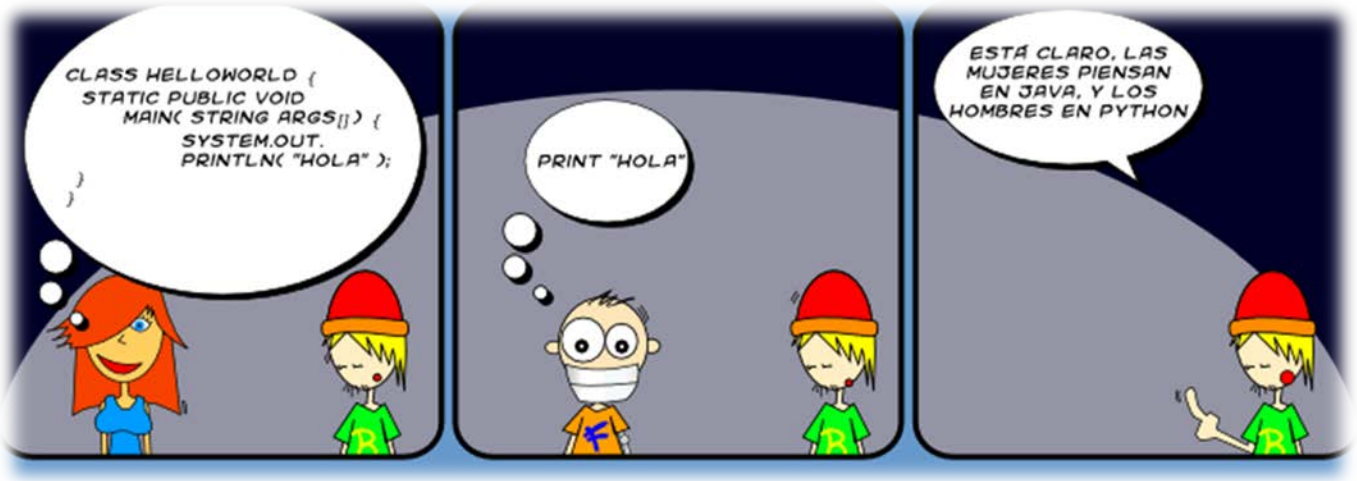
COMENÇANT A PROGRAMAR

TIPUS DE VARIABLES I OPERADORS

Mª Belén Tortosa Pedrón

INTRODUCCIÓ

- Python és un llenguatge que pot servir per desenvolupar fàcilment tant petites aplicacions com aplicacions de grans dimensions. Principals carcaterístiques de Python:
  - **Alt nivell** - Sintaxis senzilla, per tant, és ideal per aprendre a programar per primera vegada. A la vegada, facilita el desenvolupament d'aplicaciones, ja que acorta el número de línias de codi a desenvolupar amb respecte altres llenguatges de programació.
  - **Multipropòsit** - Pot ser utilitzat per desenvolupar tant scripts senzills com per desenvolupar llocs webs dinàmics.
  - **Té una gran llibreria pre-instalada de recolçament** - Pot ser que a dintre de llibreries ja estiguin desenvolupades moltes de les coses comuns que vols fer, així evitem programar un mòdul des de zero.

