SDIT

Built-in Functions in Python

Shouke Wei, Ph.D. Professor

Email: shouke.wei@gmail.com

1. Built-in Functions

Table 1: Python built-in functions

Built-in Functions			
A	**E**	**L**	**R**
abs()	enumerate()	len()	range()
aiter()	eval()	list()	repr()
all()	exec()	locals()	reversed()
any()			round()
anext()	**F**	**M**	
ascii()	filter()	map()	**S**
	float()	max()	set()
B	format()	memoryview()	setattr()
bin()	frozenset()	min()	slice()
bool()			sorted()
breakpoint()	**G**	**N**	staticmethod()
bytearray()	getattr()	next()	str()
bytes()	globals()		sum()
		O	super()
C	**H**	object()	
callable()	hasattr()	oct()	**T**
chr()	hash()	open()	tuple()
classmethod()	help()	ord()	type()
compile()	hex()		
complex()		**P**	**V**
	I	pow()	vars()
D	id()	print()	
delattr()	input()	property()	**Z**
dict()	int()		zip()
dir()	isinstance()		
divmod()	issubclass()		**_**
	iter()		__import__()

Source: https://docs.python.org/3/library/functions.html (https://docs.python.org/3/library/functions.html)

We have already touched:

- bool()
- complex()
- dict()
- float()

- format()
- input()
- int()
- len()
- list()
- print()
- range()
- set()
- str()
- tuple()
- type()

2. Examples

(1) abs(x)

- Return the absolute value of a number.

In [1]:
```
a = -5
b = abs(a)

print(b)
```

```
5
```

(2) max() and min()

- Return the largest or smallest item in an iterable or two or more arguments

In [1]:
```
c = [-2,4,5,-10]
print('Maximun:',max(c))
print('Minimun:',min(c))
```

```
Maximun: 5
Minimun: -10
```

(3) round ()

- Return number rounded to *ndigits* precision after the decimal point.

In [5]:
```
x = 5.456
print(round(x))
```

```
5
```

In [6]:
```
y = 5.5555
print(round(y))
```

Out[6]:
```
6
```

In [8]:
```
z = - 5.5555
print(round(z))
```

```
-6
```

(4) zip ()

- Iterate over several iterables in parallel, producing tuples with an item from each one.

In [17]:
```
fruits = ['apple', 'banana', 'cherry','organge']
price = [2.59, 0.66, 3.29, 4.45]

fruit_price = zip(fruits, price)

for price in fruit_price:
    print(price)
```

```
('apple', 2.59)
('banana', 0.66)
('cherry', 3.29)
('organge', 4.45)
```

In [9]:
```
fruits = ['apple', 'banana', 'cherry','organge']
price = [2.59, 0.66, 3.29, 4.45]

fruit_dict = {fruits: price for fruits,
            price in zip(fruits, price)}
print(fruit_dict)
```

```
{'apple': 2.59, 'banana': 0.66, 'cherry': 3.29, 'organge': 4.45}
```

(5) enumerate ()

`enumerate(iterable, start)` : adds counter to an iterable and returns it, i.e. an enumerate object.

Parameters:

- iterable: an iterable object
- start: a Number, which defines the start number of the enumerate object. Default is 0

In [16]:
```
# Python program to illustrate
# enumerate function in loops
fruit_list = ['apple', 'banana', 'cherry','organge']

# printing the tuples in object directly
for fruit in enumerate(fruit_list):
    print (fruit)

# changing index and printing separately
for n, fruit in enumerate(fruit_list, 100):
    print (n, fruit)

# getting desired output from list
for n, fruit in enumerate(fruit_list):
    print(n)
    print(fruit)
```

```
(0, 'apple')
(1, 'banana')
(2, 'cherry')
(3, 'organge')
100 apple
101 banana
102 cherry
103 organge
0
apple
1
banana
2
cherry
3
organge
```