PYTHON PER A LINGÜISTES

CONCEPTES BÀSICS DE PYTHON: ESTRUCTURES DE DECISIÓ, BUCLES, FUNCIONS 2

Profe: Alex Peiró Lilja

Servei de Tecnologia Lingüística (STeL)

Universitat de Barcelona, 2023

RESUM: COMPUTACIÓ?

- Computació = Dades + Algorismes
- Dades: "Ingredients del pastís" (INPUTS), "El pastís llest per servir" (OUTPUT)
- Algorisme: un procés basat en instruccions específiques, "Preparar el pastís"
- Algorisme té, normalment, dades d'entrada, i/o de sortida
- Algorisme pot tenir diversos passos intermedis
- Per tal de supervisar els canvis, fem servir les variables

RESUM: TIPUS DE DADES?

- Diferents tipus, cadascun amb les seves respectives operacions
- Tipus numètic: enters (1, 4, 19, 26); flotants (1.5, 0.5, 6.667)
- Tipus boolean: *True, False*
- Tipus string: “u”, “un”, “un string”, “un string &/$·%=“
- Estructura de dades tipus llista: [1, “hola”, 5.5, 23, “món”, False]

QUÈ SABEM SOBRE TREBALLAR AMB LES DADES?

- Hem après les operacions bàsiques de diferents tipus de dades
- El tipus de dades amb les que operarem delimita les operacions que podem realitzar:
  - Tipus numèrics es poden modificar utilitzant operacions aritmètiques
  - Els strings i les llistes poden ser concatenats o segmentats

QUÈ SABEM SOBRE LES CONDICIONS?

- Controlen el "fluxe" d'operacions del programa. Ens permeten afegir "preguntes" les quals el programa haurà de comprovar i prendre una decisió basada en aquesta condició
- Condició: "La nova contrasenya és la mateixa a l'antiga?"
- Decisió: "Si us plau, tria una contrasenya nova" o bé "Contrasenya canviada"
- Les condicions es resolen amb un boolean en TRUE o FALSE

QUÈ SABEM SOBRE ELS BUCLES?

- Els bucles ens permeten realitzar un bloc d'accions repetidament fins un cert límit
- El bucle "while" és similar a l'estructura de decisió → sempre que la condició donada sigui TRUE, el programa seguirà executant les accions associades a la decisió presa
- La condició del "while" ha de ser avaluada amb un boolean
- En un cert moment, la condició del "while" deixarà de complir-se (FALSE) degut a alguna modificació dins el seu bloc d'accions. Llavors, el programa sortirà del bucle

RESUM SOBRE LES FUNCIONS

- Són un element de programació que permeten agrupar algorismes i reutilitzar-los en qualsevol moment (similar a les variables, que ens permeten etiquetar i emmagatzemar valors que necessitem consultar o modificar més endavant)
- Les funcions poden requerir de 0 o més arguments d'entrada (de qualsevol tipus)
- Les funcions poden o bé realitzar accions *(print())* o retornar informació processada dins seu *(len())*
- Tenim funcions natives de Python o de llibreries externes. Però també podem crear-les nosaltres segons les nostres necessitats

PRACTICAR TOT ALLÒ APRÈS

- L'objectiu d'avui serà repassar i seguir practicant tot el que s'ha donat fins ara
- Temps per seguir treballant amb els exercicis i resoldre qualsevol dubte
- Veurem el següent tipus de bucle, el *for*

BUCLE *FOR*

- El bucle ***for*** manté el mateix objectiu que el *while*: executar el mateix bloc de codi múltiples vegades
- Recordem: el bucle *while* comprova una condició donada per seguir o sortir del bucle
- En canvi, el bucle ***for*** executa el bloc de codi un número definit de vegades
- El bucle ***for*** requereix d'un element d'estructura seqüencial (e.g, una llista)amb una longitud limitada el qual recórrer

BUCLE *FOR*

- El bucle ***for*** en Python té la següent sintaxi:

```
for paraula in [“Això”, “és”, “una”, “frase”]:
    print(paraula)
```

(La llista conté 4 strings. El loop ***for*** assignarà de manera ordenada i consecutiva cadascun dels string a la variable “paraula”. A cada iteració la variable “paraula” serà diferent)

- “paraula” és una variable. Es pot nombrar de qualsevol manera que ens convingui segons el que estigui emmagatzemant.
- A cada iteració del bucle, la variable “paraula” contindrà un valor diferent de la seqüència que està recorrent

BUCLE *FOR*

- Destacar que el bucle ***for*** recorre una estructura seqüencial. Hem posat d'exemple una llista, però recordeu que un *string* també és una seqüència de caràcters:

```
for c in "banana":
    print(c)
```

- Què creieu que sortirà per pantalla?

BUCLE *FOR*

- Una funció molt útil a combinar amb l'estructura de bucle ***for*** és la funció *range()*
- Retorna una seqüència de números consecutius els quals podem fer servir com a índex d'una llista o *string* per exemple:

```
for n in range(5):
    print(n)
```

```
for n in range(len("banana")):
    print(n)
```

- Què creieu que sortirà per pantalla?

DECLARACIONS EN BUCLES

- És possible forçar als bucles a finalitzar o saltar d'iteració. Per això, tenim dos declaracions: ***break*** i ***continue***.
- ***break***: surt del bucle immediatament un cop arriba a aquesta línia de codi dins del bucle. Generalment es declara sota una condició específica dins el propi bucle
- ***continue:*** si el codi arriba a aquesta declaració dins del bucle, el codi no sortirà del bucle, sinó que saltarà a la següent iteració. En el cas del *while* torna a comprovar la condició del bucle. En el cas del *for* salta al següent element de la seqüència.

DECLARACIONS EN BUCLES

```
fruits = [“plàtan”, “poma”, “prèssec”]
for x in fruits:
        print(x)
        if x == “poma”:
            break
```

```
i = 0
while i < 6:
        i += 1
        if i == 3:
            continue
        print(i)
```

DECLARACIONS EN CONDICIONS

- També existeixen declaracions per a les condicions. Tenim el ***pass*** que podem afegir quan volem definir una condició la qual el programa no hagi de fer res. Les condicions no poden quedar buides de codi, així que aquesta declaració permet deixar continuar el programa.

```
a = 50
b = 200
if b > a:
        pass
else:
        print(a)
```

GRÀCIES!

PREGUNTES?