Introducció a la Programació
Grau en Estadística
Universitat Autònoma de Barcelona

Sessió 6: Variables String, Fitxers i Tractament d'errors

Vicenç Soler

Índex

1. Introducció

En aquesta sessió es veurà com treballar amb Srings, fitxers de text i veurem com es tracten els errors per a evitar que un programa acabi de manera inesperada.

2. String

Fins ara, hem treballat amb variables numèriques i booleanes. Un altre tipus que també hem fet servir i que ara expliquem són els String.

Aquest tipus de dada guarda un text. Els Strings es declaren entre cometes dobles "..." o cometes simples '...', indistintament. Ambdós tenen el mateix significat i pots fer servir les que et vinguin més de gust. Fins ara, he fet anar les cometes dobles "..." . Exemple:

```
a="Hola"   #o també podria haver estat 'Hola'
```

La variable 'a' és un string.

Es poden definir strings llargs que ocupin més d'una línia, fent servir les 3 cometes dobles a l'inici i al final (com quan fem comentaris multilínia):

Entre cometes dobles:	També podria ser entre tres cometes simples:
a = """Hola bon dia."""	a = '''Hola bon dia.'''

Nota: Compte, has de conservar la indentació correcta.

En aquest cas, si féssim print(a) escriuria el text en 2 línies, ja que en la definició hem fet un salt de línia ("\n") implícitament.

2.1.Treballar amb Strings

Els strings són, en veritat, **llistes** de caràcters (cadascuna de les lletres o símbols). Veiem, amb alguns exemples, algunes coses que podem fer amb els Strings:

a = "Hola, bona tarda" c = a[1]	'c' contindrà la lletra 'H'
b = "Hola, bona tarda" s = b[2:5]	's' contindrà el text 'ola'
a = "Hola, bona tarda" n = len(a)	La funció **len()** retorna la longitud de caràcters del string, com una llista (16).

a = "Hola, bona tarda" s = a.replace("H", "M")	La funció "**replace()**" reemplaça un text per un altre en un string s = "Mola, bona tarda"
a = "Hola, com estem." v = a.split(",")	La funció "**split()**" divideix el string en substrings pel separador indicat com a paràmetre. 'v' conté : ['Hola', ' com estem.']
a = " Hola bona tarda " s = a.strip() print(s)	La funció "**strip()**" treu els espais al davant i al darrere d'un string. Mostra: "Hola bona tarda"
txt = "Bona tarda" if "tarda" in txt: print("Hem trobat tarda")	

Concatenació

La unió de 2 o més strings s'anomena **concatenació**, i en Python es pot fer amb l'operador "+":

Exemple 1:

```
a = "Hola"
b = "bona tarda"
c = a + b
print(c)          # Mostra: "Holabona tarda"
```

Exercici proposat 1: Sense modificar les variables a i b anteriors, fer que mostri per pantalla "Hola bona tarda" correctament espaiat.

En el cas de voler concatenar amb números, no es pot fer directament i s'ha de convertir el numero a String (ho farem amb la funció **str()**). Exemple:

```
n = 36
txt = "El nombre calculat és: " + str(n)
print(txt)
```

Pas de paràmetre a funcions

S'ha comentat abans que els Strings els podem tractar com a llistes, però una diferència important és que una variable **String es passa a una funció sempre per valor**.

3. Fitxers

3.1. Tipus de fitxers

Segons el seu format, resumidament, existeixen 2 tipus de fitxers: fitxers de text i fitxers binaris. Per a saber si un fitxer és de text o binari, només cal que l'obriu amb algun editor de text senzillet (com el Bloc de Notas de Windows): si hi podeu llegir el que conté, vol dir que és de text, i si hi veieu tot de símbols estranys, és que és binari. Un exemple de fitxer binari pot ser una fotografia, un vídeo, un fitxer de Word, etc. Un fitxer de text, usualment, té l'extensió ".txt". Els fitxers ".csv" també són de text.

En aquesta sessió ens centrarem en els fitxers de text.

3.2. Accions amb fitxers

D'una manera resumida, podem dir que podem fer 2 tipus d'accions amb fitxers:

- **Lectura**: Llegir el seu contingut
- **Escriptura**: Escriure-hi text. Dins d'aquesta segona opció tenim 2 opcions habituals:
    - **Escriptura**: Escriure text normalment.
    - **Append**: Afegir text al final del fitxer.

