TEC | Tecnológico de Costa Rica

Primeros Pasos en Python 3

Introducción a la programación

II semestre, 2019

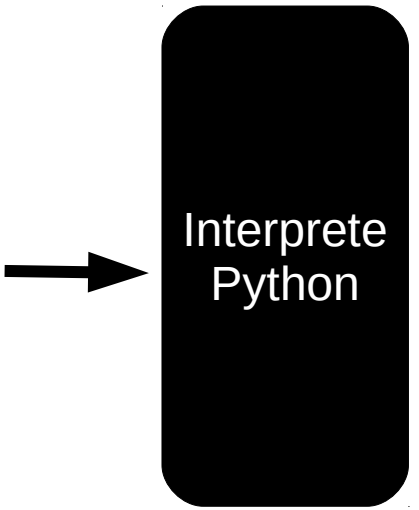
Escribir Programas en Python

Un programador debe utilizar **expresiones, funciones y métodos de forma combinada** para obtener **soluciones a problemas**.

Un **programa** en Python:

- **Archivo con extensión .py (ejemplo: circulo.py)**
- Archivo C o C++ compilados
- Un módulo cargado dinámicamente (**import**)

```
materials = []
currentMaterial = defaultMaterial
for line in self.contents.split("\n"):
    if line[:6] == 'mtllib':
        filename = ' '.join(line.split(' ')[1:])
        materials.extend(self.loader.load(filename, silent))
....
    if line[:6] == 'usemtl':
        name = line.split(' ')[1:][0]
        if name == '(null)':
            currentMaterial = defaultMaterial
            continue
        for material in materials:
```



Escribir Programas en Python

```
#!/usr/bin/python3
#Archivo: area-circulo.py

#Calcular área de un circulo
# A = Pi * r **2
radio = 10
area = 3.141516 * radio ** 2
print(area)
```

*… y ¿si después quiero **volver a calcular** el área de un circulo?*

Definición de una función

Una **mejor forma** para escribir programas es por medio de **funciones**.

```
def nombre_función(argumentos):
    Cuerpo de la función
```

Nombre de la función es un **identificador válido**

Definición de una función

```
if nombre_función(argumentos):
    Cuerpo de la función
```

Los **argumentos** son las **entradas** que recibe el programa. Se presentan como nombres simbólicos o identificadores internos que serán sustituidos.

Cuerpo de la función: **lógica** de la solución. Expresiones, combinaciones de estructuras permitidas, podría contener invocaciones a otras funciones.

Notas de **Sintaxis**: dos puntos **(:)** e **identación**.

Invocar funciones

Las funciones pueden ejecutarse directamente desde el intérprete o desde otro programa.

```
>>> nombre_función(valores_argumentos)
Resultado
```

Los **argumentos son los valores de entrada** que serán asociados a los nombres simbólicos (identificadores internos que definimos)

La **invocación de funciones dentro de otro programa se realiza igual**, pero el resultado no se presentará en el intérprete sino que estará a disposición de “quien” invocó la función.

Invocar funciones

```
>>>nombre_función(valores_argumentos)
Resultado
```

Los **argumentos son el equivalente a las entradas** que serán asociados a los nombres simbólicos. Un tipo distinto de variable

La **invocación de funciones dentro de otro programa se realiza igual.** pero el resultado no se presentará en pantalla debe ser guardado.

Escribir Programas en Python (nivel 2)

```
#!/usr/bin/python3
#archivo: area-circulo2-v2.py


#Calcular área de un circulo
# A = Pi * r **2
def area_circulo(radio):
   area = 3.141516 * radio ** 2
   return area

resultado = area_circulo(10)
print(resultado)
```

Escribir Programas en Python (v2)

```
#!/usr/bin/python3
#Archivo: area-circulo2-v2.py


#Calcular área de un circulo
# A = Pi * r **2
def area_circulo(radio):
    area = 3.141516 * radio ** 2
    return area


resultado = area_circulo(10)
print(resultado)
```

Variable local

Únicamente es **conocida dentro del ámbito de la función** en la cual está definida. Fuera de la función “área” no existe.

