Programando en Python II

GOBIERNO DE ESPAÑA
VICEPRESIDENCIA PRIMERA DEL GOBIERNO
MINISTERIO DE ASUNTOS ECONÓMICOS Y TRANSFORMACIÓN DIGITAL
SECRETARÍA DE ESTADO DE DIGITALIZACIÓN E INTELIGENCIA ARTIFICIAL

red.es

Centro de Referencia Nacional en Comercio Electrónico y Marketing CRN Digital

GARANTÍA JUVENIL

UNIÓN EUROPEA

Barrabés

The Valley

"El FSE invierte en tu futuro"

Fondo Social Europeo

1. Estructuras de control
2. Sentencias condicionales
   2.1. if
   2.2. else
   2.3. elif
3. Sentencias iterativas
   3.1. while
   3.2. for
   3.3. break

Estructuras de control

Estructuras de control

Flujo de un programa

Un programa se forma con una serie de líneas de código que se ejecutan una tras otra siguiendo el orden en el que aparecen: el flujo del programa es secuencial.

Sin embargo, es posible alterar dicho flujo.

Los programas son capaces de tomar decisiones en función de datos o resultados intermedios, y en función de éstos,

1. **Ejecutar ciertas sentencias y otras no.**
2. **Ejecutar ciertas sentencias más de una vez.**

Sentencias condicionales

Estructuras de control

Sentencias condicionales: *if*

<<Al llegar a este punto, ejecuta esta(s) acción(es) sólo si esta condición es cierta.>>

```
1 if condición:
2     acción
3     acción
4     ...
5     acción
```

Estructuras de control

Sentencias condicionales: *if*

primer_grado.py

```
print('Programa para la resolución de la ecuación a x + b = 0.')

a = float(input('Valor de a: '))
b = float(input('Valor de b: '))

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

if a == 0:
    if b != 0:
        print('La ecuación no tiene solución.')
    if b == 0:
        print('La ecuación tiene infinitas soluciones.')
```

Estructuras de control

Sentencias condicionales: *else*

<<Sino, ejecuta estas otras acciones>>

```
1 if condición:
2     acciones
3 else:
4     otras acciones
```

Estructuras de control

Sentencias condicionales: *else*

```
1  from math import sqrt  # La función sqrt calcula la raíz cuadrada de un número.
2
3  print('Programa para la resolución de la ecuación a x*x + b x + c= 0.')
4
5  a = float(input('Valor de a: '))
6  b = float(input('Valor de b: '))
7  c = float(input('Valor de c: '))
8
9  if a != 0:
10     x1 = (-b + sqrt(b**2 - 4*a*c)) / (2 * a)
11     x2 = (-b - sqrt(b**2 - 4*a*c)) / (2 * a)
12     print('Soluciones: x1={0:.3f} y x2={1:.3f}'.format(x1, x2))
13 else:
14     if b != 0:
15         x = -c / b
16         print('Solución: x={0:.3f}'.format(x))
17     else:
18         if c != 0:
19             print('La ecuación no tiene solución.')
20         else:
21             print('La ecuación tiene infinitas soluciones.')
```

Estructuras de control

Sentencias condicionales: *else*

```
1  mes = int(input('Dame un mes: '))
2
3  if 1 <= mes <= 3:
4      print('Invierno.')
5  else:
6      if mes == 4  or  mes == 5  or  mes == 6:
7          print('Primavera.')
8      else:
9          if not (mes < 7  or  9 < mes):
10             print('Verano.')
11         else:
12             if not (mes != 10  and  mes != 11  and  mes != 12):
13                 print('Otoño.')
14             else:
15                 print('Ningún año tiene {0} meses.'.format(mes))
```

Estructuras de control

Sentencias condicionales: *elif*

```
1 if condición:
2     ...
3 elif otra condición:
4     ...
```

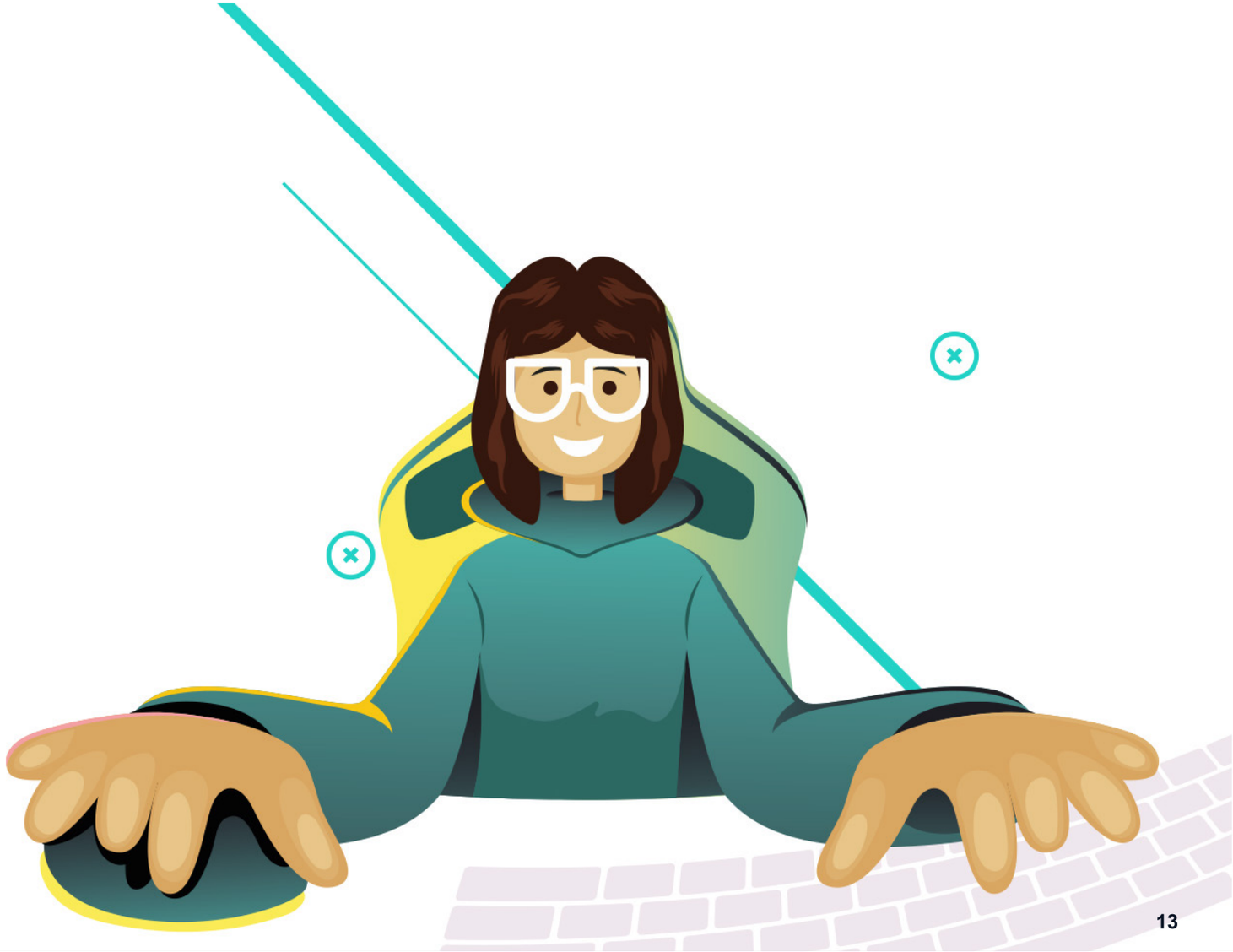
Estructuras de control

Sentencias condicionales: *elif*

```
from math import pi

radio = float(input('Dame el radio de un círculo: '))

# Menú
print('Escoge una opción: ')
print('a) Calcular el diámetro.')
print('b) Calcular el perímetro.')
print('c) Calcular el área.')
opción = input('Teclea a, b o c y pulsa el retorno de carro: ')

if opción == 'a':  # Cálculo del diámetro.
    diámetro = 2 * radio
    print('El diámetro es {0}.'.format(diámetro))
elif opción == 'b': # Cálculo del perímetro.
    perímetro = 2 * pi * radio
    print('El perímetro es {0}.'.format(perímetro))
elif opción == 'c': # Cálculo del área.
    área = pi * radio ** 2
    print('El área es {0}.'.format(área))
else:
    print('Solo hay tres opciones: a, b o c.')
    print('Tú has tecleado "{0}".'.format(opción))
```

Ejercicios

- Notebook 2. Parte 1



Sentencias iterativas

Estructuras de control

Bucles: while/for

Ejecutar un fragmento de código más de una vez.

Estructuras de control

Sentencias iterativas: *while*

<<Mientras se cumpla esta condición, repite estas acciones>>

```
1 while condición:
2     acción
3     acción
4     ...
5     acción
```

Estructuras de control

Sentencias iterativas: *while*

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

Estructuras de control

Sentencias iterativas: *while*

Bucles sin fin: CUIDADO! nunca acaba la ejecución del programa.

```
1 i = 0
2 while i < 10:
3     print(i)
```

Estructuras de control

Sentencias iterativas: *for*

<<Para todo elemento de una serie, hacer…>>

```
1 for variable in serie de valores:
2     acción
3     acción
4     ...
5     acción
```

Estructuras de control

Sentencias iterativas: *for*

```
1 número = int(input('Dame un número: '))
2
3 for potencia in [2, 3, 4, 5]:
4     print('{0} elevado a {1} es {2}'.format(número, potencia, número ** potencia))
```

Estructuras de control

Generar secuencias de valores: range

```
>>> list(range(2, 10))↵
[2, 3, 4, 5, 6, 7, 8, 9]
>>> list(range(0, 3))↵
[0, 1, 2]
>>> list(range(-3, 3))↵
[-3, -2, -1, 0, 1, 2]
>>> list(range(-10, -1))↵
[-10, -9, -8, -7, -6, -5, -4, -3, -2]
```

```
>>> list(range(5))↵
[0, 1, 2, 3, 4]
```

```
>>> list(range(10, 5, -1))↵
[10, 9, 8, 7, 6]
>>> list(range(3, -1, -1))↵
[3, 2, 1, 0]
>>> list(range(10, 1, -3))↵
[10, 7, 4]
```

Estructuras de control

Roturas de bucles: *break*

Abortar la ejecución de un bucle desde cualquier punto del mismo.

```
1 creo_que_se_cumple_para_alguno = False
2 for elemento in conjunto:
3     if condición:
4         creo_que_se_cumple_para_alguno = True
5         break
6
7 if creo_que_se_cumple_para_alguno:
8     print('Se cumple para alguno')
```

Estructuras de control

Bucles anidados

```
1 for i in range(0, 5):
2     for j in range(0, 3):
3         print(i, j)
```