En aquest curs ens centrarem només en les accions de Lectura i Escriptura.

3.3. Lectura de fitxers

Si volem llegir el contingut d'un fitxer hem de seguir els següents passos:

1. **Obrir el fitxer en mode lectura**

   Aquesta acció consisteix en cridar a la funció del sistema "open" per a especificar el nom del fitxer que volem llegir. La sintaxi és:

   f = open("dades.txt","r");

   El que fem és obrir el fitxer "dades.txt" que hem de tenir a la mateixa carpeta on tenim el fitxer de Python (si no especifiquem cap carpeta, el fitxer l'anirà a buscar a la mateixa carpeta on estem ara). El segon paràmetre de la funció és **"r",** que especifica que obrim el fitxer en mode lectura (read), ja que l'objectiu és llegir dades.

   La funció "open" retorna una variable que farem servir d'ara en endavant per a cada acció sobre el fitxer.

2. **Llegir el seu contingut**

   Per a llegir el contingut d'un fitxer tenim vàries opcions: llegir-lo tot de cop, línia a línia, etc.

   Algunes funcions disponibles:

   ```
   f.read ()  #retorna el contingut del fitxer en una variable de tipus String
   f.readLine ()  #retorna una línia del fitxer en una variable de tipus String
   f.readLines ()  #retorna una llista de Strings amb les línies del fitxer
   ```

   Sempre hem d'especificar el fitxer del qual llegim (podríem tenir més d'un obert), i ho fem mitjançant la variable que retorna la funció **"open" ('f')**.

3. **Tancar-lo**

   Finalment, quan hem finalitzat la lectura de TOTES les dades, el tanquem. La sintaxi de la funció és:

   ```
   f.close()
   ```

3.4. Escriptura en fitxers

Si volem escriure informació en un fitxer, els passos a seguir són similars a la lectura:

1. **Obrir el fitxer**

   Aquesta acció consisteix en cridar a la funció del sistema "open" per a especificar el nom del fitxer on volem escriure dades. La sintaxi és:

   ```
   f = open("dades.txt","w")
   ```

   El que fem és crear el fitxer "dades.txt", ja que aquesta vegada l'obrim en mode escriptura, tal i com indica el segon paràmetre de la funció és **"w"** (write).

   La resta de la sintaxi està explicada en l'apartat anterior 3.3.

   Només afegir que, quan escrivim dades en un fitxer, podem tenir 2 possibilitats:

   - El fitxer no existeix: Es crea un fitxer amb el nom especificat on s'hi guardaran les dades.
   - El fitxer ja existeix (hi ha un fitxer amb el mateix nom i extensió): esborra tot el contingut del fitxer existent i hi guarda les noves dades des de l'inici.

Append

També podem obrir el fitxer en mode escriptura però sense esborrar el contingut, afegint dades al final. En aquest cas, l'hem d'obrir en mode "append", que és posar una "a" en lloc de "w" en la funció open:

```
f = open("dades.txt","a")        #escriptura en mode append
```

2. **Escriure-hi dades**

   Cridarem a la funció "**write()**" per a escriure les dades al fitxer. Aquesta funció espera un string com a paràmetre. Exemple:

   f.write ("Escrivim una frase com a exemple.")

   El string pot ser una concatenació de strings (comentat en la secció anterior de strings), també amb "\n" en cas de voler fer salts de línia, etc.

3. **Tancar-lo**

   Finalment, quan hem finalitzat l'escriptura de TOTES les dades, el tanquem, com abans:

   f.close()

2.5. Modes d'obrir un fitxer

Abans hem vist que hi ha 2 modes d'obrir un fitxer: "r" per a lectura i "w" per a escriptura. Per a resumir uns quants modes més:

Mode	Lectura	Escriptura	Append
Text	"r"	"w"	"a"
Binari	"rb"	"wb"	"ab"

Aquí podeu trobar un enllaç a el llistat de les funcions:

https://www.w3schools.com/python/python_ref_file.asp

2.6. Exemples

Anem a proposar 2 exemples que llegeixen o escriuen 2 nombres enters de/en un fitxer (considerem els 2 nombres en 2 línies del fitxer).

