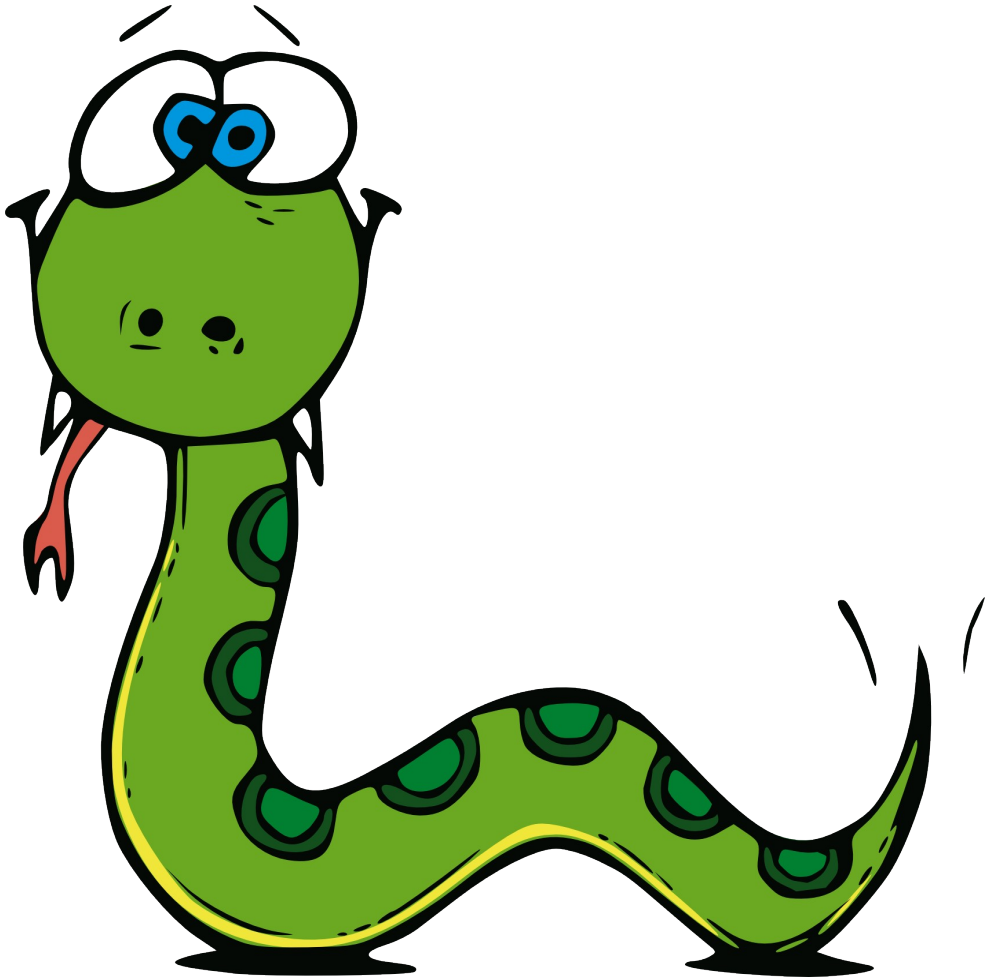
Introduzione a Python



Riccardo Lemmi

Python è

Un linguaggio di alto livello

```
1 import random
2
3 def generate_dict(n):
4     random.seed(0)
5     dict = {}
6     for i in range(0,n):
7         dict['p%i'%i] = {'score': int(random.random()*n)}
8     return dict
9
10 def sort(v):
11     tmplist = [(v['score'],k) for k,v in dict.iteritems(v)]
12     tmplist.sort()
13     return [{k:v} for v,k in tmplist]
14
15 d = generate_dict(10)
16 print d
17 print sort(d)
```

