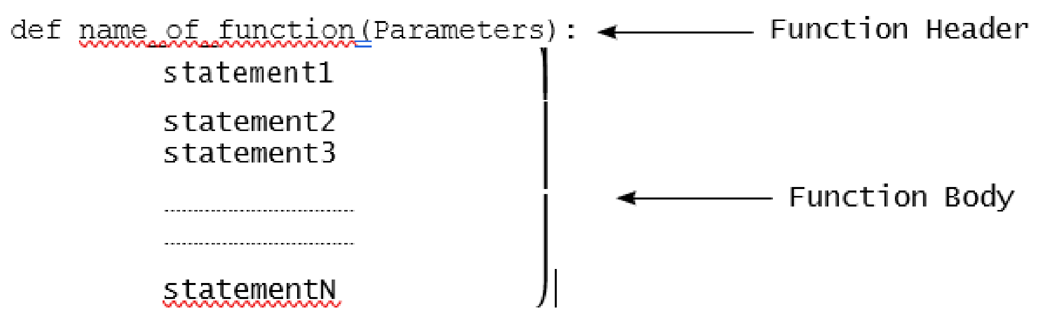
# Python Assignment No.3

## Q1.}Define Function. Describe syntax and basics of function.

**Ans.** It is difficult to prepare and maintain a large-scale program and the identification of the flow of data subsequently gets harder to understand. The best way to create a programming application isto divide a big program into small modules and repeatedly call these modules.

With the help of functions, an entire program can be divided into small independent modules (each small module is called a function). This improves the code's readability as well as the flow of execution as small modules can be managed easily.

A function is a self-contained block of one or more statements that performs a special task whencalled. The syntax for function is given as follows:



```
def name_of_function(Parameters):
        statement1

        statement2
        statement3

        ..........................

        ..........................

        statementN
```

The syntax for the Python function contains a header and body. The function header begins with the 'def' keyword. The def keyword signifies the beginning of the function's definition. The name of the function is followed by the def keyword. The function header may contain zero or more number of parameters. These parameters are called formal parameters. If a function containsmore than one parameter then all the parameters are separated by commas. A function's body is ablock of statements. The statements within the function's body define the actions that the functionneeds to perform.

```
def Display():
    print("Welcome  to Python Programming ")
Display()             #call function
```

Output

```
Welcome to Python Programming
```

```
def print_msg():
    str1=input("Please Enter Your Name:")
    print("Dear ",str1," Welcome  to Python Programming ")
print_msg()  #call function
```

Output

```
Please Enter Your Name: Virat
Dear Virat  Welcome  to Python Programming
```

## Q.2} What is the role of parameters and arguments in a function.

**Ans:** Parameters are used to give inputs to a function. They are specified with a pair of parenthesis in the function's definition. When a programmer calls a function, the values are also passed to the function.

While parameters are defined by names that appear in the function's definition, argumentsare values actually passed to a function when calling it. Thus, parameters define what types of arguments a function can accept.

```
def printMax(num1,num2):
        Statemen1                       }
        Statemen2                       }    #Define a Function
        .............................   }
        .............................   }
        StatementN                      }

printMax(10,20)     <--------   Call a function(Invoke)
```

In the above example, printMax(num1, num2) has two parameters, viz. num1 and num2.The parameters num1 and num2 are also called formal parameters. A function is invoked by calling the name of the function, i.e. printMax(10,20), where 10, 20 are the actual parameters. Actual parameters are also called arguments. num1 and num2 are the parameters of a function. Program 6.5 demonstrates the use of parameters and arguments in a function.