Llegim 2 nombres enters

```
1.  f = open ("dades.txt","r")  #obrim
2.
3.  #Llegim el primer enter
4.  n1 = float(f.readLine())
5.  print("Hem llegit el nombre: "+str(n1))
6.
7.  #Llegim el segon enter
8.  n2 = float(f.readLine())
9.  print("Hem llegit el nombre: "+str(n2))
10.
11. #Tanquem el fitxer
12. f.close()
13.
```

Escrivim 2 nombres enters

```
1.  f = open ("dades.txt","r")  #obrim
2.
3.  n1=2
4.  n2=40
5.
6.  #Escrivim el primer enter
7.  f.write(str(n1))
8.
9.  #Escrivim el segon enter
10. f.write(str(n2))
11.
12. #Tanquem el fitxer
13. f.close()
14.
15. print ("Escriptura finalitzada")
```

Exercici proposat 2: Fer que l'exemple anterior de lectura pugui llegir 10 dades i guardar-les en una llista. Suposem que cada dada està en una línia diferent al fitxer.

Exercici proposat 3: Fer que l'exemple anterior de lectura pugui escriure les dades d'una llista en un fitxer, cadascuna en una línia diferent.

Llegir un fitxer amb un 'for'

Quan llegim un fitxer, llegim les dades que conté via funcions (readLine, readLines, etc.).

De fet, no cal fer anar funcions, ja que podem llegir les línies d'un fitxer d'aquesta manera, amb un 'for':

```
1. f = open("dades.txt")
2. for linia in f:              #A cada pas del 'for' llegeix una línia
3.     print (linia)
```

Exercici proposat 4: Llegir els nombres 'float' d'un fitxer, línia a línia, seguint el format explicat en l'exemple anterior i mostrar la seva suma.

Exercici proposat 5: Llegir els nombres 'float' d'un fitxer, línia a línia, seguint el format explicat en l'exemple anterior i anar guardant-los en una llista. A priori, no sabem el nombre de dades que conté el fitxer.

4. Control d'errors: Try...Except

Quan hi ha un error en el programa (volem multiplicar variables que no son número, fem una divisió per 0, volem accedir a un índex inexistent d'un array, obrim un fitxer de lectura inexistent, etc.), Python para l'execució i genera un missatge d'error.

No obstant, nosaltres podem impedir que el programa acabi tractant aquest error de la manera que millor ens convingui. Això es fa envolvent el codi en el bloc **try...except**. La sintaxi bàsica és:

try:

Codi que s'executa

except:

Codi que s'executa quan el codi anterior ha donat un error.

Exemple:

```
1. try:
2.     f = open("dades.txt","r")
3.     txt = f.read()
4.     f.close()
5. except:
6.     print("Fitxer no trobat")
```

En aquest cas, l'únic error que es podia produir era que no trobés el fitxer. En cas de tenir el fitxer buit, el 'read()' retornarà a 'txt' un string buit "".

Comentaris:

- Podem **afegir el tipus d'error** al costat de "except". Això ens permet de poder mostrar missatges diferents segons l'error produït. Així, es poden afegit tants "except" com vulguem a un sol try:

  Exemple:

```
1. try:
2.     v=[None]*1
3.     f = open("dades.txt","r")
4.     v[0] = f.readLine()
5.     v[1] = f.readLine()
6.     f.close()
7. except IOError:          #si no troba el fitxer o, quan ha de llegir dades del iftxer, no en troba.
8.     print("Fitxer no trobat o hi ha menys de 2 línies al fixer")
9. except IndexError:       #si ens sortim de la llista
10.    print("Índex fora de rang de la llista")
11.
```

  En aquest cas donarà un error a la línia 5, ja que la llista només és de tamany 1 i la posició v[1] no existeix.

