Uso dei file (di testo)

Informatica@DSS 2025/2026

Massimo Lauria <massimo.lauria@uniroma1.it>
https://massimolauria.net/informatica2025/

Dati temporanei

Dati elaborati da un programma

- presenti in memoria RAM (veloce)
- scompaiono se il programma termina

come possiamo mantenere dati permanenti?

Dati permamenti

I **file** sono unità di dati

- permamenti
- memorizzati sul disco rigido o su internet
- letti o scritti da programmi

Permettono lo scambio e la comunicazione tra

- programmi
- computer
- momenti

Successione ordinata di bytes.

Un file di lunghezza L byte è

$$b_0\ b_1\ b_2\ \ldots\ b_{L-1}$$

dove b_i è il byte (quindi otto bit) alla posizione i del file.

File di testo

Un file può contenere dati di natura arbitraria, ma a noi interesseranno principalmente **file di testo**.

Esempio:

```
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do
eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad
minim veniam, quis nostrud exercitation ullamco laboris nisi ut
aliquip ex ea commodo consequat. Duis aute irure dolor in
reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla
pariatur. Excepteur sint occaecat cupidatat non proident, sunt in
culpa qui officia deserunt mollit anim id est laborum.
```

Codifica di un file di testo

Un file di testo è un file che può essere **visto** come una sequenza di caratteri $c_0, c_1, \ldots$.

$$\underbrace{b_0 b_1}_{c_0} \quad \underbrace{b_2 b_3 b_4}_{c_1} \quad \underbrace{b_5}_{c_2} \quad \underbrace{b_6 b_7}_{c_3} \ldots$$

```
i    -> 01101001
à    -> 11000011 10100000
ciao -> 01100011 01101001 01100001 01101111
già  -> 01100111 01101001 11000011 10100000
```

Lo schema di traduzione da bit a caratteri è la **codifica** o **encoding** del testo. Lo standard moderno è **utf-8**.

Operazioni di base

Apertura e chiusura di un file

Un file va *aperto* prima di usarlo, e dopo va *chiuso*.

```
f = open('documento.txt', encoding='utf-8') # Apre il file          1
                                                                    2
# operazioni varie sul file                                         3
                                                                    4
f.close()                                   # chiude il file        5
```

Se un file non esiste…

```
f = open('non_esiste.txt', encoding='utf-8')                    1
```

```
: Traceback (most recent call last):
:   File "<stdin>", line 1, in <module>
: FileNotFoundError: [Errno 2] No such file or directory: 'non_esiste.txt'
```

Schema consigliato

Invece di usare

```
f = open('file.txt', encoding='utf-8')
istruzione1
istruzione2
istruzione3
f.close()
```

usiamo

```
with open('file.txt', encoding='utf-8') as f:
    istruzione1
    istruzione2
    istruzione3
```

allora il file viene **sempre** chiuso se si esce dal blocco.

Riassumendo

Per aprire un file `documento.txt`

- esistente
- in sola lettura

```
with open('documento.txt') as f:                                    1
    print('Il file documento.txt è stato aperto')                   2
    print('e la variable "f" permette di accedervi')                3
```

```
Il file documento.txt è stato aperto
e la variable "f" permette di accedervi
```

Quale codifica dei testi?

La codifica **utf-8** è la più diffusa ed è quella di tutti i file che useremo. Se non indicate esplicitamente la codifica, ad esempio come in

```
open("file.txt")                                                    1
```

il file verrà aperto con la codifica di default del sistema. Purtroppo **Windows** usa la codifica `cp1252`, che non è compatibile con utf-8.

Vi conviene indicare la codifica in maniera esplicita.

Percorso del file

```
# document.txt è nella cartella dove gira il programma                    1
with open('documento.txt', encoding='utf-8') as f1:                        2
    ...                                                                    3
                                                                           4
# percorso relativo su Mac e Linux                                        5
with open('../didattica/eval.text', encoding='utf-8') as f2:               6
    ...                                                                    7
                                                                           8
# percorso assoluto Linux                                                 9
with open('/usr/share/dict/README', encoding='utf-8') as f3:               10
    ...                                                                    11
                                                                           12
# percorso assoluto su windows                                            13
with open('C:\\Documents\\romanzo.txt', encoding='utf-8') as f4:           14
    ...                                                                    15
```

Leggere i file di testo

File di testo dal Progetto Gutemberg

Project Gutenberg

search for books

- Browse Catalog
- Bookshelves
- Main Page
- Categories
- Contact Info

Project Gutenberg appreciates your donation!

