Laboratorio di Programmazione

Esercitazione di esame 2

Una *time series* (univariata) è una sequenza di coppie di valori ordinate temporalmente, dove il primo elemento della coppia corrisponde al tempo, mentre il secondo è il valore di una qualche quantità atta a descrivere un certo fenomeno.

Il file "data.csv" contiene la time series del numero totale mensile in migliaia di passeggeri su linee aeree internazionali, da Gennaio 1949 a Dicembre 1960. Il primo elemento di ogni riga rappresenta la data ed è in formato `YYYY-MM`. Il dato si presenta così:

```
date,passengers
1949-01,112
1949-02,118
1949-03,132
...
```

Vogliamo leggere questo tipo di dati e calcolare, dato un intervallo di anni, la differenza tra il numero medio di passeggeri in un anno rispetto all'anno precedente. Dobbiamo quindi (1) Raggruppare la lista di passeggeri per ogni anno nell'intervallo (2) Calcolare, per ogni anno nell'intervallo, il numero medio di passeggeri, (3) calcolare e ritornare le differenze per ogni coppia di anni nell'intervallo.
Esempio: intervallo 1949-1951. Raggruppiamo gli elementi:

1949: [112,118,132,…]
1950: [115,126,141,…]
1951: [145,150,178,…]

Calcoliamo il numero medio (per l'esempio consideriamo solo i primi tre mesi, ma la media andrà calcolata su tutti i mesi ovviamente):

$$1949: \frac{112+118+132}{3} = 120.7$$

$$1950: \frac{115+126+141}{3} = 127.3$$

$$1951: \frac{145+150+178}{3} = 157.7$$

ed infine sottraiamo il valore dell'anno corrente e quello precedente, tornando in questo caso i valori 127.3 - 120.7 e 157.7 - 127.3, che dovranno essere opportunamente strutturati come da specifiche nella sezione seguente "informazioni sullo svolgimento".

Informazioni sullo svolgimento

Create la classe `CSVTimeSeriesFile`. La classe deve essere istanziata sul nome del file tramite la variabile `name` e deve avere un metodo `get_data()` che torni una lista di liste, dove il primo elemento delle liste annidate è la data ed il secondo il numero mensile di passeggeri. Questa classe si dovrà quindi poter usare così:

```
time_series_file = CSVTimeSeriesFile(name='data.csv')
time_series = time_series_file.get_data()
```

...ed il contenuto della variabile `time_series` tornato dal metodo `get_data()` dovrà essere così strutturato (come lista di liste):

```
[
  [“1949-01”, 112],
  [“1949-02”, 118],
  [“1949-03”, 132],
  ...
]
```

Per calcorare la differenza negli anni nel numero medio di passeggeri, dovete creare una funzione a sé stante (definita NON nella classe `CSVTimeSeriesFile` ma direttamente nel corpo principale del programma), di nome `compute_variations`, che avrà come input la time series e che verrà usata così:

```
compute_variations(time_series, first_year, last_year)
```

dove `first_year` e `last_year` corrispondono rispettivamente agli estremi dell’intervallo di anni che vogliamo considerare (entrambi gli estremi devono essere inclusi). La funzione dovrà ritornare (tramite un `return`) in output un dizionario, dove le chiavi saranno gli intervalli dei due anni presi in considerazione e i valori l’incremento del numero medio di passeggeri rispetto all’anno precedente.

```
{
  “1949-1950”: 13,
  “1950-1951”: 30.5,
  ...
}
```

Chiamate il file in cui scrivere il vostro codice **"esame_matricola.py"** e le eccezioni da alzare in caso di input non corretti o casi limite devono essere istanze di una specifica classe `ExamException`, che dovete definire nel codice come segue, senza modifica alcuna (copia-incollate le due righe):

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

...e che poi userete come una normale eccezione, ad esempio:

```
raise ExamException('Errore, lista valori vuota')
```

Commentato il vostro codice in maniera da renderlo chiaro.

Qualche informazione in più sulle specifiche e qualche e suggerimento:

- Nelle misurazioni di un anno, può mancare il numero di passeggeri per uno o più mesi. Per un anno è possibile che non venga registrato il numero dei passeggeri per NESSUN mese. In tal caso, se l’anno è compreso nell’intervallo di anni dato come input, l’anno senza misurazioni andrà ignorato, quindi se per esempio non ho i dati del 1954 e studio l’intervallo 1952-1955 avrò:

  ```
  {
    “1952-1953”: 28,
    “1953-1955”: 59
  }
  ```

- Se mancano misurazioni per uno o più mesi, chiaramente il numero medio di passeggeri andrà calcolato sul numero di mesi per cui abbiamo le misurazioni.

- I valori che leggete dal file CSV sono da aspettarsi di tipo intero, un valore non numerico, oppure vuoto o nullo o negativo non deve essere accettato, ma tutto deve procedere comunque senza alzare eccezioni.

- La classe `CSVTimeSeriesFile` controlla l'esistenza del file solo quando viene chiamato il metodo `get_data()` e, nel caso il file non esista o non sia leggibile, alza un'eccezione.

Gli unici controlli obbligatori che dovete inserire nel codice sono quindi:

1. **Controllare** che il file sia apribile.
2. **Verificare** che il valore di passengers sia un intero (in caso contrario, ignorare la riga e stampare un messaggio).

Parte opzionale (per la Lode):

Altre aggiunte per rendere più generale il codice, fare almeno uno dei punti qui sotto e spiegarlo bene con un commento.

- Gli estremi dell'intervallo di tempo dati come input alla funzione devono essere validi: per esempio se vogliamo considerare un intervallo di tempo dal 1950 al 1952 chiameremo la funzione compute_variations(time_series, "1950", "1952"), dove first_year e last_year devono essere passati alla funzione come stringhe e devono essere presenti nei dati, altrimenti va alzata un'eccezione.
- La serie temporale nel file CSV è da considerare sempre ordinata, se per caso ci dovesse essere un timestamp fuori ordine va alzata un'eccezione (dentro la funzione `get_data()`) senza cercare di riordinare la serie. Stesso discorso se c'è un timestamp duplicato: si alza un'eccezione.
- In generale, il file CSV può contenere letteralmente di tutto. Da linee incomplete a pezzi di testo che non c'entrano niente, e ogni errore *salvo quello di un timestamp fuori ordine o duplicato* va ignorato (ovvero, ignoro la riga contenente l'errore e vado a quella dopo). Nota: se riuscite a leggere due valori (data e numero di passeggeri) ma c'è un campo di troppo sulla stessa riga, questo non è da considerarsi un'errore e non bisogna ignorare quella riga.