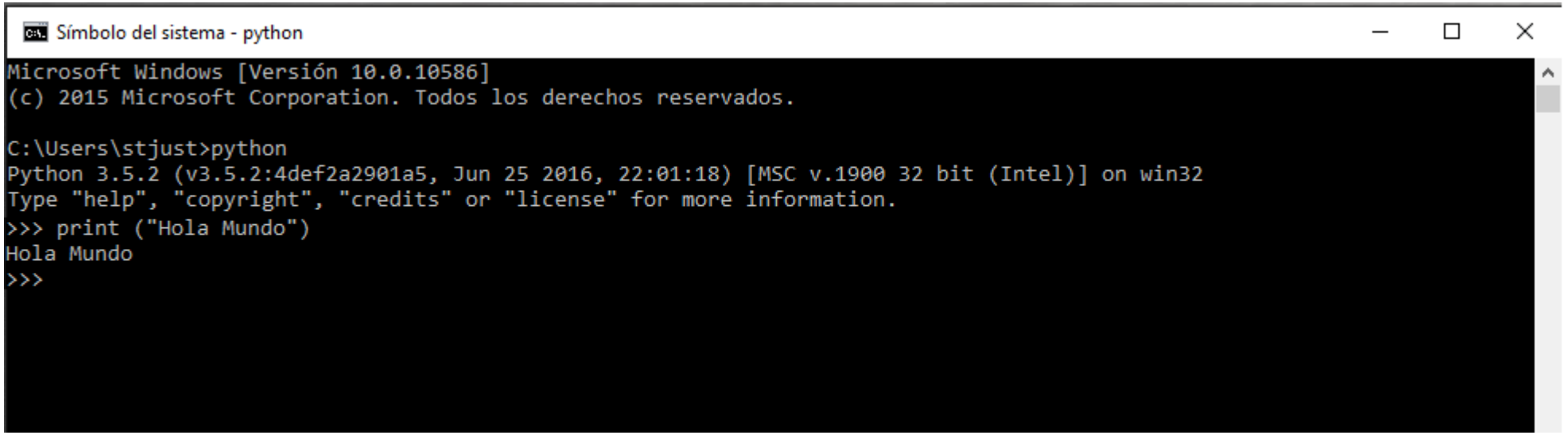


VERSIONS

- Existeixen diferents versions actives de Python de les quals podem descarregar:
- **Python 2**
  - La més compatible, ja que com ha estat més temps en el mercat existeixen una gran de quantitat de llibreries que poden ser utilitzades amb aquesta versió.
- **Python 3**
  - Funcionalitats millorades amb respecte la versió anterior però és pràcticament incompatible amb Python 2.
  - És important ressaltar que escollir la versió és un pas important ja que migrar de Python 2 a Python 3 no és tan fàcil, degut a que utilitzen sintaxis de desenvolupament diferents.
- Nosaltres utilitzarem Python 3.

COM EXECUTAR PYTHON

- Existeixen dos formes d'executar codi en Python:
  - Interactivament: Línia a línia amb l'intèrpret.



  - De forma habitual, escrivint un arxiu de codi font i executant-lo indicant el nom de l'arxiu a executar a l'intèrpret de Python.

COMENTARIS

- **Comentaris d'una línia**: Són línies de codi que Python ignora completament i no executa. La seva funció es explicar les parts més importants del codi.
- Una línia serà ignorada completament quan comença pel caràcter **#**

Línies que Python no tindrà en conte.

1_exemple

```
# Això és una cadena
c= "HOLA MON"
# i això es un enter
e=23
#Es pot comprovar amb la funció type

print(type(c))
print(type(e))
```

COMENTARIS

- **Comentaris de varies línies**: Quan volem incloure aclariments de vàries línies o ometre l'execució d'una part del codi, utilitzarem la terminologia: «"» (triple cometes dobles) per indicar a on comença el comentari i «"» per indicar a on acaba el comentari.
- Les línies que estiguin en mig, seran ignorades per l'intèrpret de Python.

Línies que Python no tindrà en conte.

```
2  import math
3  print('El valor de PI es aproximadament: ',math.pi)
4
5  """
6  Aquestes línies de codi
7  NO s'executen
8
9  """
```

TIPUS BÀSICS DE DADES

- Es divideixen en:
  - Números:
    - Enters: ***8***, ***1478***, ***-9852*** ,....
    - Decimals: ***15.58***, ***-69.25*** ,....
    - Complexes: ***7+5 j***
  - Cadenes de text: "***HOLA MON***"
  - Valors booleans: ***True*** i ***False***
- Per crear una variable:

1_exemple

```
# Això és una cadena
c= "HOLA MON"
# i això es un enter
e=23
#Es pot comprovar amb la funció type

print(type(c))
print(type(e))
```

type: *Funció en Python que ens indica de quin tipus és la variable*

ENTERS

- En Python es representen mitjançant la paraula ***int*** o ***long***. El tipus ***long*** permet emmagatzemar números més grans.
- El rang dels valors que poden representar depèn de la plataforma. En la major part de les màquines el long de C s'emmagatzema utilitzant 32 bits, es a dir, amb una variable de tipus ***int*** en Python podem emmagatzemar del número: ***-2.147.483.648*** a ***2.147.483.647***.

  En plataformes de 64 bits, el rang és de ***-9.223.372.036.854.775.808*** fins a ***9.223.372.036.854.775.807***.

  El tipus ***long*** de Python permet emmagatzemar números de qualsevol precisió, limitat per la memòria disponible en la màquina.
- Si assignem un número a una variable, aquesta passarà a ser de tipus ***int***, tret que el número sigui tan gran com per a requerir l'ús del tipus ***long***.
- En Python 3 desapareix el tipus ***long*** i tot passa a ser ***int***.