Donate

- Why donate?

in other languages

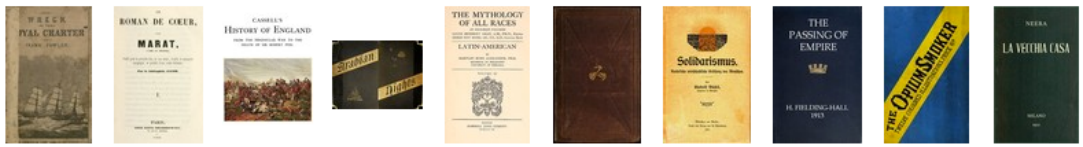
- Português
- Deutsch
- Français

hosted by ibiblio

Free ebooks - Project Gutenberg

Book search · Book categories · Browse catalog · Mobile site · Report errors · Terms of use

Some of the Latest Books



Welcome

Project Gutenberg offers over 57,000 free eBooks. Choose among free epub books, free kindle books, download them or read them online. You will find the world's great literature here, with focus on older works for which copyright has expired. Thousands of volunteers digitized and diligently proofread the eBooks, for enjoyment and education.

No fee or registration is required. If you find Project Gutenberg useful, please consider a small donation, to help Project Gutenberg digitize more books, maintain our online presence, and improve Project Gutenberg programs and offerings. Other ways to help include digitizing more books, recording audio books, or reporting errors.

Documenti sulla pagina del corso

Nella sezione ”Materiale” del corso ci sono dei libri di pubblico dominio in Inglese, in file `.txt` codificati in UTF-8.

- *Alice's Adventures in Wonderland* di Lewis Carroll
- *Frankenstein* di Mary W. Shelley
- *The Adventures of Sherlock Holmes* by A. C. Doyle
- *Moby Dick* di Herman Melville
- *The Prince* di Niccoló Machiavelli
- *Treasure Island* di Robert Louis Stevenson

Leggere i caratteri dal file `f`

- `f.read()` legge tutti i caratteri non ancora letti
- `f.read(n)` legge fino a `n` caratteri non ancora letti

```
with open('assets/alice.txt', encoding='utf-8') as f:
    f.read(711)   # roba di copyright, saltiamola
    print(f.read(32))
    print(f.read(20))
    print(f.read(20))
    print(f.read(20))
    print(f.read(20))
```

```
CHAPTER I. Down the Rabbit-Hole


Alice was beginning
 to get very tired o
f sitting by her sis
ter on the
bank, and
```

Caricare tutto il testo in una stringa

```
with open('assets/alice.txt', encoding='utf-8') as alice:
    testo=alice.read()
    print('# INFO: ',len(testo),type(testo))
    print(testo[711:1046])
    print("# Sono rimasti",len(alice.read()),'caratteri.')
```

```
# INFO:  163816 <class 'str'>
CHAPTER I. Down the Rabbit-Hole

Alice was beginning to get very tired of sitting by her sister on the
bank, and of having nothing to do: once or twice she had peeped into the
book her sister was reading, but it had no pictures or conversations in
it, ‘and what is the use of a book,’ thought Alice ‘without pictures or
conversations?’
# Sono rimasti 0 caratteri.
```

Esempio di estrazione dati

Consideriamo un file di testo `numeri.txt`

```
-345 -324
580 784 -636 524 478 695
-795 -752 273 -912
-485 349 278 384 -653 328 148 317
```

```
testo=''                                                    1
with open("assets/numeri.txt", encoding='utf-8') as f:      2
    testo = f.read()                                        3
numeri = testo.split()                  # segmentiamo       4
numeri = [ int(x) for x in numeri ] # convertiamo           5
print(numeri)                                               6
```

```
[-345, -324, 580, 784, -636,
  524, 478, 695, -795, -752,
  273, -912, -485, 349, 278,
  384, -653, 328, 148, 317]
```

Promemoria su `repr`

Visualizza i caratteri non stampabili di una stringa

```
testo = 'Prima riga\n\nSeconda riga.'
print(testo)
print(repr(testo))
```

```
Prima riga

Seconda riga.
'Prima riga\n\nSeconda riga.'
```

Leggere il file riga per riga

```
with open('assets/alice.txt', encoding='utf-8') as alice:          1
    print(repr(alice.readline()))                                  2
    print(repr(alice.readline()))                                  3
    print(repr(alice.readline()))                                  4
```

```
'Project Gutenberg's Alice's Adventures in Wonderland, by Lewis Carroll\n'
'\n'
'This eBook is for the use of anyone anywhere at no cost and with\n'
```

La funzione `readline` legge i caratteri fino al prossimo `\n` (che è incluso nel testo)

Caricare tutte le righe in memoria

Il metodo `readlines()` legge tutte le righe

```
with open('assets/alice.txt', encoding='utf-8') as alice:
    righe=alice.readlines()

    print(righe[10],end='')
    print(righe[120],end='')
    print(righe[154],end='')
    print('')
    print("Il file ha {} righe ".format(len(righe)))
```

```
Author: Lewis Carroll
sort of way, ‘Do cats eat bats? Do cats eat bats?’ and sometimes, ‘Do
into the loveliest garden you ever saw. How she longed to get out of

Il file ha 3736 righe
```

Ciclo for su un file (di testo)

Ogni iterazione considera una **riga** del file

```
with open('assets/alice.txt', encoding='utf-8') as alice:
    maxlength=0
    for riga in alice:
        maxlength = max(maxlength,len(riga))
    print("La riga più lunga ha",maxlength,"caratteri.")
```

```
La riga più lunga ha 79 caratteri.
```

