

ESERCIZIO BONUS S2

TRACCIA:

Scrivi un programma Python per gestire una lista della spesa. Il programma deve permettere all'utente di:

- 1) Aggiungere un elemento alla lista.*
- 2) Rimuovere un elemento dalla lista (se presente).*
- 3) Visualizzare tutti gli elementi della lista ordinati in ordine alfabetico.*
- 4) Salvare la lista su file.*
- 5) Caricare una lista da file.*

Il programma deve avere un menu che consente all'utente di scegliere le varie operazioni e deve terminare solo quando l'utente lo richiede.

Per svolgere questo esercizio, creiamo il nostro solito file iniziale utilizzando nano, e successivamente iniziamo a scrivere il codice.

Utilizzerò il modulo ***import os***, che permette di interagire con il sistema operativo. In questo caso, lo utilizziamo per verificare se il file della lista della spesa esiste già.

La prima funzione che utilizziamo è ***salva_lista***, che salva la lista della spesa su un file. Il nome del file è passato come argomento (ad esempio, "lista_spesa.txt") e la lista è un elenco di elementi.

Essa apre il file in modalità scrittura ('w').

Scrive gli elementi della lista nel file, separando ciascun elemento con un ritorno a capo (\n), usando il metodo "\n".join(lista).

Questo crea una stringa in cui ogni elemento della lista è separato da una nuova riga.

Una volta che la lista è scritta, il file viene automaticamente chiuso grazie all'uso del ***with***, che gestisce la chiusura del file anche in caso di errore.



```
GNU nano 8.1                                eserciziobonus
import os
"""Viene importato il modulo os che permette di interagire con il sistema operativo,
   lo utilizziamo per verificare se il file della lista della spesa esiste già"""
filename = "lista_spesa.txt"

"""Salva la lista su un file"""
def salva_lista(lista):
    with open(filename, 'w') as file:
        file.writelines([item + '\n' for item in lista])
    print("Lista salvata.")
```

La seconda funzione utilizzata è **carica_lista**.

La funzione prende un parametro chiamato **filename**, che rappresenta il percorso (e il nome) del file che vogliamo leggere. Può essere un file con qualsiasi estensione, ad esempio **.txt**.

os.path.exists(filename): Questa funzione verifica se il file esiste davvero nel percorso specificato da filename. Se il file esiste, il codice dentro il blocco **if** viene eseguito.

open(filename, 'r'): Apre il file specificato in modalità lettura ('r').

with: Utilizziamo with per gestire l'apertura e la chiusura automatica del file. Quando il blocco with termina, il file viene chiuso automaticamente.

file è il nome che abbiamo dato all'oggetto che rappresenta il file aperto. Questo oggetto ci permette di leggere le righe del file.

```
"""Carica la lista da un file, se esistente"""
def carica_lista():
    if os.path.exists(filename):
        with open(filename, 'r') as file:
            return [line.strip() for line in file]
    return []
```

Adesso utilizziamo le funzioni per poter aggiungere e rimuovere elementi dalla lista:

1. Funzione **aggiungi_elemento(lista)**:

Chiede all'utente di inserire un elemento tramite `input()`.

Il metodo `.strip()` rimuove gli eventuali spazi bianchi iniziali e finali dal testo inserito.

L'elemento viene quindi aggiunto alla lista usando il metodo **append()**, utile per aggiungere un singolo elemento alla fine di una lista

Funzione **rimuovi_elemento(lista)**:

Chiede all'utente di inserire l'elemento da rimuovere tramite **input()**.

Controlla se l'elemento è presente nella lista con `if elemento in lista`.

Se l'elemento è trovato, lo rimuove con **remove()** e stampa un messaggio di conferma.

Se l'elemento non è trovato, viene mostrato un messaggio che informa l'utente.

```
def aggiungi_elemento(lista):  
    lista.append(input("Aggiungi un elemento: ").strip())  
  
def rimuovi_elemento(lista):  
    elemento = input("Rimuovi un elemento: ").strip()  
    if elemento in lista:  
        lista.remove(elemento)  
        print(f"Elemento '{elemento}' rimosso.")  
    else:  
        print("Elemento non trovato.")
```

La funzione **visualizza_lista**, prima di permettere all'utente di visualizzare la lista, essa viene ordinata con `sorted(lista)`.

Inoltre effettua un controllo, se la lista è vuota, viene visualizzato un messaggio informativo.

```
def visualizza_lista(lista):
    print("\nLista della spesa (ordinata):")
    for item in sorted(lista):
        print(f"- {item}")
```

Adesso passiamo all'ultima parte del programma:

L'esercizio ci chiede esplicitamente la creazione di un menu per l'utente, e lo andiamo a creare attraverso la funzione **def_menu()**, includendo tutte le opzioni che ci servono.