DECIMALS

- Els números reals són els que tenen decimals.
- En Python s'expressen mitjançant el tipus ***float***. Python, implementa el tipus ***float*** a baix nivell utilitzant una variable de tipus ***double*** de C, es a dir, utilitzant 64 bits, per tant en Python sempre s'utilitza doble precisió, això significa que el rang de valors que poden representar són des de ***±2,2250738585072020 x 10-308*** fins a ***±1,79769313486231570×10308***.

Variable decimal

```
3_exemple
1  real= 14.35
2
3  decimal= 0.2e-3
4
5  print(real)
6  print(decimal)
7
```

```
14.35
0.0002
[Finished in 0.2s]
```

```
Line 8, Column 1
```

Variable decimal en notació científica

COMPLEXES

- Es representa amb la paraula ***complex*** en Python, s'emmagatzema usant coma flotant, són una extensió dels números reals.
- En concret s'emmagatzemen en una estructura de C, composta per dues variables de tipus ***double***, sent una d'elles per a emmagatzemar la part reial i l'altra per a la part imaginaria.
- Els números complexes en Python es representen de la següent forma:

```
3_exemple
1  c=7+5j
2  print(c)
```

EL VALOR NONE

- Existeix un valor especial en Python per a representar aquells casos en els que “no hi ha valor”; és el valor **None**.
- Aquest valor es l’únic del seu tipus. S’interpreta com a fals en les expressions lògiques i és el valor retornat per les funcions que no retornen cap valor explícitament.
- Quan el resultat d’una expressió és **None**, l'intèrpret no el mostra.

```
>>> None
>>>
```

CADENES

- Les cadenes són texte tancat entre cometes simples (‘cadena’) o cometes dobles (“cadena”).
- A dintre de les cometes es poden posar caràcters especials escapant-los amb ‘\’, como ‘\n’ (caràcter nova línia), o ‘\t’, el de tabulació.

```
3_exemple | 4_exemple_cadena
1  cadena="HOLA\nMON\n"
2  print(cadena)
3

HOLA
MON

[Finished in 0.2s]
```

- Les cadenes admeten operadors com la suma:

```
4_exemple_cadena
1   a = "uno"
2   b = "dos"
3
4   # c es "unodos"
5   c = a + b
6   print(c)
7
8   # c es "unounouno"
9   c = a * 3
10  print(c)
11

unodos
unounouno
[Finished in 0.2s]
```

EXEMPLE CADENES

4_exemple_cadena

```
1  # Comillas simples
2  cadenaa = 'Texto entre comillas simples'
3  print ( cadenaa)
4  print(type(cadenaa))
5  # Comillas dobles
6  cadenab="Texto entre comillas dobles"
7  print (cadenab)
8  print (type(cadenab))
9  # Cadena con codigo escapes
10
11 cadenaesc = 'Texto entre \n\tcomillas simples'
12 print (cadenaesc)
13 print (type(cadenaesc))
14 # Cadena multilinea
15 cadenac = """Texto linea 1
16 linea 2
17 linea 3
18 linea 4
19 """
20 print (cadenac)
21 print (type (cadenac))
22 # Repeticion de cadena
23 cadrep = "Cadena" * 3
24 print (cadrep)
25 print (type (cadrep))
26
27 # Concatenacion de cadena
28 nombre = "Leonardo"
29 apellido = "Caballero"
30 nombre_completo = nombre + " " + apellido
31 print (nombre_completo)
32 print (type (nombre_completo))
33
34 print ("Tamano de cadena '", nombre_completo, "' es:", len(nombre_completo))
35
36 # acceder a rango de la cadena
37 print (nombre_completo[3:13])
38
```

Execució

```
Texto entre comillas simples
<class 'str'>
Texto entre comillas dobles
<class 'str'>
Texto entre
    comillas simples
<class 'str'>
Texto linea 1
linea 2
linea 3
linea 4

<class 'str'>
CadenaCadenaCadena
<class 'str'>
Leonardo Caballero
<class 'str'>
Tamano de cadena ' Leonardo Caballero ' es: 18
nardo Caba
[Finished in 0.2s]
```

BOOLEANS

- Poden tenir només dues valors True o False. S'utilitzen en les expressions condicionals i en els bucles.

