CUSOL (Comunidad Universitaria de Software Libre)

Prioridad de las operaciones

Para las operaciones matemáticas Python sigue la convención.

- Paréntesis, pueden forzar una expresión para ser evaluada de varias maneras. ()
- Exponenciación. **
- Multiplicación y División tienen la misma precedencia. *, /
- Adición y sustracción tienen la misma precedencia. +, -

Los operadores con la misma precedencia son evaluados de izquierda a derecha, excepto la exponenciación .

Consejo: Use paréntesis

Comentarios

A medida que los programas se hacen más grandes y complicados, se hacen difíciles de leer.

```
# Esto es un comentario de una línea. Una nota para explicar en lenguaje natural.

# Calcular el porcentaje  de hora que ha transcurrido.
porcentaje = (minuto * 100) / 60

#assignar 5 a v

v = 5
```

Funciones

Una función es una secuencia de instrucciones que calculan algo. Una función se define con un nombre a partir del cual se puede “invocar”. Python tiene f**unciones internas** listas para ser usadas.

Ejemplo: type(32()
atype ‘int’>

La expresion entre parentesis es el **argumento . Esta función retorna el tipo de valor.**

Otras funciones internas: int(), str(),float()

Funciones: Módulos

Un módulo es un archivo que provee funciones y variables relacionadas con algún tema. Ejemplo: el módulo **math**

```
>>import math
>>print math

>>print math.pi
>>print math.log10(x)
```

Funciones: Creación

Es posible crear nuevas funciones, se define con la palabra clave `def`:

```
def imprimir_nombre (nombre):
    print “Hola ”, nombre
```

→ Encabezado de la función note el “:”

→ Cuerpo, debe estar identado

```
imprimir_nombre(“Camila”)
```

En este caso “Camila” es el **argumento** que se le pasa a la función.

nombre es el parámetro que recibe el argumento dentro de la función.

Funciones: Retornando un valor

Las funciones pueden retornar un valor o simplemente realizar un procedimiento.

```
def imprimir_nombre (nombre):
    print “Hola ”, nombre
```

→ Encabezado de la función note el “:”

→ Cuerpo, debe estar identado

```
imprimir_nombre(“Camila”)
```

```
def  calcular_edad(nac):
    edad = 2016 - nac
    return edad

imprimir_nombre(“Camila”)
años = calcular_edad(1993)
```

→ Encabezado de la función note el “:”

Funciones: ¿Para qué?

- Programas menos repetitivos.
- Fácil debug.
- Código mejor diseñado.
- Fácil de leer.

Funciones: Importando

Para usar un módulo se pueden usar las palabras import y from

```
>>import math
>> math.pi

>>from math import *
>>pi


>>from math import pi
```

Funciones: Importando

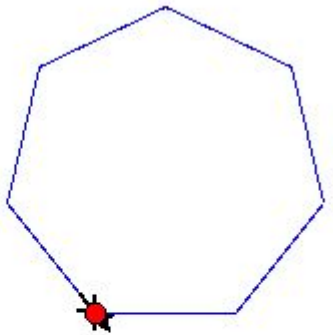
Para usar un módulo se pueden usar las palabras import y from

```
>>import math
>> math.pi

>>from math import *
>>pi


>>from math import pi
```

Ejercicios: TurtleWorld

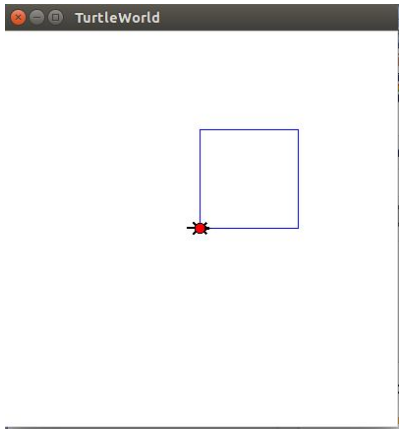
Instalar el módulo swampy y realizar los ejercicios.

- Página 33 de Think Python



1- Ejercicios: TurtleWorld

Encapsulación: envolver un fragmento de código en una función.



2- Ejercicios: TurtleWorld

Generalización: Cuando las funciones tienen parámetros.

3- Interfaz

Las interfaces proveen funciones “limpias” , solo deben tener los parámetros de entrada y el valor de retorno.

4- Refactorizar

Es arreglar un programa para mejorar interfaces de las funciones y por ende su uso.