Per fare in modo che il programma si chiuda soltanto nel momento in cui l'utente lo decide, abbiamo bisogno di entrare in un ciclo infinito utilizzando **while true**. In pratica, il ciclo while True è un modo per mantenere il programma in esecuzione fino a quando non si verifica una condizione di **interruzione esplicita**, come ad esempio un comando dell'utente (ad esempio "Uscita", break).

```
def menu():
    print("\n1. Aggiungi")
    print("2. Rimuovi")
    print("3. Visualizza")
    print("4. Salva")
    print("5. Carica")
    print("6. Esci")

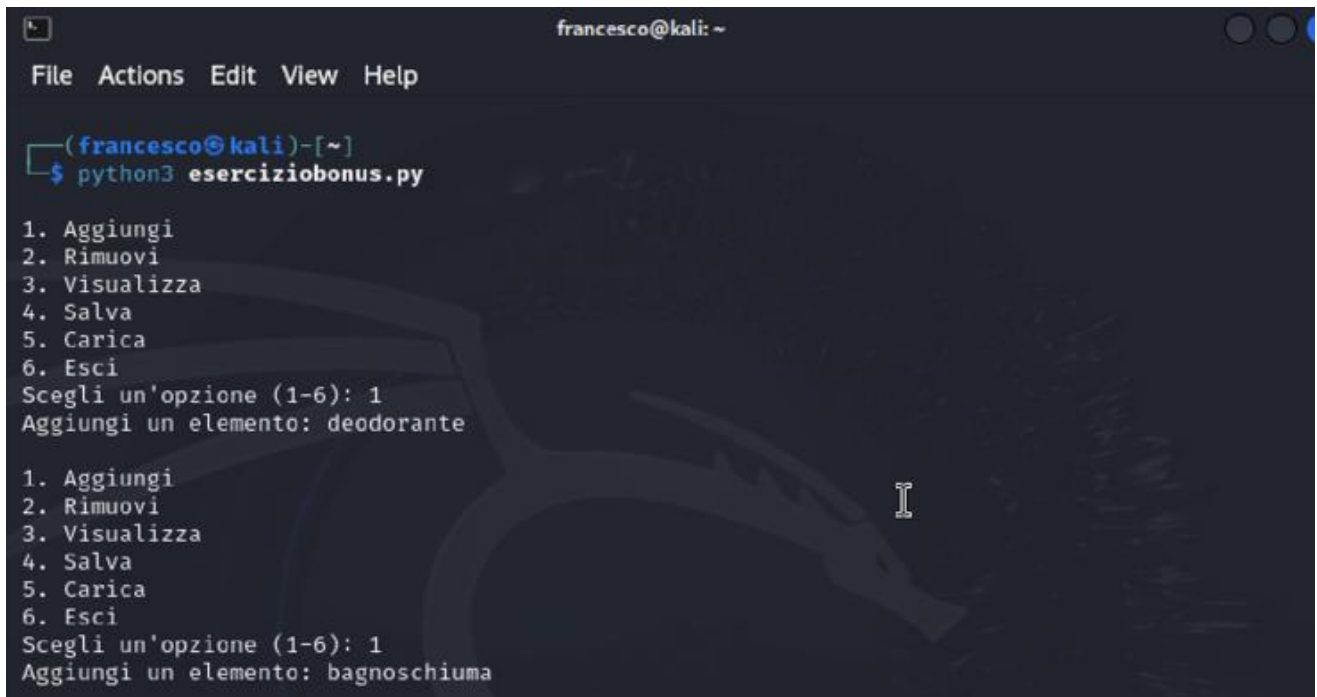
def main():
    lista_spesa = carica_lista()

    while True:
        menu()
        scelta = input("Scegli un'opzione (1-6): ").strip()

        if scelta == '1':
            aggiungi_elemento(lista_spesa)
        elif scelta == '2':
            rimuovi_elemento(lista_spesa)
        elif scelta == '3':
            visualizza_lista(lista_spesa)
        elif scelta == '4':
            salva_lista(lista_spesa)
        elif scelta == '5':
            lista_spesa = carica_lista()
            print("Lista caricata.")
        elif scelta == '6':
            print("Uscita ... ")
            break
        else:
            print("Opzione non valida.")

main()
```

Il codice adesso è completo, andiamo a verificare la funzionalità di questo programma:

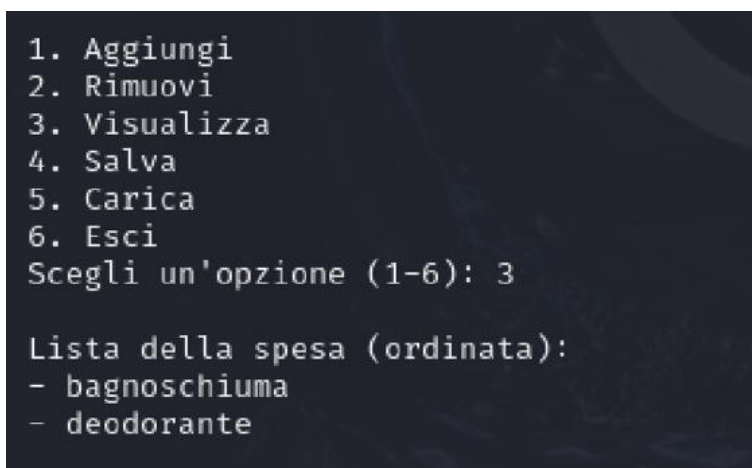


```
francesco@kali: ~  
File Actions Edit View Help  
(francesco@kali)-[~]  
$ python3 eserciziobonus.py  
1. Aggiungi  
2. Rimuovi  
3. Visualizza  
4. Salva  
5. Carica  
6. Esci  
Scegli un'opzione (1-6): 1  
Aggiungi un elemento: deodorante  
  
1. Aggiungi  
2. Rimuovi  
3. Visualizza  
4. Salva  
5. Carica  
6. Esci  
Scegli un'opzione (1-6): 1  
Aggiungi un elemento: bagnoschiuma
```

Come possiamo vedere dall'immagine, in una prima fase di prova, il programma ci chiede subito (attraverso il menu) di effettuare una scelta, scelgo due volte l'opzione numero 1 per aggiungere due elementi alla lista aventi iniziali diverse.

In questo modo potremo verificare se durante la visualizzazione il programma segue l'ordine alfabetico che gli abbiamo imposto.

Dopo aver aggiunto due elementi, chiedo al programma di visualizzare la lista, ecco il risultato:



```
1. Aggiungi  
2. Rimuovi  
3. Visualizza  
4. Salva  
5. Carica  
6. Esci  
Scegli un'opzione (1-6): 3  
  
Lista della spesa (ordinata):  
- bagnoschiuma  
- deodorante
```

Come possiamo vedere, il programma ci fa visualizzare correttamente la lista in ordine alfabetico.

Adesso proviamo le altre funzioni, rimuoviamo un elemento dalla lista e proviamo a visualizzarla nuovamente.

```
Lista della spesa (ordinata):  
- bagnoschiuma  
- deodorante  
  
1. Aggiungi  
2. Rimuovi  
3. Visualizza  
4. Salva  
5. Carica  
6. Esci  
Scegli un'opzione (1-6): 2  
Rimuovi un elemento: deodorante  
Elemento 'deodorante' rimosso.  
  
1. Aggiungi  
2. Rimuovi  
3. Visualizza  
4. Salva  
5. Carica  
6. Esci  
Scegli un'opzione (1-6): 3  
  
Lista della spesa (ordinata):  
- bagnoschiuma
```

Anche queste funzioni sono svolte correttamente.

Le ultime prove da fare sono: la funzione salva, carica ed esci:

```
Scegli un'opzione (1-6): 4
Lista salvata.

1. Aggiungi
2. Rimuovi
3. Visualizza
4. Salva
5. Carica
6. Esci
Scegli un'opzione (1-6): 6
Uscita ...

(francesco@kali)-[~]
$ python3 eserciziobonus.py

1. Aggiungi
2. Rimuovi
3. Visualizza
4. Salva
5. Carica
6. Esci
Scegli un'opzione (1-6): 5
Lista caricata.

1. Aggiungi
2. Rimuovi
3. Visualizza
4. Salva
5. Carica
6. Esci
Scegli un'opzione (1-6): 3

Lista della spesa (ordinata):
- bagnoschiuma
```

In questa schermata possiamo vedere che selezionando l'opzione 4, la lista viene salvata. Per verificare che la lista sia salvata correttamente, proviamo a caricarla attraverso l'opzione 5, il programma ci dice che la lista è stata caricata.

Per verificarlo, proviamo a visualizzarla nuovamente attraverso l'opzione 3, vediamo che ci viene subito mostrato l'unico elemento che avevamo salvato nella lista precedentemente creata. Possiamo quindi dire che il programma funziona correttamente e svolge tutte le sue funzioni come richiesto dall'esercizio.