Python è

Un linguaggio interpretato

Ruby

A Programmer's Best Friend

Perl

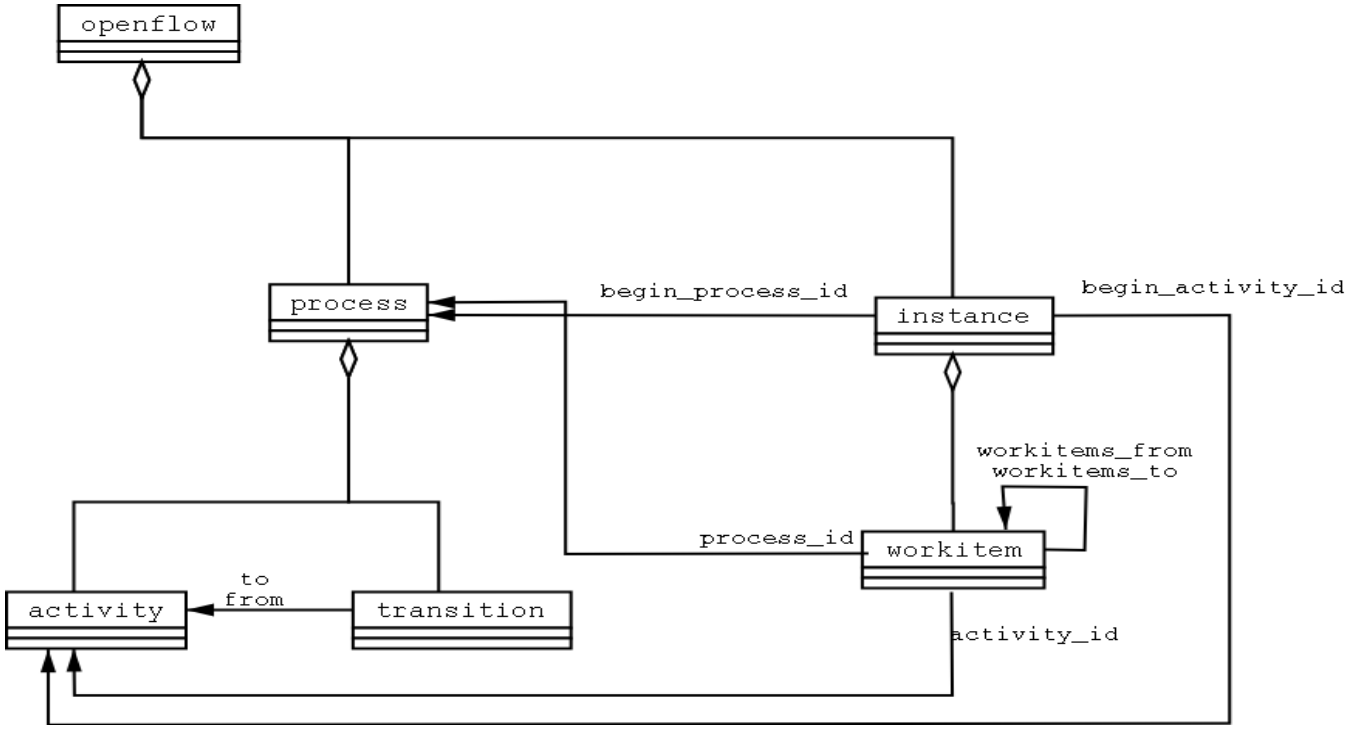
Python è

Un linguaggio interattivo

```
axa@motoko:~
 $ python
Python 2.6.5 (r265:79063, Oct  1 2012, 22:07:21)
[GCC 4.4.3] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import math
>>> for x in range(1,10):
...     print math.sin(x/10.0)
...
0.0998334166468
0.198669330795
0.295520206661
0.389418342309
0.479425538604
0.564642473395
0.644217687238
0.7173560909
0.783326909627
>>>
```

