Università degli Studi di Cagliari
Corsi di Laurea in Ingegneria Chimica ed Ingegneria Meccanica

FONDAMENTI DI INFORMATICA

http://people.unica.it/gianlucamarcialis

A.A. 2018/2019

Docente: **Gian Luca Marcialis**

LINGUAGGIO Python
Gestione dei file

Sommario

- Scrivere e leggere da qualunque I/O
- Il file visto da Python
  - Apertura
  - Scrittura e lettura
  - Chiusura
- Esercizi

File e Python

- La maggior parte dei programmi non accettano dati da tastiera né scrivono il risultato sul monitor
- In generale avranno bisogno di recuperare i primi e depositare gli ultimi su file
- Motivazioni nell'uso dei file:
  - Quantità dei dati
  - Maggiore velocità nell'immissione e sicurezza nel salvataggio
  - I dati su file possono essere usati più volte dallo stesso programma o da programmi diversi (v. Basi di Dati)

Gestione di file

- `fp = open(NomeFile, Modalità_di_accesso)`
  - "Apre" la lavagna (stream) identificata da `NomeFile` (assegnando un valore a fp)
  - `NomeFile` è una stringa con il nome del file. Es. `NomeFile="pippo.txt"`
  - Modalità è una stringa di uno o due caratteri:
    - `"w"` → se il file è aperto in scrittura
    - `"r"` → se è aperto in lettura
    - `"a"` → se è aperto in modalità "append", ovvero si vogliono aggiungere ai contenuti presenti nuovi contenuti scritti "in fondo" al file
- `fp.close()`
  - E' il metodo (funzione «dedicata» agli identificatori di file) che chiude il file dopo che su esso sono state compiute le operazioni desiderate
  - Un file va **sempre** chiuso al termine del suo utilizzo
    - Se un file in scrittura non viene chiuso, nessuna modifica sarà salvata

Scrittura/lettura su file

Data una stringa `s` ed un file `f`	
`f.write(s)`	Scrive `s` sul file `f`
`s=f.readline()`	Legge una riga del file `f` e lo salva in `s`. Restituisce "" se il file è finito.

Esempio: sia dato il file di testo sottostante

```
123 45 ciao
Classe V
34.5 temperatura corporea
```

Dopo aver aperto il file **in lettura** ed aver assegnato `f` tramite `open`, scrivendo:

```
s=f.readline()
```

`s` conterrà la stringa:

```
"123 45 ciao"
```

Scrivendo un'altra volta:

```
s=f.readline()
```

`s` conterrà la stringa:

```
"Classe V"
```


Esercizio 1

- Scrivere una programma Python che, leggendo da tastiera una stringa, la salvi su file «stringa.txt»
- Aprire il file «stringa.txt» e verificare il salvataggio

```
stringa=raw_input("Immetti una stringa")
f=open("stringa.txt", "w")
f.write(stringa)
f.close()
```


Esercizio 2

- Scrivere un programma che salva in un file «numeri.txt» l'insieme dei numeri naturali da 0 ad un numero n letto da tastiera.
- I numeri salvati devono risultare su righe del file differenti. Esempio, se n=3, nel file deve risultare:

```
0
1
2
3
```


Soluzione

```
n=input("Immetti un numero naturale:")

f=open("numeri.txt", "w")
i=0
while i<=n:
    f.write(str(i)+"\n")#per l'a capo
    i=i+1
f.close()
```

Esercizio 3

- Aprite un file di testo, scriveteci a caso 10 numeri interi, uno per riga. Salvate il file con il nome «esercizio.txt»
- Scrivere un programma che legga i 10 numeri dal file appena salvato e li stampi in un'unica riga sullo schermo

Soluzione

```
filep=open("esercizio.txt", "r")
s=""
for i in range(10):
    n=file.readline()
    s=s+n[0:len(n)-1]+" "
        #rimuovo il carattere 'a capo'
filep.close()
print s
```


Soluzione alternativa

```
filep=open("esercizio.txt", "r")
s=""
for i in range(10):
    n=file.readline()
    n=int(n)
    s=s+str(n)+" "
filep.close()
print s
```


La fine del file

- In Python, l'operatore `readline()` legge una riga e la restituisce come stringa.
- Ma se il file è finito? Ovvero, non c'è più niente?
  - La funzione restituisce il carattere vuoto ""
- Questo ci permette di creare una condizione di «fine file» in modo da leggere finché non arriviamo al termine, secondo lo schema:

```
linea=f.readline()
while linea!="":
    ...istruzioni da eseguire...
    linea=f.readline() #leggi prossima linea
```

Esercizio

- Scrivere un programma Python che legga da file ”numeriEstringhe.txt” il suo contenuto che è costituito da numeri e stringhe alternate a caso, separate da «a capo», e ponga ogni elemento in una lista.
- Esempio:

```
Ciao
-2.4
1
0.4
buonanotte
```


Soluzione

```
f=open(”numeriEstringhe.txt”, ”r”)
l=[]
linea=f.readline()
while linea!= ””: #se il file non è finito…
     l=l+[linea]
     linea=f.readline()
f.close()
```


Esercizio

- Scrivere un programma Python che legga da file ”numeriEstringhe.txt” il suo contenuto che è costituito da numeri e stringhe alternate a caso, separate da «a capo», e gli elementi formati **da sole cifre** nella lista `numeri` e tutti gli altri nella lista `stringhe`.
- Esempio:

```
Ciao
-2.4
1
404
buonanotte
```

1, 404 → numeri
”Ciao”, ”-2.4”, ”buonanotte” → stringhe

Soluzione

```
f=open(”numeriEstringhe.txt”, ”r”)
numeri=[]
stringhe=[]
linea=f.readline()
while linea!= ””: #se il file non è finito…
     if linea.isdigit() :
          numeri=numeri+[linea]
     else:
          stringhe=stringhe+[linea]
     linea=f.readline()
f.close()
```

Esercizio

- Scrivere un programma che legge da file una sequenza di interi, separati da “a capo” (quindi un intero per riga), e la memorizza in una lista.
  - Il nome del file viene fornito da tastiera.
  - Il vettore può avere al massimo 50 valori, ma il file presenta un numero di interi non definito.
- Si stampi a video eventualmente un avviso se la lunghezza massima del vettore è stata raggiunta interrompendo l’acquisizione di valori
- Si stampi a video infine il vettore acquisito.

Soluzione

```
#Programma che legge max 50 interi da file e li memorizza in un vettore

nomefile=raw_input("Inserire il nome del file:\n")

fp=open(nomefile,"r") #senza le virgolette: è una variabile, non una stringa

lista=[] #inizializzazione del vettore
i=0      #inizializzazione del contatore di interi letti da file
linea=fp.readline()
while((linea!="") and (i<50)):     #finché il file non è finito AND i<50
   lista = lista + [int(linea)]
   i=i+1                           #incrementa i
   linea=fp.readline()

if(i==50):
   print("\nMax numero di valori acquisibile raggiunto.\n")

fp.close()

print lista
```


Esercizio

- E’ dato un file «dati.txt» caratterizzato dal seguente formato rappresentante delle coppie nome/età. Per esempio:

```
Pippo 23
Ada 19
Felice 32
Geronima 40
Ciccio 19
```

- Si legga tutto il file creando un dizionario **le cui chiavi sono fornite dall’età.** A ciascuna età viene associata una lista con i nomi di persone che hanno quell’età.
- Il dizionario generato dal file di cui sopra sarà:

```
{19:["Ada", "Ciccio"], 23:["Pippo"], 32:["Felice"],
40:["Geronima"]}
```


Soluzione

```
f=open("dati.txt","r") #apro in lettura
d={}                   #inizializzo il dizionario
s=f.readline() #leggo una riga
while s!="": #finché il file non è finito
  s=s.split() #divido la riga nei campi dati
  eta=int(s[1]) #converto il secondo elemento
  if eta in d: #se l'età è già presente
    d[eta]=d[eta]+[s[0]] #aggiungo il nome
  else:                  #altrimenti
    d[eta]=[s[0]]    #aggiungo età e nome
  s=f.readline()     #leggo una nuova riga
f.close()            #chiudo il file
```

Esercizio 2

- Dato un dizionario popolato secondo l'esercizio precedente, scrivere un programma Python che, letta da tastiera la stringa indicante un nome proprio, stampi su file l'età associata al nome indicato se presente, altrimenti scriva su file "non presente".
- Il nome del file è dato dal nome inserito a cui viene aggiunta l'estensione ".txt"
- **Esempio**.

Si consideri il dizionario dell'esercizio precedente. Se da tastiera viene inserito il nome "Felice", il programma scriverà su file "Felice.txt" il valore 32 (vedi esempio).

Soluzione

```
nome=input("Inserisci un nome:") #leggo un nome
presente=False #tecnica: uso una variabile per verificare
               #l'evento:"ho trovato il nome nel dizionario"

f=open(nome+".txt","w") #apro il file in scrittura

#inizio la ricerca
for eta in d: #per ogni età presente in d
    if nome in d[eta]:    #se il nome compare almeno una volta
        f.write(str(eta)+"\n") #stampo l'età
        presente=True   #ho trovato il nome nel dizionario

if not presente: #se non l'ho trovato nemmeno una volta
    f.write("Non presente") #scrivo che non è presente

f.close() #chiudo il file
```


Per saperne di più

- K.A. Lambert, *Programmazione in Python*, Cap. 4, Apogeo