## Q.3} Explain Local & Global scope of variable with example.

**Ans.**

Variables and parameters that are initialized within a function including parameters, are said to exist in that function's local scope. Variables that exist in local scope are called local variables. Variables that are assigned outside functions are said to exist in global scope. Therefore, variables that exist in global scope are called global variables.

```
p = 20          #global variable p
def Demo():
    q = 10      #Local variable q
    print('The value of Local variable q:',q)
    #Access global variable p within this function
    print('The value of Global Variable p:',p)
Demo()
#Access global variable p outside the function Demo()
print('The value of global variable  p:',p)
```

**Output**

```
The value of Local variable q: 10
The value of Global Variable p: 20
The value of global variable p: 20
```

In the above example, we have created one local variable ‘q’ and one global variable‘p’. As global variables are created outside all functions and are accessible to all functions in their scope, in the above example as well the global variable ‘p’ is accessed from the function `Demo()` and it is also accessed outside the function.

## Q.4} Explain Recursive function in details.

**Ans.** We have seen that it is legal for one function to call another function. In programming, there might be a situation where a function needs to invoke itself. Python also supports the recursive feature, which means that a function is repetitively called by itself. Thus, a function is said to be recursive if a statement within the body of the function calls itself.

Let us consider a simple example of recursion. Suppose we want to calculate the factorial valueof an integer. We know that the factorial of a number is the product of all the integers between 1 and that number, i.e. n! is defined as n * (n-1)!.

Consider the following example.

Formula to calculate the factorial of a number (n)! = n*(n-1)!

$$\begin{aligned} 5! &= 5*(4)! \\ &= 5*4*(3)! \\ &= 5*4*3*(2)! \\ &= 5*4*3*2*(1) \\ &= 120 \end{aligned}$$

```
def factorial(n):
    if n==0:
        return 1
    return n*factorial(n-1)
print(factorial(5))
```

Output

```
120
```

In the above program, `factorial()` is a recursive function. The number is passedto function `factorial()`. When the function factorial is executed, it is repeatedly invoked by itself. Every time a function is invoked, the value of ‘n’ is reduced by one and multiplication is carried out.The recursion function produces the number 5, 4, 3, 2 and 1. The multiplication of these numbers iscarried out and returned. Finally, the print statement prints the factorial of the number.

## Q.5} Define strings and describe string class.

**Ans.** Characters are building blocks of Python. A program is composed of a sequence of characters. When a sequence of characters is grouped together, a meaningful string is created. Thus, a string is a sequence of characters treated as a single unit.

In many languages, strings are treated as arrays of characters but in Python a string is an objectof the str class. This string class has many constructors.

**The str class:**

Strings are objects of the str class. We can create a string using the constructor of str class as:

```
S1=str() #Creates an Empty string Object
S2=str(“Hello”) #Creates  a  String  Object  for  Hello
```

An alternative way to create a string object is by assigning a string value to a variable.

```
S1 = “”  # Creates a Empty String
S2= “Hello” # Equivalent to S2=str(“Hello”)
```

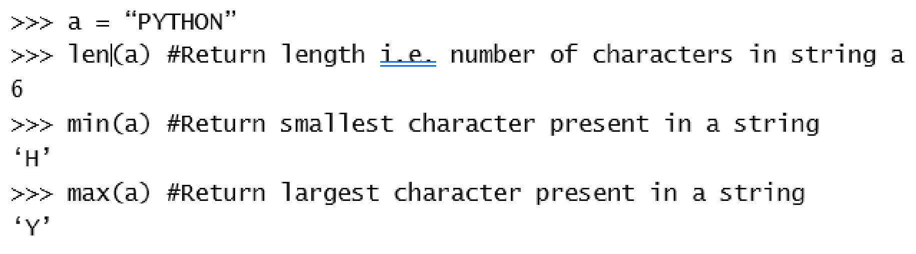
All the characters of a string can be accessed at one time using the index operator.

## Q.6} Describe basic inbuilt Python function for string.

**Ans.**

Python has several basic inbuilt functions that can be used with strings. A programmer can make use of `min()` and `max()` functions to return the largest and smallest character in a string. We canalso use `len()` function to return the number of characters in a string.