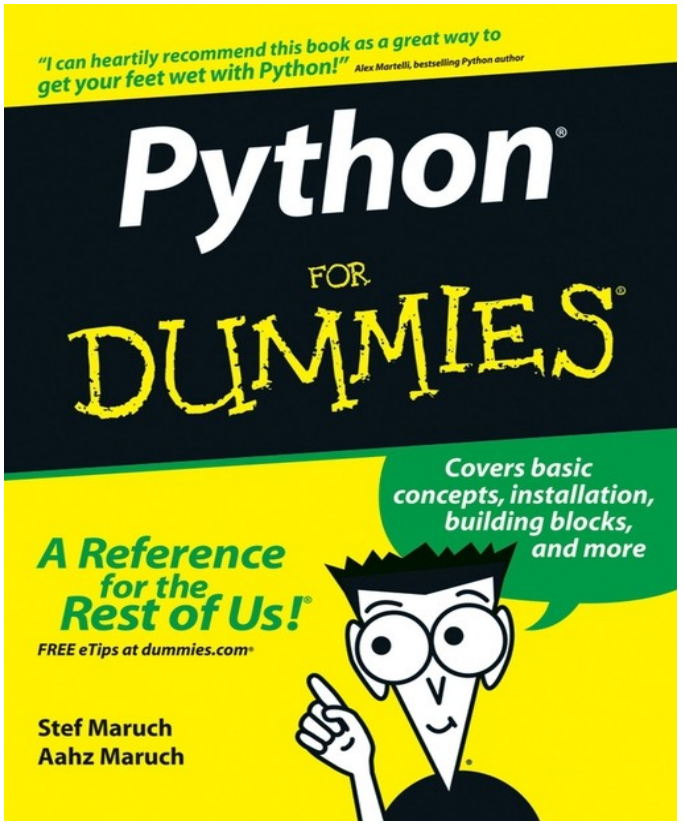
Python è

Un linguaggio object-oriented



Python è

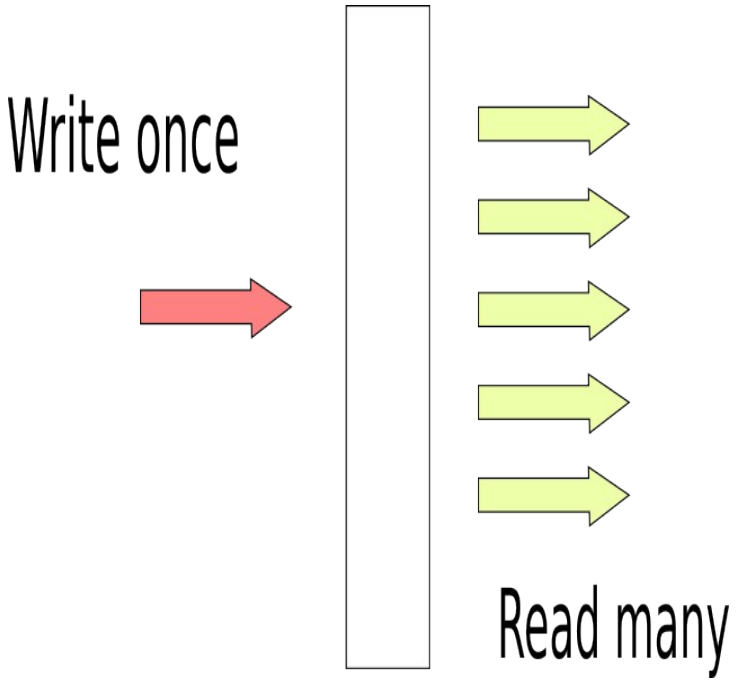
Un linguaggio per i principianti



Python è

Un linguaggio per tutti i campi applicativi

Punti di forza



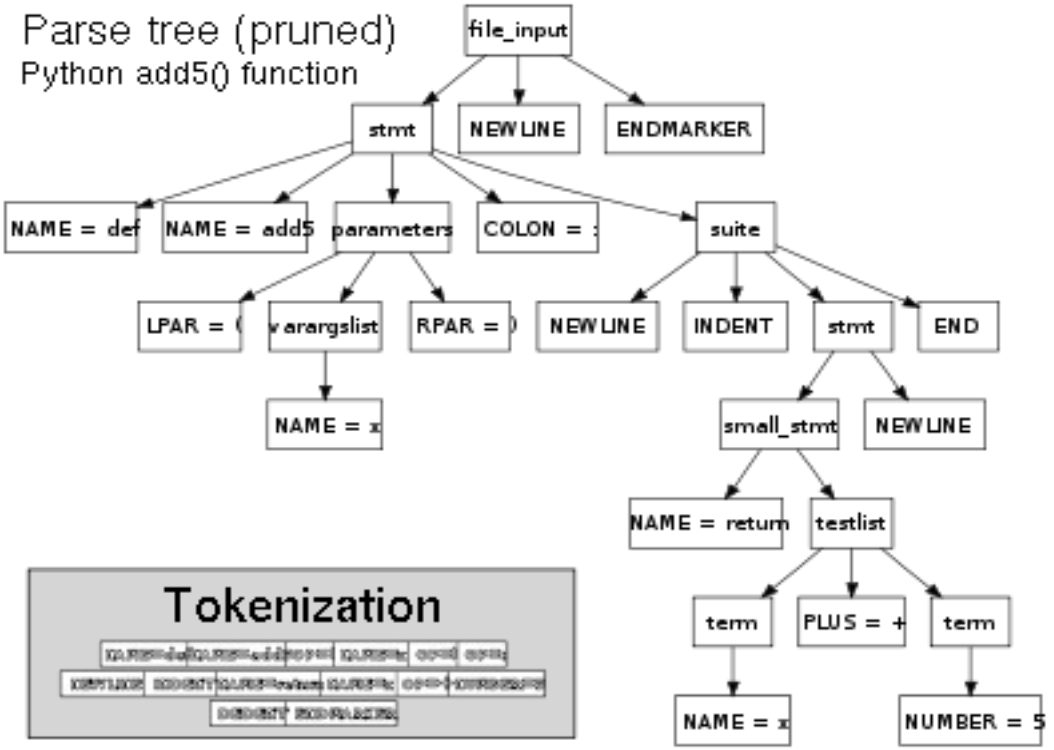
- Facile da scrivere
- Facile da leggere
- Facile da mantenere

