enumerate()

A lot of times when dealing with iterators, we also get a need to keep a count of iterations. Python eases the programmers' task by providing a built-in function enumerate() for this task.
Enumerate() method adds a counter to an iterable and returns it in a form of enumerate object. This enumerate object can then be used directly in for loops or be converted into a list of tuples using list() method.

Ex:

```
# enumerate function
l1 = ["eat","sleep","repeat"]
s1 = "geek"

# creating enumerate objects
obj1 = enumerate(l1)
obj2 = enumerate(s1)

print ("Return type:",type(obj1))
print (list(enumerate(l1)))

# changing start index to 2 from 0
print (list(enumerate(s1,2)))
```

```
Return type: < type 'enumerate' >
[(0, 'eat'), (1, 'sleep'), (2, 'repeat')]
[(2, 'g'), (3, 'e'), (4, 'e'), (5, 'k')]
```

Lambda Expression

An anonymous function refers to a function declared with no name. Although syntactically they look different, lambda functions behave in the same way as regular functions that are declared using the def keyword but we use the lambda keyword instead

syntax: lambda argument(s): expression

Ex:

```
remainder = lambda num: num % 2
print(remainder(5))
```

output: 1

The operator Module

The operator module supplies functions that are equivalent to Python's operators. These functions are handy in cases where callables must be stored, passed as arguments, or returned as function results. The functions in operator have the same names as the corresponding special methods.

The operator module functions: