



UNIVERSITAT
POLITÈCNICA
DE VALÈNCIA

Programación para aplicaciones geoespaciales

Introducción al Lenguaje Python

LAURA SEBASTIÁ

TEMA 4. ESTRUCTURAS DE CONTROL

Contenidos



- 1.- Estructuras condicionales
- 2.- Estructuras iterativas
- 3.- Funciones
- 4.- Excepciones

Bibliografía:


- Introducción a la programación con Python 3
- Tema 4
- Tema 6

TEMA 4. ESTRUCTURAS DE CONTROL

Estructuras de control de flujo

- Sentencias condicionales

```
if <expresión booleana>:  
    instrucción-1  
    ...  
    instrucción-n
```

indentación 

```
if <expresión booleana>:  
    instrucciones  
else:  
    instrucciones
```

```
if <expresión booleana>:  
    instrucciones  
elif <expresión booleana>:  
    instrucciones  
else:  
    instrucciones
```

Se puede repetir tantas veces
como sea necesario



TEMA 4. ESTRUCTURAS DE CONTROL

Estructuras de control de flujo

- Sentencias condicionales: Ejemplos

```
print("Programa para la resolución de la ecuación ax+b=0")
a = float(input("Valor de a:"))
b = float(input("Valor de b:"))

if a!=0:
    x = -b/a
    print("Solución: ", x)
```

Programa para la resolución de la ecuación ax+b=0
Valor de a:3
Valor de b:6
Solución: -2.0

Programa para la resolución de la ecuación ax+b=0
Valor de a:0
Valor de b:6

```
print("Programa para la resolución de la ecuación ax+b=0")
a = float(input("Valor de a:"))
b = float(input("Valor de b:"))

if a!=0:
    x = -b/a
    print("Solución: ", x)
else:
    print("La ecuación no tiene solución")
```

Programa para la resolución de la ecuación ax+b=0
Valor de a:0
Valor de b:6
La ecuación no tiene solución

Programa para la resolución de la ecuación ax+b=0
Valor de a:0
Valor de b:0
La ecuación no tiene solución

TEMA 4. ESTRUCTURAS DE CONTROL

Estructuras de control de flujo

- Sentencias condicionales: Ejemplos

```
print("Programa para la resolución de la ecuación ax+b=0")
a = float(input("Valor de a:"))
b = float(input("Valor de b:"))

if a!=0:
    x = -b/a
    print("Solución: ", x)
elif b!=0:
    print("La ecuación no tiene solución")
else:
    print("La ecuación tiene infinitas soluciones")
```

```
Programa para la resolución de la ecuación ax+b=0
Valor de a:0
Valor de b:0
La ecuación tiene infinitas soluciones
```

TEMA 4. ESTRUCTURAS DE CONTROL

Estructuras de control de flujo

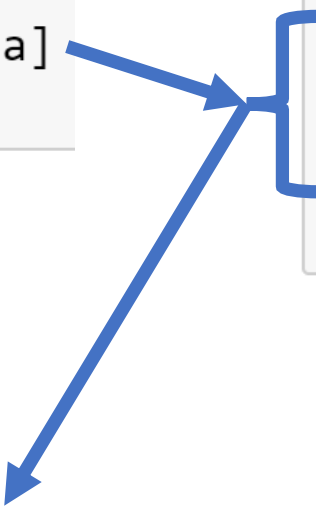
- Sentencias iterativas

```
lista = [2,5,7,6]  
lista2 = [l*2 for l in lista]  
print(lista2)
```

```
[4, 10, 14, 12]
```

```
lista = [2,5,7,6]  
lista2 = []  
for l in lista:  
    lista2.append(l*2)  
print(lista2)
```

```
[4, 10, 14, 12]
```



```
for <variable> in <serie de valores>:  
    instrucción-1  
    ...  
    instrucción-n
```

TEMA 4. ESTRUCTURAS DE CONTROL

Estructuras de control de flujo

- Sentencias iterativas

```
for i in range(1,6):  
    print (i)  
    print ('Hecho')
```

range(a): lista de valores entre 0 y a-1
range(a,b): lista de valores entre a y b-1
range(a,b,n): lista de valores entre a y b-1, con incrementos de n

1
2
3
4
5
Hecho

```
for i in range(3):  
    v = int(input("Introduce un valor:"))  
    print ('Doble:', v*2)
```

Introduce un valor:1
Doble: 2
Introduce un valor:5
Doble: 10
Introduce un valor:8
Doble: 16

```
for i in range(1,6,2):  
    print (i)  
    print ('Hecho')
```

1
3
5
Hecho

TEMA 4. ESTRUCTURAS DE CONTROL

Estructuras de control de flujo

Sumatorio:

```
suma = 0
for i in range(1,6):
    suma = suma + i
print ('Suma:', suma)
```

Suma: 15

```
i: 1 suma: 1
i: 2 suma: 3
i: 3 suma: 6
i: 4 suma: 10
i: 5 suma: 15
```

```
lista = [2,5,7,6]
suma = 0
for l in lista:
    suma = suma + l
print ('Suma:', suma)
```

