PROGRAMMING WITH PYTHON

Karuna sheel

NIELIT kurukshetra

PYTHON BUILT-IN FUNCTIONS

The Python built-in functions are defined as the functions whose functionality is pre-defined in Python. The python interpreter has several functions that are always present for use. These functions are known as Built-in Functions. There are several built-in functions in Python which are listed below:

Python abs() Function

The python abs() function is used to return the absolute value of a number. It takes only one argument, a number whose absolute value is to be returned. The argument can be an integer and floating-point number. If the argument is a complex number, then, abs() returns its magnitude.

Python abs() Function Example

```
#  integer number
integer = -20
print('Absolute value of -40 is:', abs(integer))
#  floating number
floating = -20.83
print('Absolute value of -40.83 is:', abs(floating))
```

Output:

```
Absolute value of -20 is: 20
Absolute value of -20.83 is: 20.83
```

PYTHON ALL() FUNCTION

The python all() function accepts an iterable object (such as list, dictionary, etc.). It returns true if all items in passed iterable are true. Otherwise, it returns False. If the iterable object is empty, the all() function returns True.

Python all() Function Example

```
# all values true
k = [1, 3, 4, 6]
print(all(k))
# all values false
k = [0, False]
print(all(k))
# one false value
k = [1, 3, 7, 0]
print(all(k))
```

```
# one true value
k = [0, False, 5]
print(all(k))

# empty iterable
k = []
print(all(k))
```

Output:

```
True
False
False
False
True
```

PYTHON BIN() FUNCTION

The python bin() function is used to return the binary representation of a specified integer. A result always starts with the prefix 0b.

Python bin() Function Example

```
x =  10
y =  bin(x)
print (y)
```

Output:

```
0b1010
```

PYTHON BOOL()

The python bool() converts a value to boolean(True or False) using the standard truth testing procedure.

Python bool() Example

```
test1 = []
print(test1,'is',bool(test1))
test1 = [0]
print(test1,'is',bool(test1))
test1 = 0.0
print(test1,'is',bool(test1))
test1 = None
print(test1,'is',bool(test1))
test1 = True
print(test1,'is',bool(test1))
test1 = 'Easy string'
print(test1,'is',bool(test1))
```

Output:

```
[] is False
[0] is True
0.0 is False
None is False
True is True
Easy string is True
```

PYTHON CALLABLE() FUNCTION

A python callable() function in Python is something that can be called. This built-in function checks and returns true if the object passed appears to be callable, otherwise false.

Python callable() Function Example

```
x = 8
print(callable(x))
```

Output:

```
False
```

PYTHON COMPILE() FUNCTION

The python compile() function takes source code as input and returns a code object which can later be executed by exec() function.

Python compile() Function Example

```
# compile string source to code
code_str = 'x=5\ny=10\nprint("sum =",x+y)'
code = compile(code_str, 'sum.py', 'exec')
print(type(code))
exec(code)
exec(x)
```

Output:

```
<class 'code'>
sum = 15
```

PYTHON EXEC() FUNCTION

The python exec() function is used for the dynamic execution of Python program which can either be a string or object code and it accepts large blocks of code, unlike the eval() function which only accepts a single expression.

Python exec() Function Example

```
x = 8
exec('print(x==8)')
exec('print(x+4)’)
```

Output:

```
True
12
```

PYTHON SUM() FUNCTION

As the name says, python sum() function is used to get the sum of numbers of an iterable, i.e., list.

Python sum() Function Example

```
s = sum([1, 2,4 ])
print(s)

s = sum([1, 2, 4], 10)
print(s)
```

Output:

```
7
17
```

PYTHON ANY() FUNCTION

The python any() function returns true if any item in an iterable is true. Otherwise, it returns False.

Python any() Function Example

```
l = [4, 3, 2, 0]
print(any(l))

l = [0, False]
print(any(l))

l = [0, False, 5]
print(any(l))

l = []
print(any(l))
```

Output:

```
True
False
True
False
```

PYTHON EVAL() FUNCTION

The python eval() function parses the expression passed to it and runs python expression(code) within the program.

Python eval() Function Example

```
x = 8
print(eval('x + 1'))
```

Output:

```
9
```

PYTHON FLOAT()

The python float() function returns a floating-point number from a number or string.

Python float() Example

```
# for integers
print(float(9))

# for floats
print(float(8.19))

# for string floats
print(float("-24.27"))

# for string floats with whitespaces
print(float("     -17.19\n"))

# string float error
print(float("xyz"))
```

Output:

```
9.0
8.19
-24.27
-17.19
ValueError: could not convert string to float: 'xyz'
```

PYTHON FORMAT() FUNCTION

The python format() function returns a formatted representation of the given value.