4_exemple_bool

```
1  # -*- coding: utf8 -*-
2
3  print ('\nTipos de datos booleanos' )
4  print ('========================\n')
5  aT= True
6  # Tipos de datos booleanos
7  print ("El valor es Verdadero:", aT, ", el cual es de tipo", type(aT), "\n")
8
9  aF = False
10 print( "El valor es Falso:", aF, ", el cual es de tipo", type(aF))
11
12 print( '\nOperadores booleanos')
13 print ('====================\n')
14
15 # Operadores booleanos
16 aAnd = True and False
17 print ("SI es Verdadero Y Falso, entonces es:", aAnd, ", el cual es de tipo", type(aAnd), "\n")
18
19 aOr = True or False
20 print( "SI es Verdadero O Falso, entonces es:", aOr, ", el cual es de tipo", type(aOr), "\n")
21
22 aNot = not True
23 print ("Si NO es Verdadero, entonces es:", aNot, ", el cual es de tipo", type(aNot), "\n")
24
```

```
Tipos de datos booleanos
========================

El valor es Verdadero: True , el cual es de tipo <class 'bool'>

El valor es Falso: False , el cual es de tipo <class 'bool'>

Operadores booleanos
====================

SI es Verdadero Y Falso, entonces es: False , el cual es de tipo <class 'bool'>

SI es Verdadero O Falso, entonces es: True , el cual es de tipo <class 'bool'>

Si NO es Verdadero, entonces es: False , el cual es de tipo <class 'bool'>

[Finished in 0.2s]
```

OPERADORS ARITMÈTICS

OPERADOR	DESCRIPCIÓ	EXEMPLE
+	Suma	r = 3+2 # r és 5
-	Resta	r = 30-20 # r és 10
*	Producte	r = 30 * 20 # r és 600
**	Exponent	r = 3**3 # r és 9
/	Divisió	r = 3.5/2 # r és 1.75
//	Divisió entera	r = 3.5//2 # r és 1.0
%	Mòdul: Residu divisió entera	r = 7%2 # r és 1

NOTA (en Python v2): Quan fem una divisió hem de tenir en conte que si utilitzem ***dos operadors enters***, Python determina que volen que la variable ***resultat també sigui un enter***, per tant el resultat de, per exemple, ***3 / 2*** i ***3 // 2*** seria el mateix: ***1***

Si volem obtenir els decimals necessitarem que al menys un dels operands sigui un número decimal. Per això, tenim dos opcions:

1. **r = 3.0 / 2**
2. o be utilitzant la funció ***float***: **r = float(3) / 2**

Quan es barregen tipus diferents de números, Python converteix tots els operands al tipus més complex.

OPERADORS LÒGICS

Serveixen per establir condicions i poden permetre que més d'una condició es compleixi a la vegada:

OPERADOR	DESCRIPCIÓ	EXEMPLE
and	Es compleix a i b?	r= True and False # r és False
or	Es compleix a ó b?	r= True or False # r és True
not	No a	r= not True # r és False

OPERADOR	DESCRIPCIÓ	EXEMPLE
==	Són iguals a i b?	r = 5 ==3 # r és False
!=	Són diferents a i b?	r = 5 !=3 # r és True
<	a és menor que b?	r = 5<3 # r és False
>	a és major que b?	r = 5>3 # r és True
<=	a és menor o igual que b?	r = 5<=3 # r és False
>	a és major o igual que b?	r = 5>=3 # r és True

OPERADORS D'ASSIGNACIÓ

- Els operadors d'assignació s'utilitzen per assignar un valor a una variable, així com quan utilitzem el "**=**".
- Els operadors d'assignació són "=,+=,-=,*=,/=,**=,//=", a continuació, posem alguns exemples:
- = , igual a, és el més simple de tots i assigna a la variable del costat esquerre qualsevol variable o resultat del costat dret.
- += , suma a la variable del lado izquierdo el valor del lado derecho.
  ej. si "a" es igual a 5 y a+=10, entonces "a" sera igual a 15
- -= , resta a la variable del lado izquierdo el valor del lado derecho.
  ej. si "a" es igual a 5 y a-=10, entonces "a" sera igual a -5
- *=, multiplica a la variable del lado izquierdo el valor del lado derecho.
  ej. si "a" es igual a 5 y a*=10, entonces "a" sera igual a 50
- Espero que hasta el momento hayas podido encontrar este tutorial de ayuda, espero tus comentarios.