Garbage Collector

Garbage Collector

Python's memory allocation and deallocation method is automatic. The user does not have to preallocate or deallocate memory by hand as one has to when using dynamic memory allocation in languages such as C or C++. Python uses two strategies for memory allocation **reference counting** and **garbage collection**.

The chosen method is called *reference counting*. The principle is simple: every object contains a counter, which is incremented when a reference to the object is stored somewhere, and which is decremented when a reference to it is deleted. When the counter reaches zero, the last reference to the object has been deleted and the object is freed.

Garbage Collector

```
>>> import sys
>>> help(sys.getrefcount)
...
>>> one = []
>>> print 'At start         :',  sys.getrefcount(one)
At start         : 2
>>> two = one
>>> print 'Second reference :', sys.getrefcount(one)
Second reference : 3
>>> del two
>>> print 'After del        :', sys.getrefcount(one)
After del        : 2
```

Garbage Collector

```
>>> help(globals)

>>> for k in globals().keys():
...     print k ,  str(sys.getsizeof(globals()[k]))
```

Garbage Collector

```
>>> def make_cycle(l):
...     l.append(l)

>>> lista  = []
>>> [make_cycle(lista) for i in xrange(100000000) ]
>>> print sys.getrefcount(l)
...
>>> sys.getsizeof(lista)
>>> import gc
>>> gc.collect()
4
```