {'Name': 'Radha', 'Age': 30}
Inside texample function
 {'Name': 'Karan', 'Age': 30}
Outside texample function - No Change reflected
 {'Name': 'Radha', 'Age': 30}
```

```
In [18]: # Use of Dictionary argument in Function

         def texample(two):
             two['Name'] ="Karan"  # Change of Name key value
             print("Inside texample function\n",two)
```

```
two={'Name' : 'Radha' , 'Age':30}
print("Initial Dictionary is ",two)
texample(two)
print("Outside texample function - Change reflected \n",two)
```

```
Initial Dictionary is  {'Name': 'Radha', 'Age': 30}
Inside texample function
 {'Name': 'Karan', 'Age': 30}
Outside texample function - Change reflected
 {'Name': 'Karan', 'Age': 30}
```

In [ ]: