Esercizi di Laboratorio di Programmazione

Elena Buscaroli Federico Pigozzi
Gianluca Guglielmo Stefano A. Russo (supervisione)

26 Novembre 2021, Foglio 1

Versione modificata del foglio esercizi preparato da Andrea Mecchina, Michele Rispoli, Pietro Morichetti e Nicolas Solomita per il laboratorio di programmazione a.a. 2020/2021.

1 Un primo approccio a Python

Esercizio 1

Realizzare un programma con le seguenti funzioni per la manipolazione di liste:

- *stampa*, una funzione che stampa il contenuto di una lista passata come argomento;
- *statistiche*, una funzione che riceve una lista e, se è una lista di interi, ne determina la somma, la media, il minimo ed il massimo degli elementi;
- *somma_vettoriale*, una funzione che riceve in ingresso due liste, determina se sono due liste di interi, se hanno la stessa dimensione e ne calcola la somma vettoriale, poi ritornata come lista, altrimenti ritorna una lista vuota;

Testare le funzioni passandogli in input diverse liste.
Nota. Potete usare la funzione built-in *type* per verificare se una certa variabile è un (oggetto) *int* oppure no (es: `type(var int) == int` → `true`). Non serve importare alcun modulo.

2 Manipolazione dati su file

Esercizio 2

Implementare una funzione (simile a quella creata le lezioni precedenti) che ritorni una lista i cui elementi saranno le date delle vendite del file *shampoo_sales.csv*.
Attenzione: Avevate usato la funzione *float* per convertire i prezzi da stringhe a valori numerici. Questa volta dovrete usare un'altra funzione:

```
from datetime import datetime
...
my date = datetime.strptime (elements[0],'%d-%m-%Y')
```

Questa conversione vi permetterà di eseguire operazioni con le date (come trovare la data più lontana, quella più vicina, la conversione in mesi, ecc ecc..). Stampando la lista però, vederete degli oggetti di tipo DateTime. Se volete stampare le date in modo più leggibile, i comandi sono i seguenti:

```
for data in date_vendite:
    print(data.strftime('%d-%m-%Y'))
```

Esercizio 3

Estendere la classe *CSVFile* che avete creato la scorsa lezione, aggiungendo i seguenti metodi:

- *get_date_vendite()*: questa funzione ritornerà una lista con le date delle vendite. (Hint: usare la funzione creata nell'esercizio 2);
- *__str__()*: questa funzione ritornerà l'intestazione (header) del file CSV.

3 Classi e Oggetti

Esercizio 4

Implementare una classe *Automobile* che presenta i seguenti **attributi**: *casa_automo*, *modello*, *numero_posti*, *targa*.
Inoltre, la presente classe deve comprendere i seguenti **metodi**:

- *__init__*, metodo per inizializzare una istanza della classe;
- *__str__*, metodo che stampa tutte le informazioni associate ad una specifica istanza (aka oggetto) della classe *Automobile*;
- *parla*, metodo che stampa a schermo "Broom Broom";
- *confronta*, metodo che, data in ingresso un'altra istanza di *Automobile*, determina se i due oggetti hanno le stesse informazioni (eccetto per la targa che è univoca!).

Esercizio 5

Estendere l'esercizio 4 realizzando la sottoclasse *Transformer* della classe *Automobile*. La sottoclasse è caratterizzata dai seguenti **attributi**:

- *nome*, ossia il nome dell'istanza della classe *Transformer*;
- *generazione*, ossia la generazione di appartenenza del Transformer come un intero positivo (1, 2, 3, ...);
- *grado*, ossia il grado militare (es. "soldato semplice", "sergente", "capitano");
- *reparto*, ossia la divisione a cui fa parte l'istanza della classe quindi "corpo a corpo", "artiglieria leggera", "artiglieria pesante", "spionaggio", ...

E **metodi**:

- sovrascrivere il metodo *__init__* per fare in modo che accetti i nuovi attributi;
- definire il metodo *scheda_militare* che stampa a schermo le informazioni "militari" di una istanza della classe Transformer.

Utilizzare *super()* dove lo si ritiene appropriato.

Esercizio 6 - opzionale

Considerate il seguente frammento di pseudo-codice che fa riferimento alle classi sviluppate negli esercizi 4 e 5.

```
auto_0 = Automobile(...);
...
t_0 = Transformer(...);
t_1 = Transformer(...);
...
```

Rispondere ai seguenti quesiti:

1. auto_0 è di tipo Automobile?
2. Identificare la superclasse e sottoclasse tra Transformer e Automobile.
3. t_0 è di tipo Transformer?
4. t_0 e t_1 sono dello stesso tipo?
5. auto_0 e t_1 sono dello stesso tipo?
6. t_1 potrebbe essere di tipo Automobile?
7. auto_0 potrebbe essere di tipo Transformer?

Potete verificare la vostra risposta usando le funzioni booleane built-in:

- `issubclass(nome_super_classe, nome_sotto_classe)`,
- `type(nome_variabile)`,
- `isinstance(nome_variabile, nome_classe)`.