Suma: 20

```
suma = 0
for i in range(1,6):
    v = int(input("Introduce un valor:"))
    suma = suma + v
print ('Suma:', suma)
```

```
Introduce un valor:3
Introduce un valor:6
Introduce un valor:2
Introduce un valor:8
Introduce un valor:1
Suma: 20
```


TEMA 4. ESTRUCTURAS DE CONTROL

Estructuras de control de flujo

- Sentencias iterativas

```
while <expresión booleana>:  
    instrucción-1  
  
    ...  
  
    instrucción-n
```

```
i=0  
while i<3:  
    print (i)  
    i+=1  
print ('Hecho')
```

```
0  
1  
2  
Hecho
```

TEMA 4. ESTRUCTURAS DE CONTROL

Definición de funciones (avanzado)

- Valores por defecto de los parámetros

```
def imprimir(texto, veces=1):  
    print (veces * texto)  
  
imprimir('Hola')  
imprimir('Hola', 3)
```

```
Hola  
HolaHolaHola
```

- Retorno de varios valores

```
def dividir(a,b):  
    return a/b, a%b  
  
c,r=dividir(10,3)  
print (c, r)  
  
res=dividir(12,4)  
print (res[0], res[1])
```

```
3  1  
3  0
```

TEMA 4. ESTRUCTURAS DE CONTROL

Excepciones

Las excepciones son errores detectados durante la ejecución del programa. Por ejemplo, una división por 0 o intentar acceder a un archivo que no existe.

Si la excepción no se captura, el flujo de ejecución se interrumpe y se muestra información asociada a la excepción en la consola para que el programador pueda solucionar el problema

```
def division(a, b):  
    return a/b
```

```
division(1,0)
```

```
-----  
ZeroDivisionError                                Traceback (most recent call last)  
<ipython-input-19-acfe3b3c369e> in <module>  
      2     return a/b  
      3  
----> 4 division(1,0)  
  
<ipython-input-19-acfe3b3c369e> in division(a, b)  
      1 def division(a, b):  
----> 2     return a/b  
      3  
      4 division(1,0)  
  
ZeroDivisionError: division by zero
```

TEMA 4. ESTRUCTURAS DE CONTROL

Captura de excepciones

`try...except`

- El bloque `try` define el fragmento de código en el que creemos que podría producirse una excepción.
- El bloque `except` permite indicar el tratamiento que se llevará a cabo en el caso de que se produzca la excepción.

```
def division(a, b):  
    try:  
        return a/b  
    except:  
        print('No se puede realizar la division de', a, 'entre', b)  
  
division(1,0)
```

No se puede realizar la division de 1 entre 0

TEMA 4. ESTRUCTURAS DE CONTROL

Captura de excepciones

- Python permite utilizar varios bloques except para un solo bloque try, de forma que podemos dar un tratamiento distinto a la excepción dependiendo del tipo de excepción de la que se trate.

```
def division(a, b):  
    try:  
        return a/b  
    except ZeroDivisionError:  
        print('Division por 0')  
    except TypeError:  
        print('Uno de los operandos no es un numero')  
    except:  
        print('No se puede realizar la division de', a, 'entre', b)  
  
division(1, 0)  
division('hola', 8)
```

Division por 0

Uno de los operandos no es un numero

TEMA 4. ESTRUCTURAS DE CONTROL

Captura de excepciones

- También se puede utilizar un mismo `except` para tratar más de una excepción.
- La cláusula opcional `finally` se ejecuta siempre, se produzca o no una excepción.

```
def division(a, b):  
    try:  
        return a/b  
    except (ZeroDivisionError, TypeError):  
        print('Error')  
    finally:  
        print('Fin de la funcion')  
  
division(1,0)
```

Error

Fin de la funcion

Como programadores también podemos crear nuestras propias excepciones.

Pero esto queda fuera del alcance de este curso.

TEMA 4. ESTRUCTURAS DE CONTROL

Ejemplo: ecuación de segundo grado

```
print("Programa para la resolución de la ecuación ax+b=0")
a = float(input("Valor de a:"))
b = float(input("Valor de b:"))

try:
    x = -b/a
    print("Solución: ", x)

except ZeroDivisionError:
    if b!=0:
        print("La ecuación no tiene solución")
    else:
        print("La ecuación tiene infinitas soluciones")
```

TEMA 4. ESTRUCTURAS DE CONTROL

(Algunos) tipos de excepciones

Tipo	Significado
FloatingPointError	Error en una operación de coma flotante
OverflowError	Resultado demasiado grande para poder representarse
ZeroDivisionError	División por 0
EOFError	Se intentó leer más allá del final de fichero
IOError	Error en una operación de entrada/salida
IndexError	El índice de la secuencia está fuera del rango posible
KeyError	La clave no existe (diccionarios)
NameError	No se encontró ningún elemento con ese nombre
TypeError	Tipo de argumento no apropiado
ValueError	Valor del argumento no apropiado