2.0 Data Types Theory

November 11, 2022

0.1 1. Strings

Hay infinidad de funciones útiles sobre cadenas de caracteres que se usan con mucha frecuencia:

[ ]:
```
cadena = "En un lugar de la Mancha"
len(cadena)
```

[ ]: 24

[ ]:
```
cadena.lower()
```

[ ]: 'en un lugar de la mancha'

[ ]:
```
cadena.upper()
```

[ ]: 'EN UN LUGAR DE LA MANCHA'

[ ]:
```
cadena = "    En un lugar de la Mancha     "
cadena.strip()
```

[ ]: 'En un lugar de la Mancha'

[ ]:
```
cadena = "&&&En un lugar de la Mancha&"
cadena.strip("&")
```

[ ]: 'En un lugar de la Mancha'

[ ]:
```
cadena.count("a")
```

[ ]: 4

[ ]:
```
cadena = "En un lugar de la Mancha"
cadena.find("lugar")
```

[ ]: 6

[ ]:
```
cadena[cadena.find("lugar"):len(cadena)]
```

[ ]: 'lugar de la Mancha'

[ ]:
```
cadena.split(" ")
```

[ ]:
```
['En', 'un', 'lugar', 'de', 'la', 'Mancha']
```

Replicación

Además de concatenar con el signo '+', se puede replicar con el '*':

[ ]:
```
nombre = "Guille"
replicado = nombre*5
replicado
```

[ ]:
```
'GuilleGuilleGuilleGuilleGuille'
```

Casting

int(cadena)

float(cadena)

string([cualquier número por ejemplo integer o float])

Placeholders

Por último, los placeholders se colocan con las llaves y permiten formatear texto:

[ ]:
```
frase = "En un {} de {} de cuyo nombre no quiero acordarme"
formateada = frase.format('garito','La Mancha')
formateada
```

[ ]:
```
'En un garito de La Mancha de cuyo nombre no quiero acordarme'
```

0.2 2. Integers

Los enteros (tipo *int*) se declaran tal cual (sin comillas como las cadenas) y sin puntos decimales que los convetirían en float. En el caso de que se manejen números muy grandes, para hacerlos más legibles al ojo humanos se puede usar el guión bajo a modo de "separador de miles".

[ ]:
```
a = 156_589_698_547
a
```

[ ]:
```
156589698547
```

0.3 3. Float

Representa a los números decimales en coma flotante. La función *round(float,i)* redondea el float que pasemos como parámetro a 'i' dígitos

[ ]:
```
round(3.1415928,4)
```

[ ]:
```
3.1416
```

0.4 4. Boolean

Se usan muchísimo. Dos valores posibles con la primera letra en mayúscula:

True

False

Nota: la función ***type*** permite saber el tipo de datos de cualquier variable.

Los operadores lógicos son:

Símbolo	Operador lógico
and	True si las dos expresiones booleanas son True
or	True si una de las dos expresiones es True
not	Invierte la expresión booleana a la que acompaña

Y los operadores de comparación:

Símbolo	Operador lógico
==	Igual
!=	Distinto
>	Mayor que
<	Menor que
>=	Mayor o igual
<=	Menor o igual

Es útil cuando haya que evaluar muchas expresiones, dibujar el diagrama de flujo con www.draw.io

0.5 5. Operaciones matemáticas

Las básicas son:

Símbolo	Operación
+	Suma
-	Resta
*	Multiplicación
/	División
**	Potencia
//	División entera
%	Resto de la división entera

Prioridad –> PEMDAS-LR

Parentheses

Exponents

Multiplication

Division

Adition

Substraction

Left to **R**ight

0.6 6. f-Strings

Son muy útiles para combinar strings con otros tipos de datos y ahorrarnos algunos castings. Por ejemplo:

score = 0 height = 1.8 print("El jugador de" + str(height) + " metros de altura lleva " + str(score) + " puntos") print(f"El jugador de {height} metros de altura lleva {score} puntos")