Python format() Function Example

```
# d, f and b are a type

# integer
print(format(123, "d"))

# float arguments
print(format(123.4567898, "f"))

# binary format
print(format(12, "b"))
```

Output:

```
123
123.456790
1100
```

PYTHON FROZENSET()

The python frozenset() function returns an immutable frozenset object initialized with elements from the given iterable.

Python frozenset() Example

```
# tuple of letters
letters = ('m', 'r', 'o', 't', 's')

fSet = frozenset(letters)
print('Frozen set is:', fSet)
print('Empty frozen set is:', frozenset())
```

Output:

```
Frozen set is: frozenset({'o', 'm', 's', 'r', 't'})
Empty frozen set is: frozenset()
```

PYTHON DICT()

Python dict() function is a constructor which creates a dictionary. Python dictionary provides three different constructors to create a dictionary:

If no argument is passed, it creates an empty dictionary.
If a positional argument is given, a dictionary is created with the same key-value pairs. Otherwise, pass an iterable object.
If keyword arguments are given, the keyword arguments and their values are added to the dictionary created from the positional argument.
Python dict() Example

```
# Calling function
result = dict() # returns an empty dictionary
result2 = dict(a=1,b=2)
# Displaying result
print(result)
print(result2)
```

Output:

```
{}
{'a': 1, 'b': 2}
```

PYTHON FILTER() FUNCTION

Python filter() function is used to get filtered elements. This function takes two arguments, first is a function and the second is iterable. The filter function returns a sequence of those elements of iterable object for which function returns true value.

The first argument can be none, if the function is not available and returns only elements that are true.

Python filter() Function Example

```
# Python filter() function example
def filterdata(x):
    if x>5:
        return x
# Calling function
result = filter(filterdata,(1,2,6))
# Displaying result
print(list(result))
```

Output:

```
[6]
```

PYTHON HASH() FUNCTION

Python hash() function is used to get the hash value of an object. Python calculates the hash value by using the hash algorithm. The hash values are integers and used to compare dictionary keys during a dictionary lookup. We can hash only the types which are given below:

Hashable types: * bool * int * long * float * string * Unicode * tuple * code object.

Python hash() Function Example

```
# Calling function
result = hash(21) # integer value

result2 = hash(22.2) # decimal value

# Displaying result

print(result)

print(result2)
```

Output:

```
21
461168601842737174
```

PYTHON MIN() FUNCTION

Python min() function is used to get the smallest element from the collection. This function takes two arguments, first is a collection of elements and second is key, and returns the smallest element from the collection.

Python min() Function Example

```
# Calling function
small = min(2225,325,2025) # returns smallest element
small2 = min(1000.25,2025.35,5625.36,10052.50)
# Displaying result
print(small)
print(small2)
```

Output:

```
325
1000.25
```

PYTHON SET() FUNCTION

In python, a set is a built-in class, and this function is a constructor of this class. It is used to create a new set using elements passed during the call. It takes an iterable object as an argument and returns a new set object.

Python set() Function Example

```
# Calling function
result = set() # empty set
result2 = set('12')
result3 = set('javatpoint')
# Displaying result
print(result)
print(result2)
print(result3)
```

Output:

```
set()
{'1', '2'}
{'a', 'n', 'v', 't', 'j', 'p', 'i', 'o'}
```

PYTHON MAP() FUNCTION

The python map() function is used to return a list of results after applying a given function to each item of an iterable(list, tuple etc.).

Python map() Function Example

```
def calculateAddition(n):
  return n+n


numbers = (1, 2, 3, 4)
result = map(calculateAddition, numbers)
print(result)


# converting map object to set
numbersAddition = set(result)
print(numbersAddition)
```

Output:

```
<map object at 0x7fb04a6bec18>
{8, 2, 4, 6}
```

PYTHON OPEN() FUNCTION

The python open() function opens the file and returns a corresponding file object.

Python open() Function Example

```
# opens python.text file of the current directory
f = open("python.txt")
# specifying full path
f = open("C:/Python33/README.txt")
```

Output:

Since the mode is omitted, the file is opened in 'r' mode; opens for reading.

PYTHON SORTED() FUNCTION

Python sorted() function is used to sort elements. By default, it sorts elements in an ascending order but can be sorted in descending also. It takes four arguments and returns a collection in sorted order. In the case of a dictionary, it sorts only keys, not values.