The following example illustrates the use of the basic function on strings.



```
>>> a = “PYTHON”
>>> len(a) #Return length i.e. number of characters in string a
6
>>> min(a) #Return smallest character present in a string
‘H’
>>> max(a) #Return largest character present in a string
‘Y’
```

## Q.7} Describe Traversing string with for loop.

**Ans.** A programmer can use the for loop to traverse all characters in a string. For example, the followingcode displays all the characters of a string.

```
S="India"
for ch in S:
    print(ch,end="")
```

Output

```
India
```

The string 'India' is assigned to the variable S. The for loop is used to print all the characters of a string S. The statement 'for ch in S:' can read as 'for each character ch in S print ch'.

```
S="ILOVEPYTHONPROGRAMMING"
for ch  in range(0,len(S),2):#Traverse each Second character
    print(S[ch],end=" ")
```

**Output**

```
I O E Y H N R G A M N
```

## Q.8} Explain string operations, Explain in details for following:-

### a)String comparison.

**Ans.**

Operators such as ==,<,>,<=,>=and != are used to compare the strings. Python compares strings by comparing their corresponding characters.

*Example*

```
>>> S1="abcd"
>>> S2="ABCD"
>>> S1>S2
True
```

The string 'abcd' is assigned to the string S1 and the String 'ABCD' is assigned to S2. The statementS1 > S2 returns True because Python compares the numeric value of each character. In the above example, the numeric value, i.e. ASCII value of 'a' is 97 and ASCII numeric value of 'A' is 65. It means 97 > 65. Thus, it returns True. However, character by character comparison goes on till the end of the string.

*Some More Examples of String Comparison*

```
>>> S1="abc"
>>> S2="abc"
>>> S1==S2
True
>>> S1="ABC"
>>> S2="DEF"
>>> S1>S2|
False

>>> S1="AAA"
>>> S2="AAB"
>>> S2>S1
True

>>> S1="ABCD"
>>> S2="abcd".upper()
>>> S2
'ABCD'
>>> S1>S2
False
>>> S1>=S2
True
```

## b) String.format() method.

**Ans.**

In Python 2 and 3, programmers can include %s inside a string and follow it with a list of values for each %.

```
>>> "My Name is %s and I am from %s"%("JHON","USA")
'My Name is JHON and I am from USA'
```

In the above example, we have seen how to format a string using % (modulus) operator. However,for more complex formatting, Python 3 has added a new string method called `format()` method. Instead of % we can use {0}, {1} and so on. The syntax for `format()` method is:

```
        template.foramt(P0,P1,……… ,k0=V0,K1=V1…}
```

whereas the arguments to the .format() method are of two types. It consists of zero or more positional arguments $P_i$ followed by zero or more keyword arguments of the form, $K_i=V_i$.

*Example*

```
>>> '{} plus {} equals {}'.format(4,5,'Nine')
'4 plus 5 equals Nine'
```

The `format()` method is called on the string literal with arguments 4,5 and 'nine'. The empty {} are replaced with the arguments in order. The first {} curly bracket is replaced with the first argument and so on. By default, the index of the first argument in format always start from zero. One can also give a position of arguments inside the curly brackets.

The following example illustrates the use ofindex as argument inside the curly bracket.

```
>>>"My Name is {0} and I am from {1}".format("Milinda","USA")
'My Name is Milinda and I am from USA'
```

The `format()` method contains various arguments. In the above example, the `format()` methodhas two arguments, viz. "Milinda" and "USA". The index of the first argument of the `format()` method always starts from 0. Therefore, {0} replaces the 0th argument of the format. Similarly {1} replaces the first argument of the format.

**c) split() method.**

**Ans.**

The `split()` method returns a list of all the words in a string. It is used to break up a string into smaller strings.