Punti di forza

Le librerie standard coprono da sole le necessità più comuni

String Services
Data Types
Numeric and Mathematical Modules
File and Directory Access
Data Persistence
...

Sintassi di base



Indentazione

```
def generate_dict(n):
        random.seed(0)
        dict = {}
        for i in range(0,n):
                dict['p%i'%i] = {'score': int(random.random()*n)}
        return dict

print generate_dict(10)
```

Operatori

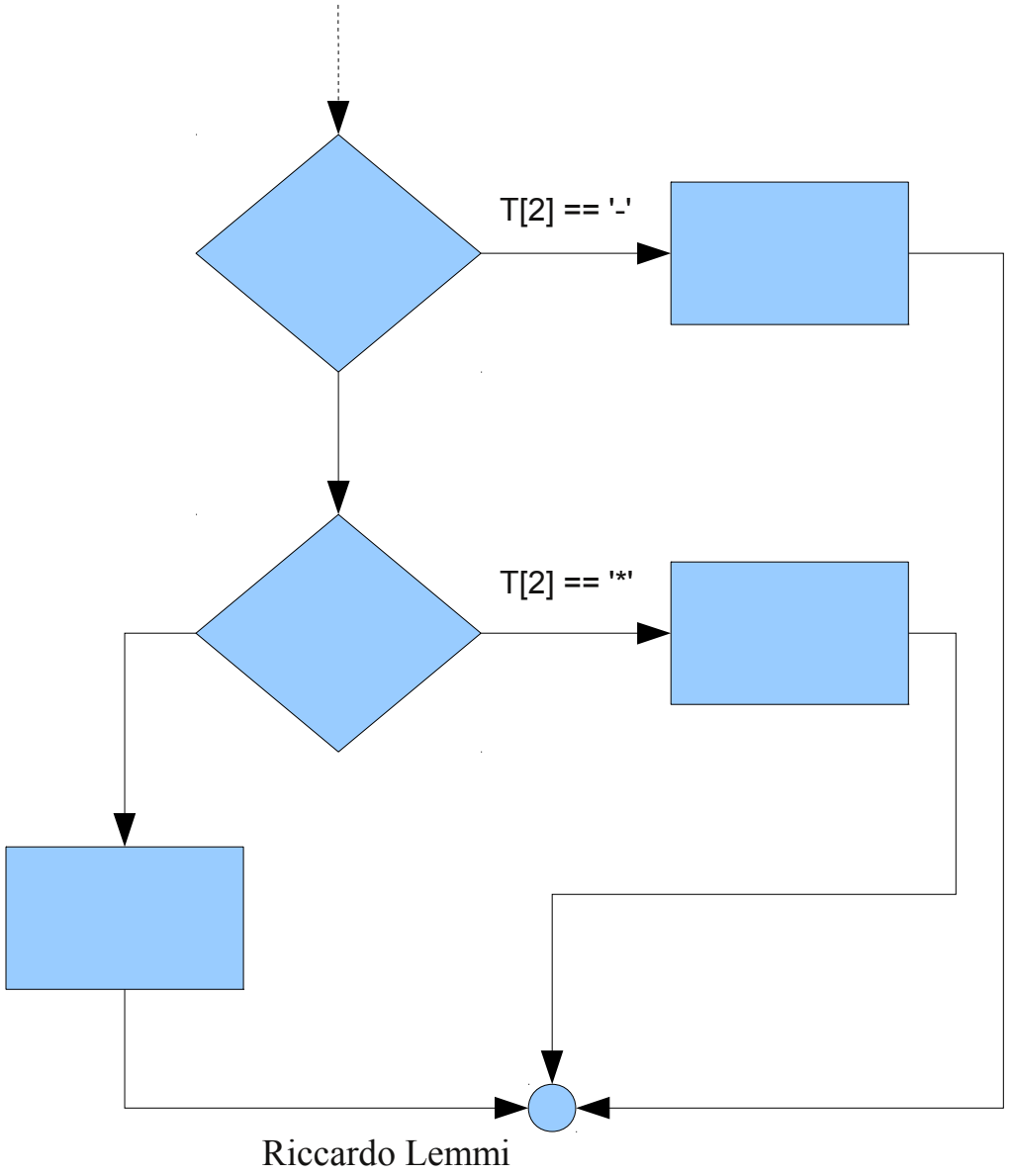
Tutti quelli che si trovano negli altri linguaggi
...ma senza le cose strane:

== vs ===

and vs &&

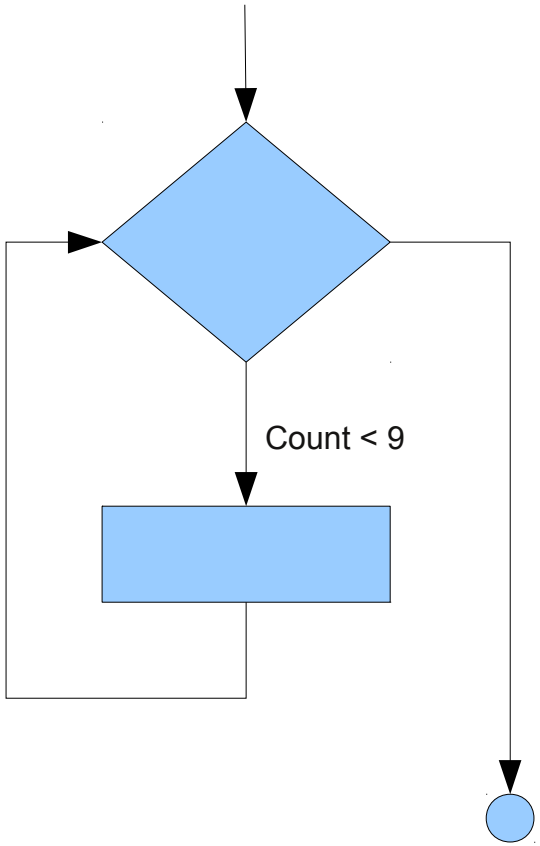
...

if



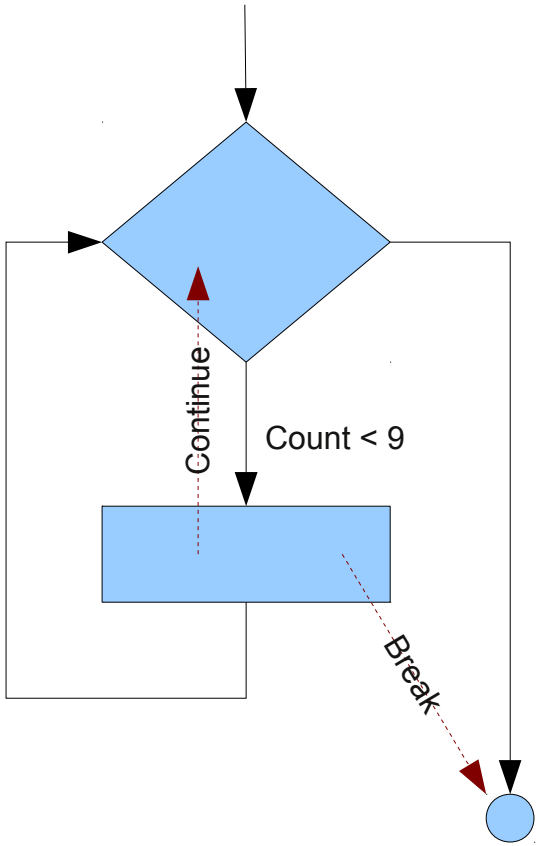

```
if t[2] == '+'  :
    t[0] = t[1] + t[3]
elif t[2] == '-':
    t[0] = t[1] - t[3]
elif t[2] == '*':
    t[0] = t[1] * t[3]
else:
    t[0] = t[1] / t[3]
```