  En el següent enllaç podem trobar un llistat amb els tipus d'errors:

  https://www.tutorialspoint.com/python/python_exceptions.htm

- Podem **afegir un "else"** al final dels "except". El codi inclòs al "else" s'executaria si tot el codi s'executa sense errors. Exemple:

```
try:
    f = open("dades.txt","r")
    txt = f.read()
    f.close()
except:
    print("Fitxer no trobat")
else:
    print("Fitxer llegit sense problemes")

```

- Podem **afegir un "finally"** al final dels "except" o "else". El codi inclòs al "finally" s'executaria tant si el codi de "try" s'executa correctament o amb errors.

```
try:
    f = open("dades.txt","r")
    txt = f.read()
    f.close()
except:
    print("Fitxer no trobat")
else:
    print("Fitxer llegit sense problemes")
finally:
    print("Procés acabat.")

```

Els afegits "**else**" i "**finally**" són opcionals.

Exercici proposat 6: Escriure els 10 primers nombres Naturals (1,2,3,...,10) en un fitxer de text, línia a línia. Afegeix el try...except .

Exercici proposat 7: Escriure els 10 primers nombres Naturals (1,2,3,...,10) en un fitxer de text, en una sola línia, separats per una coma. Afegeix el try...except .

5. Solucions als exercicis proposats

Exercici proposat 1: Sense modificar les variables a i b anteriors, fer que mostri per pantalla "Hola bona tarda" correctament espaiat.

```
a = "Hello"
b = "World"
c = a + " " + b
print(c)
```

Exercici proposat 2: Fer que l'exemple anterior de lectura pugui llegir 10 dades i guardar-les en una llista. Suposem que cada dada està en una línia diferent al fitxer.

```
1. v=[None]*10
2. f = open ("dades.txt","r")  #obrim per a llegir
3.
4. for i in range(10):
5.     v[i] = float(f.readLine())
6.
7. f.close()
8.
9. print(v)
```

Exercici proposat 3: Fer que l'exemple anterior de lectura pugui escriure les dades d'una llista en un fitxer, cadascuna en una línia diferent.

```
1. v=[1,2,3,4,5,6,7,8,9,10]
2. f = open ("dades.txt","w")  #obrim per a escriure
3.
4. for x in v:
5.     f.write(x)
6.     f.write("\n")
7.
8. f.close()
9.
```

Exercici proposat 4: Llegir els nombres 'float' d'un fitxer, línia a línia, seguint el format explicat en l'exemple anterior, i mostrar la seva suma.

```
1. f = open ("dades.txt","r")  #obrim per a llegir
2.
3. suma = 0
4. for linia in f:
5.     suma += float( linia )
6.
7. f.close()
8.
9. print(suma)
```

Exercici proposat 5: Llegir els nombres 'float' d'un fitxer, línia a línia, seguint el format explicat en l'exemple anterior, i anar guardant-los en una llista. A priori, no sabem el nombre de dades que conté el fitxer.

Solució: En aquest cas, com no sabem a priori el tamany de la llista (no sabem quants nombres hi ha al fitxer), creem una llista buida i hi anem afegint els valors (amb "append"), transformats a 'float', ja que cada línia es llegeix com un String.

```
f = open ("dades.txt","r")          #obrim per a llegir

v=[]
for linia in f:
   v.append(float( linia ))

f.close()
print(v)
```

Exercici proposat 6: Escriure els 10 primers nombres Naturals (1,2,3,...,10) en un fitxer de text, línia a línia. Afegeix el try...except

Solució: En aquest cas, com volem escriure nombres i la funció 'write(...)' només accepta Strings, hem de convertir els nombres a Strings fent servir la funció 'str(...)':

```
try:
    f = open ("fout.txt","w")        #obrim per a llegir

    for i in range(10):
       f.write( str( i+1 ) )
       f.write("\n")       #fem un salt de línia després d'escriure cada nombre

    f.close()
except:
   print("Problemes per a guardar les dades al fitxer")
```

Exercici proposat 7: Escriure els 10 primers nombres Naturals (1,2,3,...,10) en un fitxer de text, en una sola línia, separats per una coma. Afegeix el try...except .

Solució: En l'exercici anterior hem decidit sempre posar un salt de línia després de cada nombre, i en aquest s'ha decidit no posar una coma després del darrer nombre. Això ho implementem amb un simple 'if':

```
try:

   f = open ("fout.txt","w")  #obrim per a llegir

   for i in range(10):
       f.write(str(i+1))
       if i<9:
         f.write(",")

   f.close()
except:
   print("Problemes per a guardar les dades al fitxer")
```