Documentación interna del código

Dos formas de escribir comentarios en el código:

- **De una línea: se utiliza el símbolo #**

```
#!/usr/bin/python3
# Función que calcula el área de un circulo
# …
def area_circulo(radio):
    area = 3.141516 * radio ** 2  # A = Pi * r **2
    return area
```

Documentación interna del código

Dos formas de escribir comentarios en el código:

- **Multilínea: se definen con tres comillas**

```
#!/usr/bin/python3
'''
 Función que calcula el área de un circulo
 Entradas: radio.
'''
def area_circulo(radio):
      area = 3.141516 * radio ** 2   # A = Pi * r **2
    return area
```

Documentación interna del código

Dos formas de escribir comentarios en el código:

- **Multilínea: se definen con tres comillas**

```
#!/usr/bin/python3
def area_circulo(radio):
    '''
 Función que calcula el área de un circulo
 Entradas: radio.
'''
    area = 3.141516 * radio ** 2   # A = Pi * r **2
    return area
```

Documentación interna del código

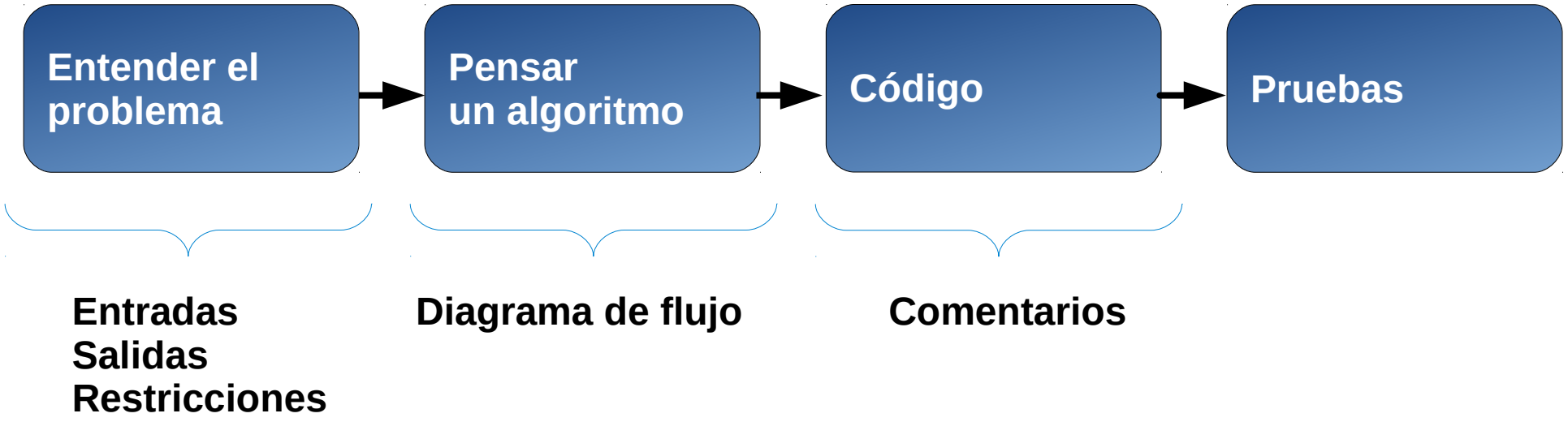
```
#!/usr/bin/python3
def area_circulo(radio):
    '''
    Función que calcula el área de un circulo
    E: radio.
    '''
     area = 3.141516 * radio ** 2  # A = Pi * r **2
     return area
```

```
>>> print(area_circulo.__doc__)
```

Ejercicio: Área y Perímetro de un rectángulo

Escriba un programa de python que contenga dos funciones para operar con **rectángulos:**

- **Área = Base * Altura**
- **Perímetro = Base + Base + Altura + Altura**



Errores.... (pulgas o bugs)

Dos tipos de errores:

- **Sintácticos**: se da por **incumplimiento de las reglas** definidas para el lenguaje.

- **Semánticos**: se dan en tiempo de ejecución y representar **fallos en la lógica** del programa. Si hay resultados pero no son los esperados.

Ejercicio: Conversión de grados

Escriba **dos funciones**, bien **documentas**, para **convertir grados centígrados a farenheit**.

Formulas:

- Farenheit = 9 /5 * centígrados + 32
- Centígrados: (Farenheit – 32) / 1,8

Ejercicio: Conversión de grados

Más práctica:

- Construya una función para cada una de las conversiones que aparecen en esta tabla:

  http://es.wikipedia.org/wiki/Grado_Fahrenheit#Conversi.C3.B3n_a_otras_unidades

Más Información

Funciones:

- http://docs.python.org/release/3.1.3/tutorial/controlflow.html#defining-functions

Errores:

- http://docs.python.org/release/3.1.3/tutorial/errors.html

Referencias y Lecturas Complementarias

- Material suministrado por el profesor Jeff Schmidt, Instituto Tecnológico de Costa Rica. I semestre 2011.