while



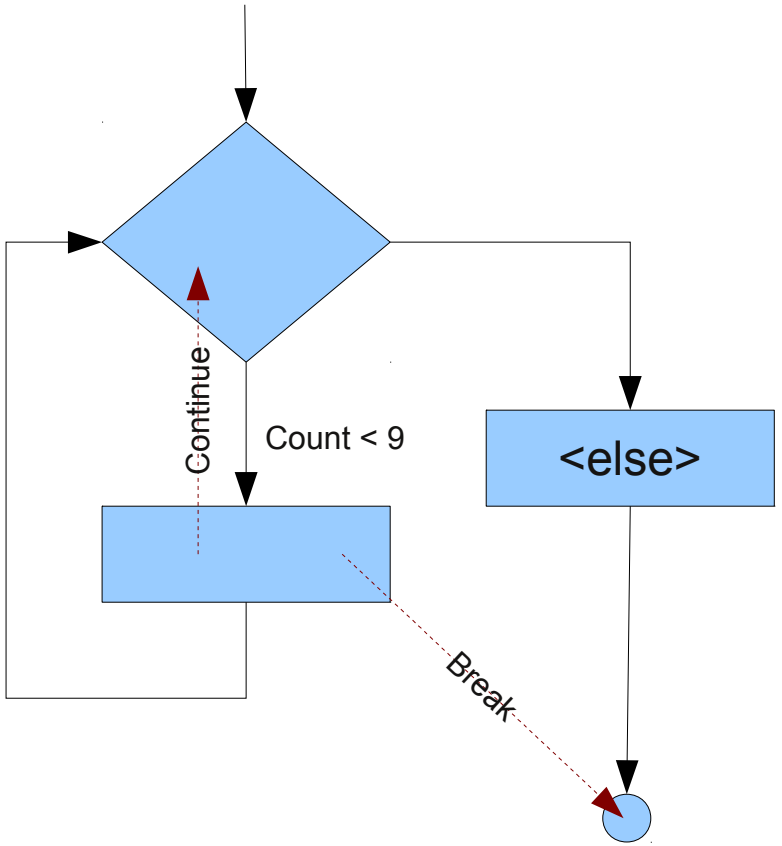

```
count = 0
while count < 9:
    print 'The count is:',count
    count = count + 1
print "Good bye!"
```

while



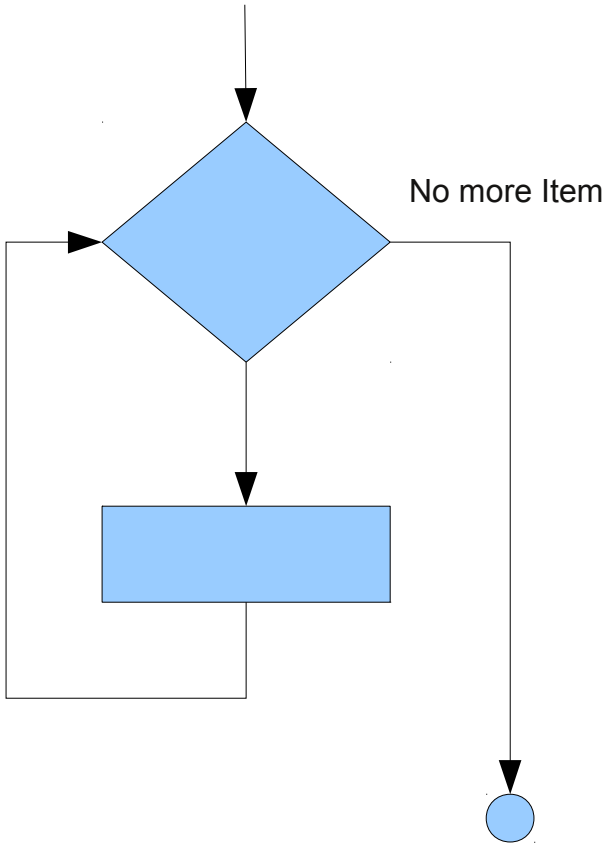

```
count = 0
while count < 9:
    print 'The count is:',count
    count = count + 1
    if count ==5:
        continue
print "Good bye!"
```