Example

Consider the following example where names of different programming languages such as C, C++,Java and Python is assigned to a variable Str1. Applying split() method on str1 returns the list of programming languages.

```
>>>Str1="C C++ JAVA Python"#Assigns names of Programming languages to Str1
>>>Str1.split()
['C,C++,JAVA,Python']
```

```
TOP_10_Company="TCS,INFOSYS,GOOGLE,MICROSOFT,YAHOO"
Company=TOP_10_Company.split(",")
print(Company)
for c  in Company:
    print(end="")
    print(c)
```

Output

```
['TCS', 'INFOSYS', 'GOOGLE', 'MICROSOFT', 'YAHOO']
TCS
INFOSYS
GOOGLE
MICROSOFT
YAHOO
```

## Q.9} Explain searching substring present in a string.

**Ans.**

| *Methods of str Class for Searching the Substring in a Given String* | *Meaning* |
|---|---|
| `bool endswith(str Str1)`<br>Example:<br>`>>> S="Python Programming"`<br>`>>>S.endswith("Programming")`<br>`True` | Returns true if the string ends with the substring Str1. |
| `bool startswith(str Str1)`<br>Example:<br>`>>> S="Python Programming"`<br>`>>>S.startswith("Python")`<br>`True` | Returns true if the string starts with the substring Str1. |
| `int find(str Str1)`<br>Example:<br>`>>> Str1="Python Programming"`<br>`>>> Str1.find("Prog")`<br>`7#Returns the index from where the string "Prog" begins`<br>`>>> Str1.find("Java")`<br>`-1#Returns -1 if the string "Java" is not found in the string str1` | Returns the lowest index where the string Str1 starts in this string or returns -1 if the string Str1 is not found in this string. |
| `int rfind(str Str1)`<br>Example:<br>`>>> Str1="Python Programming"`<br>`>>> Str1.rfind("o")`<br>`9#Returns the index of last occurrence of string "o" in Str1` | Returns the highest index where the string Str1 starts in this string or returns -1 if the string Str1 is not found in this string. |
| `int count(str S1)`<br>Example:<br>`>>> Str1="Good Morning"`<br>`>>> Str1.count("o")`<br>`3` | Returns the number of occurrences of this substring. |

## Q.10} Explain methods to convert string into another string.

**Ans.**

A string may be present in lower case or upper case. The string in lower case can be converted into upper case and vice versa using various methods of the str class.

| *Methods of Str Class to Convert a String from One Form to Another* | *Meaning* |
|---|---|
| `str capitalize()`<br>Example:<br>`>>> Str1=”hello”`<br>`>>> Str1.capitalize ()`<br>`‘Hello’ #Convert first alphabet of String Str1 to uppercase` | Returns a copy of the string with only the first character capitalised. |
| `str lower()`<br>Example:<br>`>>> Str1=”INDIA”`<br>`>>> Str1.lower ()`<br>`‘india’` | Returns a copy of the string with all the letters converted into lower case. |
| `str upper()`<br>Example:<br>`>>> Str1=”iitbombay”`<br>`>>> Str1.upper()`<br>`‘IITBOMBAY’` | Returns a copy of the string with all the letters converted into upper case. |
| `str title()`<br>Example:<br>`>>> Str1=”welcome to the world of programming”`<br>`>>> Str1.title()`<br>`‘Welcome To The World Of Programming’` | Returns a copy of the string with the first letter capitalised in each word of the string. |
| `str swapcase()`<br>Example:<br>`>>> Str1=”IncreDible India”`<br>`>>> Str1.swapcase ()`<br>`‘iNCREdIBLE iNDIA’` | Returns a copy of the string which converts upper case characters into lower case characters and lower case characters into upper case characters. |
| `str replace (str old, str new [,count])`<br>Example:<br>`>>> S1=”I have brought two chocolates, two cookies and two cakes”`<br>`#Replace the old string i.e “two” by new string i.e. “three”.`<br>`>>> S2=S1.replace(“two”,”three”)`<br>`#Replace all occurrences of old string “two” by “three”`<br>`>>> S2`<br>`‘I have brought three chocolates, three cookies and three cakes’` | Returns a new string that replaces all the occurrences of the old string with a new string. The third parameter, i.e. the count is optional. It tells the number ofold occurrences of the string to be replaced with new occurrences of the string. |