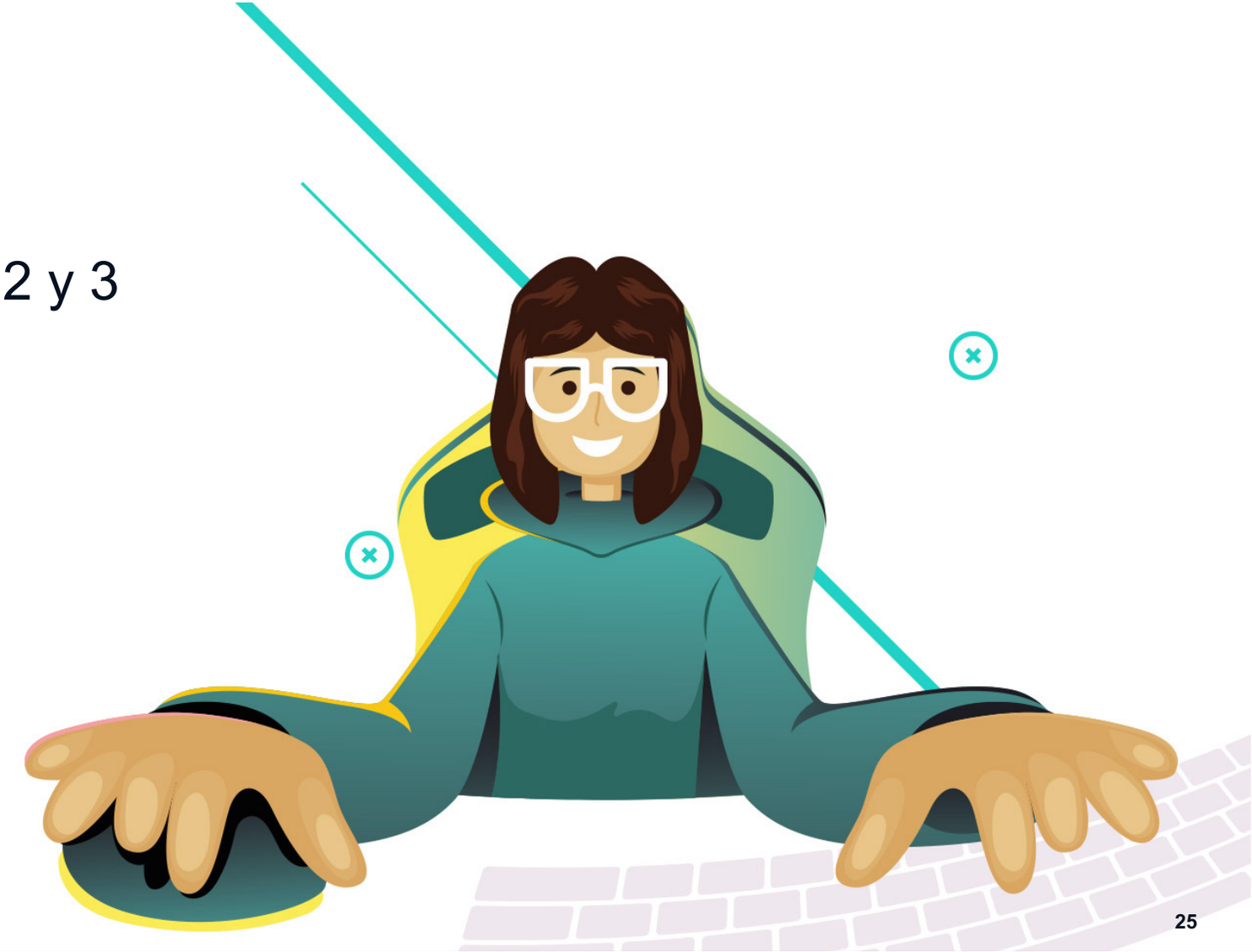
Más

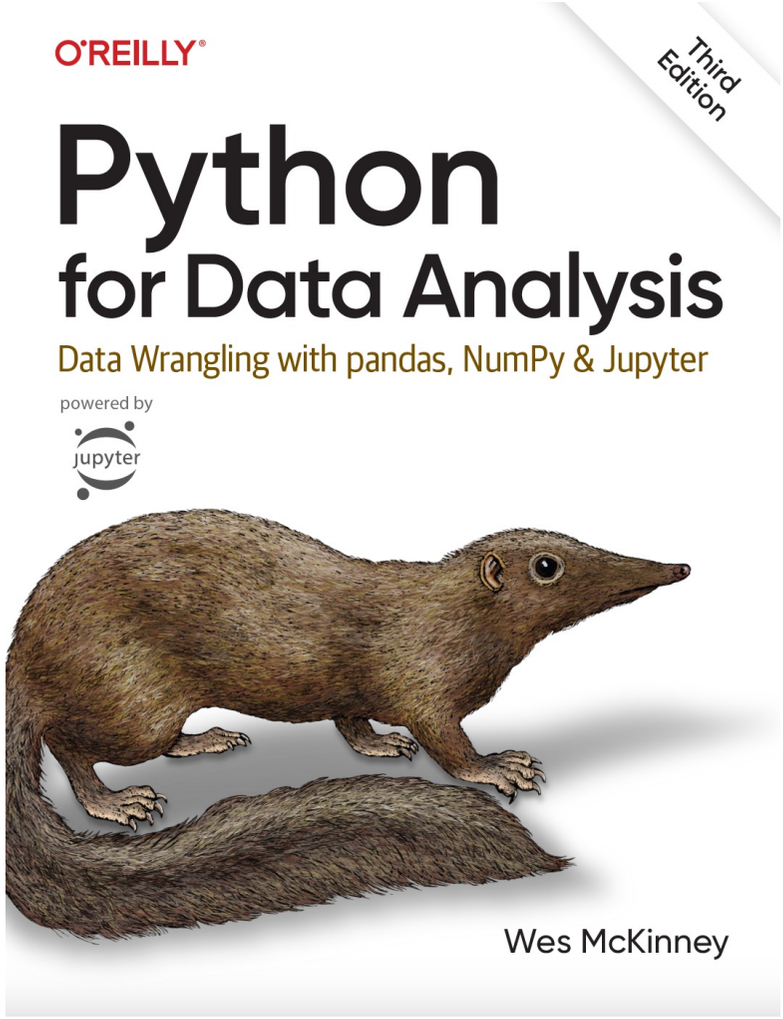
Solución

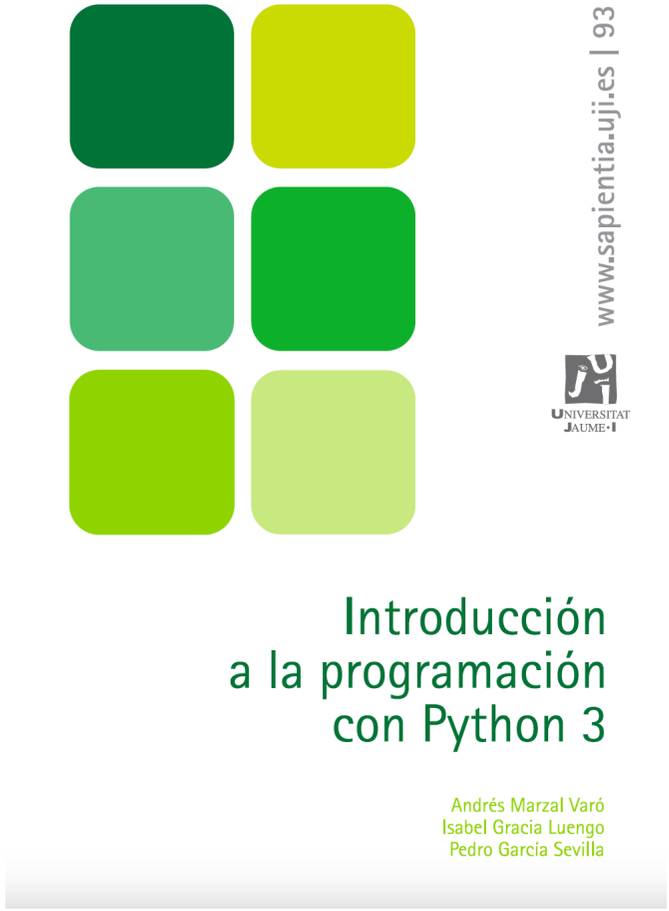
Cada problema de programación puede tener más de una solución. Es decir, podemos escribir un programa que realice lo mismo de muchas maneras.

Ejercicios

- Notebook 2. Parte 2 y 3




O'REILLY
Third Edition
Python
for Data Analysis
Data Wrangling with pandas, NumPy & Jupyter
powered by
jupyter
Wes McKinney


www.sapientia.uji.es | 93
Universitat Jaume I
Introducción
a la programación
con Python 3
Andrés Marzal Varó
Isabel Gracia Luengo
Pedro García Sevilla

Contacto

Correo: a.cobo.aguilera@gmail.com

LinkedIn: Aurora Cobo Aguilera
GitHub: AuroraCoboAguilera
Google Scholar: Aurora Cobo Aguilera

red.es

Centro de Referencia Nacional en Comercio Electrónico y Marketing
CRN Digital

GARANTÍA JUVENIL

"El FSE invierte en tu futuro"

Fondo Social Europeo

Barrabés

The Valley