while



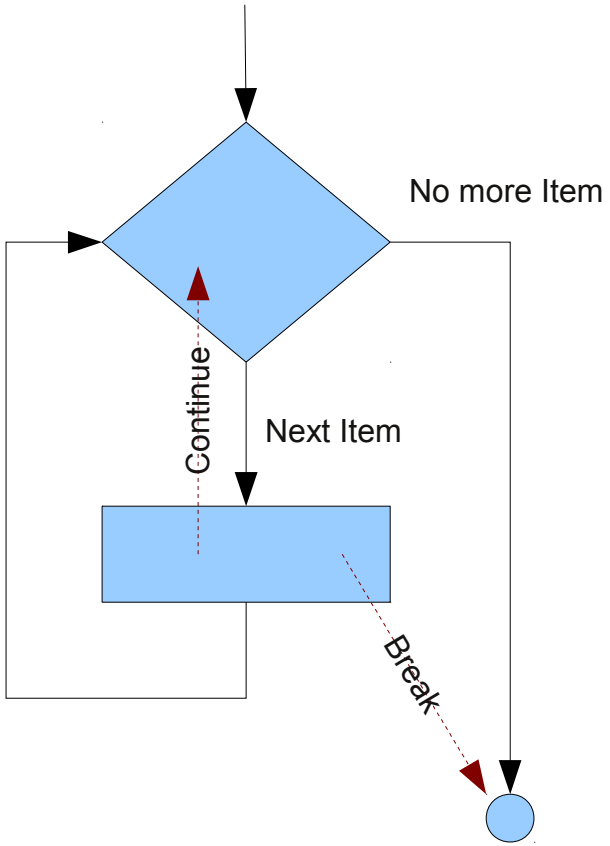

```
count = 0
while count < end:
    print 'The count is:',count
    count = count + 1
    if count == 5:
        break
else:
    print "Counted all"
```

for



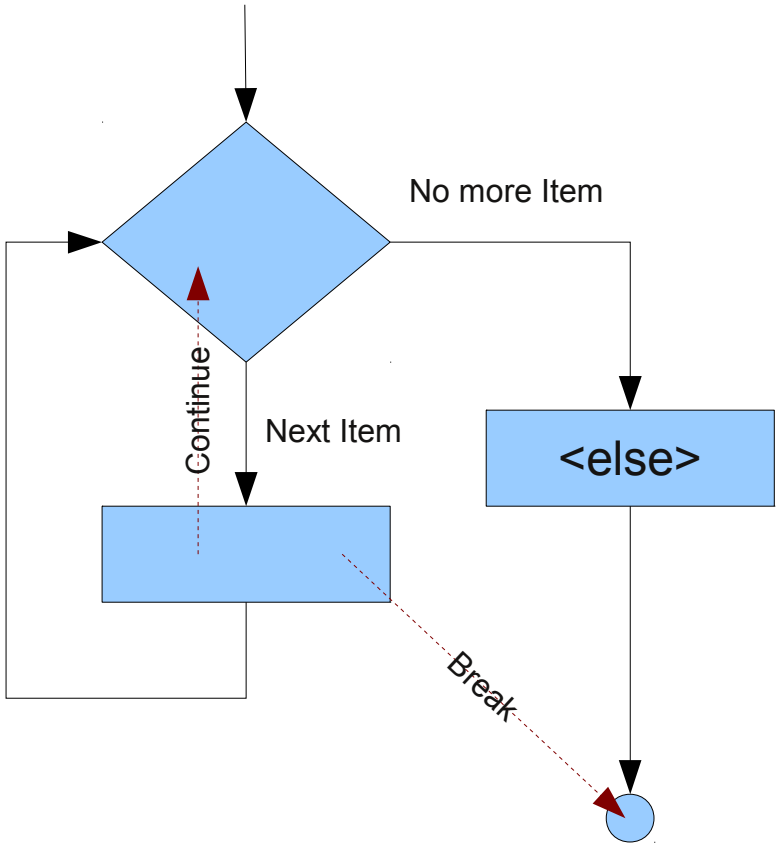

```
for i in range(1,10):
    print i**2
```

for




```
for i in range(1,10):
    print i**2
    if i == 5:
        continue
```

for




```
for i in range(1, x):
    print i**2
    if i == 5:
        break
else:
    print “that's all”
```

pass

```
x = 0
try:
    x =  y / z
except:
   pass
...
```

Numeri

- Interi:
  - int 10
  - long 10L
- Float 10.0
- Complex 10+10j

Stringhe

str1 = 'LinuxDay 2012'

str2 = “LinuxDay 2012”

slice […]

str1[5] → 'D'

str1[:5] → 'Linux'

str1[-4:] → '2012'

str1[5:8] → 'Day'

