Laboratorio di Programmazione

Compito d'esame per l'appello del 1/3/2024

Una time series (univariata) è una sequenza di valori ordinati nel tempo, ed esprime la variazione di un certo fenomeno nel tempo. Per ogni coppia di punti, il primo elemento corrisponde a un istante o intervallo di tempo, mentre il secondo è il valore di una qualche quantità relativa a quell'istante o intervallo.
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 Anno-Mese. Il dato si presenta così:

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

ovvero, messa sotto forma di tabella per comodità:

date	passengers
1949-1	112
1949-2	118
1943-3	132
…	…

Vogliamo leggere questo tipo di dati e trovare per ogni anno il mese/i con il numero minimo e massimo di passeggeri.
Quindi, per ogni anno andranno rilevati mesi con valore minimo e massimo di passeggeri e andranno poi tornati da una funzione. Se per un anno sono presenti due o più mesi con stesso numero massimo/minimo di passeggeri, allora andranno ritornati tutti i mesi in questione.

Informazioni sullo svolgimento

Dovete tenere in considerazione che possono esserci dei dati mancanti! Nelle misurazioni di un anno, può mancare il numero di passeggeri per uno o più mesi. Inoltre, possono mancare misurazioni per uno o più anni interi. mancare misurazioni per uno o più anni interi.
Alla luce di tutto questo, create la classe `CSVTimeSeriesFile`, modificando o estendendo la classe `CSVFile` vista a lezione (oppure scrivendola da zero). 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 di passeggeri, in formato numerico.

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 rilevare i mesi con il numero mensile minimo e massimo di passeggeri, dovete creare una funzione a sé stante (definita NON nella classe `CSVTimeSeriesFile` ma direttamente nel corpo principale del programma), di nome `find_min_max`, che avrà come input la time series e che verrà usata così: find_min_max, che avrà come input la time series e che verrà usata così:

```
	find_min_max(time_series)
```

La funzione dovrà ritornare (tramite un `return`) in output un dizionario di dizionari, dove le chiavi del dizionario più esterno saranno gli anni considerati (come stringa) ed il valore per ogni anno sarà un altro dizionario con due

chiavi (`‘min’` e `‘max’`) che avranno come valori delle liste che conterranno i mesi (sempre in formato stringa!) con i valori minimi e massimi di passeggeri per quel determinato anno:

```
{“1951”: {“min”:[” 03”], “max”:[“01”,” 07]”},
 “1952”: {}, ...,}
```

Il file in cui scrivere il vostro codice deve chiamarsi "esame.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')
```

Qualche informazione in più sulle specifiche e qualche e suggerimento:

- Attenzione che, come accennato sopra, nel Attenzione che, come accennato sopra, nel file CSV possiamo avere misurazioni mancanti per uno o più mesi, o anche per uno o più anni. Nel caso di presenza di un solo mese la funzione dovrà ritornare un dizionario vuoto.

- Come accennato possiamo anche avere anni con più mesi aventi lo stesso numero minimo o massimo di passeggeri. Per esempio, per l’anno 1949 possiamo avere Marzo e Giugno con il numero minimo di passeggeri e Agosto con il numero massimo, l'output della funzione dovrà contenere come minimo i mesi di Marzo e Giugno con i mesi ordinati temporalmente e come massimo il solo mese di Agosto.
- I valori di 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 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 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.

- 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.
- 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

Informazioni sulla consegna

La consegna dell'esame deve avvenire tassativamente entro l'ora di inizio dell'appello orale, e può avvenire in due modi: allegando lo script esame.py, oppure indicando il esame.py, oppure indicando il commit hash da valutare su un repository GitHub, entrambi mandati via mail a stefanoalberto.russo@phd.units.it, come descritto più in dettaglio sul repository del corso: https://github.com/sarusso/ProgrammingLab

Attenzione: se il file non si chiama "esame.py", se le eccezioni alzate in caso di errori non sono di tipo "ExamException" o se le classi ed i metodi non si chiamano come indicato da specifiche, l'esame non potrà essere valutato!

Nota benissimo: il vostro file deve includere SOLO una classe, una funzione ed una eccezione, non deve includere nessun codice "main" o tantomeno chiedere input all'utente.

Se non vi ricordate bene come si usa Git, potete fare riferimento al tutorial dei tutor che è disponibile qui: https://github.com/drpOpZ/proglab2021-tutors/blob/master/git_quickstart.md