Python sorted() Function Example

```
str = “Sonia Anand" # declaring string
# Calling function
sorted1 = sorted(str) # sorting string
# Displaying result
print(sorted1)
```

PYTHON NEXT() FUNCTION

Python next() function is used to fetch next item from the collection. It takes two arguments, i.e., an iterator and a default value, and returns an element.

This method calls on iterator and throws an error if no item is present. To avoid the error, we can set a default value.

Python next() Function Example

```
number = iter([256, 32, 82]) # Creating iterator
# Calling function
item = next(number)
# Displaying result
print(item)
# second item
item = next(number)
print(item)
# third item
item = next(number)
print(item)
```

Output:

```
256
32
82
```

PYTHON POW() FUNCTION

The python pow() function is used to compute the power of a number. It returns x to the power of y. If the third argument(z) is given, it returns x to the power of y modulus z, i.e. (x, y) % z.

Python pow() function Example

```
# positive x, positive y (x**y)
print(pow(4, 2))

# negative x, positive y
print(pow(-4, 2))

# positive x, negative y (x**-y)
print(pow(4, -2))

# negative x, negative y
print(pow(-4, -2))
```

Output:

```
16
16
0.0625
0.0625
```

PYTHON REVERSED() FUNCTION

The python reversed() function returns the reversed iterator of the given sequence.

Python reversed() function Example

```
# for string
String = 'Java'
print(list(reversed(String)))

# for tuple
Tuple = ('J', 'a', 'v', 'a')
print(list(reversed(Tuple)))

# for range
Range = range(8, 12)
print(list(reversed(Range)))

# for list
List = [1, 2, 7, 5]
print(list(reversed(List)))
```

Output:

```
['a', 'v', 'a', 'J']
['a', 'v', 'a', 'J']
[11, 10, 9, 8]
[5, 7, 2, 1]
```

PYTHON ROUND() FUNCTION

The python round() function rounds off the digits of a number and returns the floating point number.

Python round() Function Example

```
#  for integers
print(round(10))

#  for floating point
print(round(10.8))

#  even choice
print(round(6.6))
```

Output:

```
10
11
7
```

PYTHON STRING METHODS

Method	Description
capitalize()	Converts the first character to upper case
casefold()	Converts string into lower case
center()	Returns a centered string
count()	Returns the number of times a specified value occurs in a string
encode()	Returns an encoded version of the string
endswith()	Returns true if the string ends with the specified value
expandtabs()	Sets the tab size of the string
find()	Searches the string for a specified value and returns the position of where it was found
format()	Formats specified values in a string
format_map()	Formats specified values in a string
index()	Searches the string for a specified value and returns the position of where it was found
isalnum()	Returns True if all characters in the string are alphanumeric
isalpha()	Returns True if all characters in the string are in the alphabet

isascii()	Returns True if all characters in the string are ascii characters
isdecimal()	Returns True if all characters in the string are decimals
isdigit()	Returns True if all characters in the string are digits
isidentifier()	Returns True if the string is an identifier
islower()	Returns True if all characters in the string are lower case
isnumeric()	Returns True if all characters in the string are numeric
isprintable()	Returns True if all characters in the string are printable
upper()	Converts a string into upper case
zfill()	Fills the string with a specified number of 0 values at the beginning

isspace()	Returns True if all characters in the string are whitespaces
istitle()	Returns True if the string follows the rules of a title
isupper()	Returns True if all characters in the string are upper case
join()	Converts the elements of an iterable into a string
ljust()	Returns a left justified version of the string
lower()	Converts a string into lower case
lstrip()	Returns a left trim version of the string
maketrans()	Returns a translation table to be used in translations
partition()	Returns a tuple where the string is parted into three parts
replace()	Returns a string where a specified value is replaced with a specified value
rfind()	Searches the string for a specified value and returns the last position of where it was found
rindex()	Searches the string for a specified value and returns the last position of where it was found
rjust()	Returns a right justified version of the string

rpartition()	Returns a tuple where the string is parted into three parts
rsplit()	Splits the string at the specified separator, and returns a list
rstrip()	Returns a right trim version of the string
split()	Splits the string at the specified separator, and returns a list
splitlines()	Splits the string at line breaks and returns a list
startswith()	Returns true if the string starts with the specified value
strip()	Returns a trimmed version of the string
swapcase()	Swaps cases, lower case becomes upper case and vice versa
title()	Converts the first character of each word to upper case
translate()	Returns a translated string