Functions in Python

Tushar B. Kute,
http://tusharkute.com

Function

- A function is a block of code which only runs when it is called.
- You can pass data, known as parameters, into a function.
- A function can return data as a result.
- In Python a function is defined using the def keyword:

Function Types

- Basically, we can divide functions into the following two types:
  - Built-in functions - Functions that are built into Python.
  - User-defined functions - Functions defined by the users themselves.

A Sample Function

```
# Function Definition
def show():
    print "Hello World"

# Function Call
show()
```

Parameterized Functions

- Information can be passed to functions as parameter.
- Parameters are specified after the function name, inside the parentheses.
- You can add as many parameters as you want, just separate them with a comma.

Example

```
def square(x):
    y = x * x
    print "Square: ", y

num = input("Enter number: ")
square(num)

square(34)
```

Function Returning values

```
num = input("Enter number: ")

def square(x):
    y = x * x
    return y

def cube(x):
    y = x * square(x)
    return y

print "Square is", square(num)
a = cube(num)
print "Cube is", a
print "Cube is", num * square(num)
```

Default Parameters

```
def mul(x = 2, y = 5, z = 10):
    result = x * y * z
    return result

print "Multi is", mul(12, 3, 6) #216
print "Multi is", mul(12, 3) #360
print "Multi is", mul(12) #600
print "Multi is", mul() #100
print "Multi is", mul(z=2,y=3,x=7) #42
```

Multi – return statement

- The Python function can return multiple values at a time with multiple independent variables.

Example

```
# Functions returning multiple values
def array(n):
    add = 0
    for x in n:
        add += x
    avg = add / len(n)
    return add, avg      # Multi-return

arr = [43,65,76,11.0,23,67,82]
a, b = array(arr)        # Function Call
print "Addition is", a
print "Average is %.2f" %b
```

Recursion

- Python functions have ability to call by themselves. This is termed as Recursion.

```
def factorial(n):
    if n <= 1:
        return 1
    else:
        return n * factorial(n-1)

num = input("Enter the number: ")
print "Factorial is", factorial(num)
```

Variable length arguments

```
# Simple function to loop args
def show(*args):
    for a in args:
        print a

# Call the function
show(1,2,3)
```

Variable length arguments

```
# Simple function to loop
def display(**kwargs):
    for a in kwargs:
        print a, kwargs[a]

# Call the function
display(name='Rashmi', age=30)
```

Scope of variables

```
# Global variable
x = 10
# Simple function to add two numbers
def add(y):
    return x + y

# Call the function and print result
print add(20)
```

Anonymous Function

- In Python, anonymous function is a function that is defined without a name.
- While normal functions are defined using the def keyword, in Python anonymous functions are defined using the lambda keyword.
- Hence, anonymous functions are also called lambda functions.

Syntax:

- A lambda function in python has the following syntax.

```
lambda arguments: expression
```

- Lambda functions can have any number of arguments but only one expression.
- The expression is evaluated and returned. Lambda functions can be used wherever function objects are required.

Example:

```
# Program to show the use of lambda functions

square = lambda x: x ** 2

# Output: 144

print square(12)
```

Using lambda function

- We use lambda functions when we require a nameless function for a short period of time.
- In Python, we generally use it as an argument to a higher-order function (a function that takes in other functions as arguments).
- Lambda functions are used along with built-in functions like filter(), map() etc.

The filter( )

- The filter() function in Python takes in a function and a list as arguments.
- The function is called with all the items in the list and a new list is returned which contains items for which the function evaluates to True.

Example:

```
# a list contains both even and odd numbers.
seq = [0, 1, 2, 3, 5, 8, 13]

# result contains odd numbers of the list
result = filter(lambda x: x % 2, seq)
print result

# result contains even numbers of the list
result = filter(lambda x: x % 2 == 0, seq)
print result
```

The map( )

- The map() function in Python takes in a function and a list.
- The function is called with all the items in the list and a new list is returned which contains items returned by that function for each item.

Example:

```
def square(n):
    return n * n

# We square all numbers using map()
numbers = (1, 2, 3, 4)
result = map(square, numbers)
print(result)

# List of strings
l = ['sat', 'bat', 'cat', 'mat']

# map() can listify the list of strings individually
test = map(list, l)
print(test)
```

The global keyword

- In Python, global keyword allows you to modify the variable outside of the current scope. It is used to create a global variable and make changes to the variable in a local context.
- Rules of global Keyword
  - When we create a variable inside a function, it's local by default.
  - When we define a variable outside of a function, it's global by default. You don't have to use global keyword.
  - We use global keyword to read and write a global variable inside a function.
  - Use of global keyword outside a function has no effect

Example:

```
c = 0 # global variable

def add():
    global c
    c = c + 2 # increment by 2
    print "Inside add():", c

add()
print "In main:", c
```

Thank you

This presentation is created using LibreOffice Impress 5.1.6.2, can be used freely as per GNU General Public License

@mitu_skillologies

/mITuSkillologies

@mitu_group

/company/mitu-skillologies

Web Resources
http://mitu.co.in
http://tusharkute.com

contact@mitu.co.in

tushar@tusharkute.com