Stringhe

Sono IMMUTABILI

```
str = 'LinuxDay 2012'

str[0] = 'l'
str[1:5] = 'Windows'
del str[0]
```

generano una eccezione

Stringhe

```
str = 'LinuxDay 2012'
str = 'Windows'+str[5:]
```

Liste

list1 = [1,2,'a',3,'b']

list1[0] → 1

list[:4] → [1,2,'a']

…

sono mutabili

list[1] = 1 → [1,1,'a',...

del list[0] → [2,'a',...

Tuple

tuple1 = (1,2,'a',3,'b')

tuple1[0] → 1

tuple[:4] → [1,2,'a']

...

ma sono IMmutabili

list[1] = 1

del list[0]

e simili lanciano una eccezione

Dizionari

dict1 = {'a':1, 'b':2, 'c':4, 4:'4', 'd':[1,2,3]}

dict1['a'] = 0 → {'a':0, …

dict1['a'] = [4, 5, 6] → {'a':[4,5,6], ...

del dict['a] → {'b':2, ...

for e tipi enumerabili

```
for x in [1,2,3]: …
for x in (1,2,3): …
for x in {'a':1, 'b',2}: …
for line in f.readlines(): ...
```

il 'while' serve a poco

Generatori

yield

```
def counter(start, end):
    count = start
    while count <= end:
        yield count
        count += 1
```

Generatori: Tipi

```
class firstn(object):
    def __init__(self, n):
        self.n = n
        self.num, self.nums = 0, []

    def __iter__(self):
        return self

    def next(self):
        if self.num < self.n:
            cur, self.num = self.num, self.num+1
            return cur
        else:
            raise StopIteration()
```

Funzioni

```
def f(x):
    return x**2
```

Parametri

```
def f(x,y=1):     # posizionali e per nome
    …

def f(*args, **kargs):
    ...
```

Visibilita' variabili

Global

- Le variabili definite a livello di modulo

Local

- Le variabili definite in una funzione

Visibilita' variabili

```
total = 0         # This is a global variable.
def sum( arg1, arg2 ):
   total = arg1 + arg2    # total is a local variable.
   return total
```

Visibilita' variabili

```
total = 0       # This is a global variable.
def sum( arg1, arg2 ):
   global total             # total is global
   tot = arg1 + arg2    # tot is a local variable.
   total += tot
   return tot
```

Classi

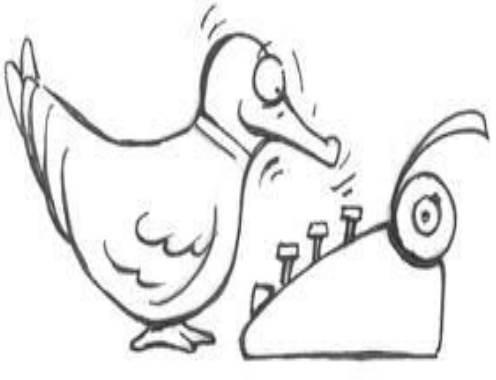
```
class Employee:
    """Common base class for all employees"""
    empCount = 0

    def __init__(self, name, salary):
        self.name = name
        self.salary = salary

    def pretty_repr(self):
        return '<Employee %s>'%self.name

Employee.empCount += 1
```

Duck Typing



“When I see a bird that walks like a duck and swims like a duck and quacks like a duck,

I call that bird a duck.”

“Se si comporta come un'oca...

deve essere un oca”

Ereditarietà

```
class Person(object):
    def __init__(self, name):
        self.name = name

class Employee(Person):
    ""Common base class for all employees""
    def __init__(self, name, salary):
        super(Person, self).__init__(name)
        self.salary = salary
```

Riferimenti

- http://docs.python.org/
- http://www.learnpython.org/
- http://en.m.wikipedia.org/wiki/Duck_typing
- http://www.tutorialspoint.com/python

La maggior parte delle immagini sono prese da internet: ai rispettivi autori il credito.