Type of Functions

1. Built in Functions
2. User Define Functions

Built in Functions

Functions that are built into Python.

1. abs()

```
# find the absolute value

num = -100.58
print(abs(num))
```

```
100.58
```

2. all()

return values of all function

True: if all elements in an iterable are true

False: if any element in an iterable is false

```
lst = [ 1,2,3,4,-15,-10]
print(all(lst))
```

```
True
```

```
lst = [ 0,1,2,3]     # 0 present in list
print(all(lst))
```

```
False
```

```
lst = []              # empty list always true   #  "\0"
print(all(lst))
str1 = ""
print(all(str1))
```

```
True
True
```

```
lst = [False,1,2]     # False present in a list so all(lst) is false
print(all(lst))
```

```
False
```

3. dir()

The dir() tries to return a list of valid attributes of the object.

If the object has dir() method, the method will be called and must return the list os attributes.

If the object dosen't have dir() method, this method tries to find information form the dirt attibute(if defined), and from type object.In this case, the list return from dir() may not be complete.

```
number = [ 1,2,3,6,7]
print(dir(number))
print(len(dir(number)))
```

```
['__add__', '__class__', '__contains__', '__delattr__', '__delitem__',
'__dir__', '__doc__', '__eq__', '__format__', '__ge__',
'__getattribute__', '__getitem__', '__gt__', '__hash__', '__iadd__',
'__imul__', '__init__', '__init_subclass__', '__iter__', '__le__',
'__len__', '__lt__', '__mul__', '__ne__', '__new__', '__reduce__',
'__reduce_ex__', '__repr__', '__reversed__', '__rmul__',
'__setattr__', '__setitem__', '__sizeof__', '__str__',
'__subclasshook__', 'append', 'clear', 'copy', 'count', 'extend',
'index', 'insert', 'pop', 'remove', 'reverse', 'sort']
46
```

4. divmod()

The divmod() method takes two numbers and return a pair of number(a tuple) consisting of their quotient and remainder.

```
print(divmod(28,6))
```

```
(4, 4)
```

5. enumerate()

The enumerate() method adds counter to an iterable and return it.

syntax: enumerate(iterable, start= 0)

```
number = [10,20,30,40,50,60,70]

for index, num in enumerate(number, start = -10):
    print("index {0} has value {1} ".format(index,num))
```

```
index -10 has value 10
index -9 has value 20
```

```
index -8 has value 30
index -7 has value 40
index -6 has value 50
index -5 has value 60
index -4 has value 70
```

6. filter()

The filter() method constructs an interator from elements of an iterable for which function returns true.

syntex: filter(function, iterable)

```
def find_positive_number(num):
    """
    The function return positive number if num is positive
    """
    if num > 0:
        return num

number_list =  range(-70,10) # create  a list with number  from -30 to
40
print(list(number_list))
print("")
positive_number_list =  list(filter(find_positive_number,
number_list))

print(positive_number_list)
```

```
[-70, -69, -68, -67, -66, -65, -64, -63, -62, -61, -60, -59, -58, -57,
-56, -55, -54, -53, -52, -51, -50, -49, -48, -47, -46, -45, -44, -43,
-42, -41, -40, -39, -38, -37, -36, -35, -34, -33, -32, -31, -30, -29,
-28, -27, -26, -25, -24, -23, -22, -21, -20, -19, -18, -17, -16, -15,
-14, -13, -12, -11, -10, -9, -8, -7, -6, -5, -4, -3, -2, -1, 0, 1, 2,
3, 4, 5, 6, 7, 8, 9]

[1, 2, 3, 4, 5, 6, 7, 8, 9]
```

7. isinstance()

The isinstance() function checks if the objects(first argument) is an instance or subclass of classinfo class(second argument).

syntex: isinstance(object, classinfo)

```
lst = [1,2,3,4]
print(isinstance(lst,list))

# try with other datatype tuple,set
```

```
t = (1)
print(isinstance(t,int))
```

```
True
True
```

8. map()

Map applies a function to all the items in an input_lst.

syntax: map(function_to_apply, list_of_inputs)

```
number = [1,2,3,4,5,7,8,9,10,11,12]

# normal method of computing num^2 for each element in the list
squared = []
for num in number:
    squared.append(num**2)
```

```
number = [1,2,4,5,6,7,8,10,11,12,13,14]

def powerOfTwo(num):
    return num**2
# using map function
squared =  list(map(powerOfTwo, number))
%timeit print(squared)
```

```
[1, 4, 16, 25, 36, 49, 64, 100, 121, 144, 169, 196]
```

9. reduce()

The reduce() function is for performing some computation on a list and returning a result.

It applies a rolling computation to sequential pairs of value in list.

```
# product of element in the list
product = 1
lst = [1,2,3,4,5,6]

# traditinal program without reduce()
for num in lst:
    product *= num
print(product)
#%timeit print(product)
```

```
720
```

```
# with reduce()
from functools import reduce # python 3
def multiply(x,y):
```

```
    return x*y;

lst = [1,2,3,4,5,6]
product = reduce(multiply,lst)
print(product)
#%timeit print(product)
```

```
720
```

Note:

filter(), map(), reduce() are most used or importent build_in function for data_science purpose. Because these function simply avoid the use of for loop specialy in data_science.

User Define Function:

Function that we define ourselves to do certain specific task are referred as user-defined functions.

If we use functions written by other in the form of library, it can be termed as libaray function.

Advantages:

1. User define functions help to decompose a large program into small segments which makes program to easy to understand, maintain and debug.
2. If the repeated code occurs in a program. Function can be used to include those code and execute when needed by calling that function.
3. Programmars working on large project can divide the workload by making different functions.

Example:

```
def product_nums(a,b):
    """
    This function return the product of two numbers
    """
    product = a*b
    return product

num1 = 200
num2 = 13
print("product of {0} and {1} is
{2}".format(num1,num2,product_nums(num1,num2)))
```

```
product of 200 and 13 is 2600
```

Python Program to make a simple calculator that can add, subtract, multiplya and division.

```
def add(a,b):
    """
    This function add two numbers
    """
    return a + b

def sub(a,b):
    """
    This function subtract two numbers
    """
    return a - b

def mult(a,b):
    """
    This function multiply two numbers
    """
    return a * b
def div(a,b):
    """
    This function divide two numbers
    """
    return a / b

print("select option")
print("1. Addition")
print("2. Subtraction")
print("3. Multiplication")
print("4. division")

# take input form user
choice = int(input("Enter the choice 1/2/3/4"))

num1 = float(input("enter the value of num1 : "))
num2 = float(input("enter the value of num2 : "))

if choice == 1:
    print("Addition of {0} and {1} is
{2}".format(num1,num2,add(num1,num2)))
elif choice == 2:
    print("Subtraction of {0} and {1} is
{2}".format(num1,num2,sub(num1,num2)))
elif choice == 3:
    print("multiplication of {0} and {1} is
{2}".format(num1,num2,mult(num1,num2)))
elif choice == 4:
    print("Division of {0} and {1} is
{2}".format(num1,num2,div(num1,num2)))
```

```
else:
    print("Invalid choice")
```

```
select option
1. Addition
2. Subtraction
3. Multiplication
4. division
Enter the choice 1/2/3/43
enter the value of num1 : 32
enter the value of num2 : 2
multiplication of 32.0 and 2.0 is 64.0
```

```
a = 50.00
b = 6
print(add(a,b))
```

```
11
```

```
5
```