PROGRAMACIÓN EN PYTHON

Patricio Páez Serrato

Índice

- Instalación
- Lenguaje
- Ejemplos

El lenguaje

- Tipos de datos
- Expresiones
- Funciones predefinidas
- Entrada y Salida
- Control de flujo
- Funciones de usuario
- Cadenas y listas

Programación orientada a objetos

- Objetos
  - Entidad con datos y procedimientos
  - Objeto.atributo
  - Objeto.método()

Tipos de datos

- Enteros
- Punto flotante
- Cadenas
- Listas

Enteros y flotantes

1234 12345679 0x003f 0776
10L 1234567890123490L
0.5 5e10
1.5+0.3j

Cadenas

’hola mundo’
"Python es fácil de aprender"
’con "las baterías incluídas".’ "Guido’s"
’’’Esta cadena
tiene tres
renglones’’’

Listas

[0,1,2,3,4,5]

[’hola’, ’mundo’, 1, 5.6, 0x0ff, [1,2,3]]

[[1,2,3], [4,5,6], [7,8,9]]

Expresiones

- Operadores
- Variables
- Valor lógico

Operadores

```
>>> -5**2
-25
```

```
min <= y <= max
```

```
7 in [ 10, 5, 7, 8 ]
```

```
'c' in 'Python es conciso'
```

Funciones predefinidas

- Integradas
- Definidas en módulos

Funciones integradas

```
>>> int(4.5)
4
>>> long(5)
5L
 >>> float(22)
22.0
```

Funciones integradas

```
>>> hex(255)
'0xff'
>>> oct(255)
'0377'
>>> round(1.7)
2.0
```

Funciones integradas

```
>>> abs(-5)
5
>>> abs( 1 + 1j)
1.4142135623730951
>>> round( _, 3)
1.4139999999999999
```

Funciones definidas en módulos

```
>>> from math import pi,sin
>>> pi
3.141592653589793
>>> sin(pi/2)
1.0
```

Entrada y Salida

- raw_input()
- print
- open()
- read()

Control de flujo

- if
- for
- while
- continue
- break
- pass

Control de flujo

```
if expresion:
    bloque
elif expresion:
    bloque
else:
    bloque
```

Control de flujo

```
for elemento in secuencia:
    bloque
while expresion:
    bloque
else:
    bloque
```

Funciones de usuario

- def
- return
- lambda

Funciones de usuario

```
def nombre( [parámetros] ):
    "Documentación."
    bloque
   [return expresión]
nombre( [argumento [,argumento ...] ] )
```

Funciones lambda

lambda argumentos: expresión

lambda x,y : x*y

Operaciones con cadenas y listas

- len()
- Subíndices [n]
- Cortes [i:f]
- e in s
- Métodos

Operaciones con cadenas y listas

len(’1234’)
4
len([’0’])
1

Operaciones con cadenas

s = ’hola mundo’
s[7]
s[-1]
s[11]
s[2:4]

Operaciones con listas

l = [10, 4, 7, ’cadena’, [22, 23], (x,y)]

l[2]

l [0:2]

l [3:5]

Operaciones con listas

l = [10, 4, 7, ’cadena’, [22, 23], (x,y)]

l.append(’cabus’)

[10, 4, 7, ’cadena’, [22, 23], (x,y), cabus]

l.insert(4, 5)

[10, 4, 7, ’cadena’, 5, [22, 23], (x,y), cabus]

Operaciones con listas

[10, 4, 7, ’cadena’, 5, [22, 23], (x,y), cabus]

l.pop(3)

’cadena’

l

[10, 4, 7, 5, [22, 23], (x,y), cabus]

Listas - métodos

- sort()
- reverse()
- sort(cmpfunc)

Listas - Usos

- Pila LIFO
- Cola FIFO
- Matriz
- Lista recursiva

Listas - Usos

- Pila LIFO
  - append(x)

  [1, 2, 3, 4, 5] <---
  - pop()

  [1, 2, 3, 4, 5] --->

Listas - Usos

- Cola FIFO
  - append(x)

[1, 2, 3, 4, 5] <---

  - pop(0)

<--- [1, 2, 3, 4, 5]

Listas - Usos

- Matriz
  - L[r][c]

```
l = [ [1, 2, 3],  # 1er. elemento
      [4, 5, 6],  # 2do.
      [7, 8, 9] ] # 3ro.
l[0]          l[1][2]
[1, 2, 3]       6
```

Caracteres especiales

- #
- \
- ([{
- ;

PROGRAMACIÓN EN PYTHON

www.python.org

pp.com.mx

Patricio Páez Serrato