Esame Laboratorio di Programmazione I

08/06/2026

Svolgimento

Chiamate il file in cui scrivere il vostro codice `matricola.py` (la matricola è `SMnumeri` o `ECnumeri`).
All'inizio del file scrivete un commento con il vostro nome e numero di matricola.
Consegnate un codice che compila: se il codice non compila non sarà valutato.
Commentate il codice descrivendo in modo chiaro cosa state facendo.

Dataset Il file `ElectricityByBuilding.csv` contiene una time series dei consumi mensili di energia elettrica registrati per diversi edifici, espressi in kWh.
Le colonne rilevanti sono:

- `dt` (`YYYY-MM-DD`): la data della misurazione;
- `AverageConsumption`: il consumo mensile;
- `Building`: l'identificativo dell'edificio.

Il dataset può contenere dati mancanti oppure valori non numerici.

1 (10 punti) Lettura dei dati

Scrivere la classe `CSVTimeSeriesFile`:

- (2 punti) La classe deve essere istanziata con il nome del file tramite la variabile `name`.
- (8 punti) Deve avere un metodo `get_data` che prende in input il nome di un edificio e restituisce una lista di coppie `[data, consumo]`, dove il primo elemento è la data (stringa) e il secondo è il consumo mensile (float), per tutti e soli i valori validi associati all'edificio richiesto.

Esempio d'uso:

```
time_series_file = CSVTimeSeriesFile(name="ElectricityByBuilding.csv")
time_series_A =   time_series_file.get_data("A")
```

Output atteso:

```
[
    ["2019-01-01", 383.7],
    ["2019-02-01", 407.7],
    ["2019-03-01", 413.5],
    ...
]
```

2 (12 punti) Calcolo delle variazioni tra due serie temporali

Definire la funzione `compute_variations`, **NON** nella classe `CSVTimeSeriesFile` ma direttamente nel corpo principale del programma, che prende in input due serie temporali e un intervallo di anni e restituisce un dizionario contenente la differenza tra i consumi medi annuali delle due serie:

```
compute_variations(time_series_1, time_series_2, first_year, last_year)
```

Attenzione che possono esserci anni interi senza valori per un edificio e non per l'altro. Se ad una serie temporale manca un anno dell'intervallo, quell'anno deve essere ignorato nel calcolo delle differenze.

Nota Questo punto è indipendente dal precedente: la funzione può funzionare su due generiche serie temporali nel formato richiesto (liste di coppie `[data, valore]`), anche se tali serie non sono necessariamente prodotte dal metodo `get_data`.

La funzione deve:

- (3 punti) estrarre i consumi per ciascun anno;
- (3 punti) calcolare il consumo medio annuale per ogni anno nell'intervallo dato;
- (4 punti) calcolare la differenza tra le medie annuali delle due serie temporali mantenendo l'ordine: `differenza = media_serie_2 - media_serie_1`
- (2 punti) restituire un dizionario contenente le differenze. Le differenze devono essere arrotondate a 2 cifre decimali (ad esempio utilizzando la funzione `round` di Python).

Hint Potete costruirvi una funzione ausiliaria che calcoli le medie annuali di una serie temporale di modo da non dover scrivere due volte lo stesso codice.

Esempio d'uso:

```
time_series_file = CSVTimeSeriesFile(name="ElectricityByBuilding.csv")

time_series_A = time_series_file.get_data("A")
time_series_B = time_series_file.get_data("B")

compute_variations(time_series_A, time_series_B, 2019, 2022)
```

Output atteso:

```
{
    "2019": 88.38,
    "2020": 98.98,
    "2021": 87.2,
    "2022": 84.13
}
```

3 (8 punti) Validazione input ed eccezioni

Definire la classe:

```
class ExamException(Exception):
    pass
```

e usarla per gestire i seguenti casi:

- (3 punti) Se il nome dell'edificio dato in input non è presente nel file, si deve alzare un'eccezione che dice `"Errore: il nome dell'edificio non è presente nel file"`.
- (2 punti) Gli anni passati alla funzione devono essere interi (di tipo `int`), altrimenti alzare un'eccezione che dice `"Errore: l'anno inserito non è un intero"`.
- (3 punti) L'intervallo selezionato deve contenere almeno un valore valido, altrimenti alzare un'eccezione che dice `"Errore: l'intervallo selezionato non contiene valori validi"`.

I valori letti dal file devono essere numerici; valori non numerici o mancanti devono essere ignorati senza alzare eccezioni.