Ricerca in un file

Vediamo un esempio di come ottenere tutte le righe che contengono una data parola, in un file.

```
def trova_righe(nome_file, stringa):
    '''Trova nel file

    Restituisce la lista dei numeri delle righe del
    file nome_file in cui appare la stringa.'''
    with open(nome_file, encoding='utf-8') as f:
        lista_righe = []
        riga_corrente = 1
        for riga in f:
            if riga.find(stringa) != -1:
                lista_righe.append(riga_corrente)
            riga_corrente += 1
        return lista_righe
```

Ricerca in un file (esempio)

```
indici = trova_righe('assets/alice.txt', 'Turtle')
print(indici)
```

```
[2214, 2354, 2356, 2358, 2372, 2390, 2397, 2403, 2410,
 2411, 2415, 2420, 2422, 2426, 2432, 2439, 2442, 2449,
 2453, 2457, 2464, 2469, 2484, 2486, 2494, 2500, 2509,
 2521, 2533, 2537, 2551, 2560, 2564, 2572, 2578, 2584,
 2588, 2595, 2627, 2633, 2639, 2641, 2680, 2685, 2690,
 2697, 2708, 2713, 2741, 2747, 2752, 2775, 2783, 2785,
 2787, 2790, 2813, 3348, 3358]
```

```
indici = trova_righe('assets/alice.txt', 'Alice')
print(len(indici))
```

```
396
```

Scrivere su file

Aprire un file in modalita scrittura

```
with open('testo.txt', 'w') as f:
    f.write('Scrivi questo contenuto nel file.')

with open('testo.txt') as f:
    print(f.read())
```

```
Scrivi questo contenuto nel file.
```

Per scrivere su un file

- aggiungere `'w'` come argomento di `open`
- usare `write()` per scrivere testo

Nota: il contenuto precedente viene cancellato

Modalità di apertura del file

Si può indicare la **modalità** di apertura del file

Modalità	Operazione	Tipo di file
`r`	Lettura	Testo
`w`	Scrittura ex novo	Testo
`a`	Scrittura in coda	Testo

- `r` è la modalità di default
- `w` scrive cancellando tutto il contenuto precedente
- `a` aggiunge testo in coda

Scrivere nel file sequenze di caratteri

```
with open('testo.txt', 'w') as f:                    1
    f.write('Primo*')                                2
    f.write('Secondo*')                              3
    f.write('Terzo')                                 4
    f.write('\n')                                    5
    f.write('Quarto e quinto.')                      6
                                                     7
with open('testo.txt') as f:                         8
    print(f.read())                                  9
```

```
Primo*Secondo*Terzo
Quarto e quinto.
```

Scrivere nel file sequenze di stringhe

```
with open('testo.txt', 'w') as f:                                    1
    f.writelines(['Primo','Secondo','Terzo','Quarto'])               2
    f.writelines(['Quinto','Sesto'])                                 3
    f.write('\n')                                                    4
    f.writelines(['Primo\n','Secondo\n','Terzo\n','Quarto\n'])       5
    f.writelines(['Quinto\n','Sesto\n'])                             6
                                                                     7
with open('testo.txt') as f:                                         8
    print(f.read())                                                  9
```

```
PrimoSecondoTerzoQuartoQuintoSesto
Primo
Secondo
Terzo
Quarto
Quinto
Sesto
```

Scrivere in coda al file

```
with open('testo.txt', 'w') as f:
    f.write('Linea1.\n')
with open('testo.txt', 'a') as f:
    f.write('Linea2.\n')
with open('testo.txt') as f:
    print(f.read())

print('------')

with open('testo.txt', 'w') as f:
    f.write('Linea Alternativa.\n')
with open('testo.txt') as f:
    print(f.read())
```

```
Linea1.
Linea2.

------
Linea Alternativa.
```

Importanza di usare la giusta codifica (1)

Consideriamo il file `assets/godel.txt` codificato in utf-8, con il seguente contenuto

```
Gödel è un famoso logico
```

```
with open("assets/godel.txt", encoding='utf-8') as f:
    testo= f.read()

print(testo)
```

```
Gödel è un famoso logico

```

Importanza di usare la giusta codifica (2)

Consideriamo il file `assets/godel.txt` codificato in utf-8, con il seguente contenuto

```
Gödel è un famoso logico
```

```
with open("assets/godel.txt", encoding='latin1') as f:
    testo= f.read()
print(testo)

with open("assets/godel.txt", encoding='cp1252') as f:
    testo= f.read()
print(testo)
```

```
GÃ¶del Ã¨ un famoso logico

GÃ¶del Ã¨ un famoso logico
```

Importanza di usare la giusta codifica (3)

Consideriamo il file `assets/godel.txt` codificato in utf-8, con il seguente contenuto

```
Gödel è un famoso logico
```

```
with open("assets/godel.txt", encoding='ascii') as f:
    testo= f.read()
print(testo)
```

```
Traceback (most recent call last):
  ...
UnicodeDecodeError: 'ascii' codec can't decode byte 0xc3 in
                    position 1: ordinal not in range(128)
```