# Laboratorio di Programmazione

## Compito d'esame per l'appello del 10/2/2022

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 `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 |
|---|---|
| 1949-01 | 112 |
| 1949-02 | 118 |
| 1949-03 | 132 |
| ... | ... |

**Vogliamo leggere questo tipo di dati e calcolare, per un dato intervallo di anni, la variazione media nel numero di passeggeri per ogni mese.**

Per esempio, prendiamo in considerazione i primi tre anni (1949, 1950, 1951). Possiamo creare una lista di 12 elementi per ogni anno, dove l'elemento all'indice `i` corrisponde al numero di passeggeri per il mese `i+1` in quell'anno (all'indice 0 avremo il valore per il mese 1, Gennaio):

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

Per ottenere l'incremento medio per ogni mese bisognerà quindi calcolare la differenza tra il numero di passeggeri di un anno e dell'anno precedente, mese per mese, e fare la media tra queste differenze, ottenendo una lista come la seguente:

$$\left[\frac{(115-112)+(145-115)}{2}, \frac{(126-118)+(150-126)}{2}, \frac{(141-132)+(178-141)}{2}, \ldots\right] \rightarrow [16.25, 16, 23, \ldots]$$

Dove 16.25 è la variazione media di Gennaio, 16 quella di Febbraio, 23 quella di Marzo, e così via.

# Informazioni sullo svolgimento

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.

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 calcolare la differenza media del numero di passeggeri mensile tra anni consecutivi, dovete creare una funzione a sé stante (definita cioè NON nella classe `CSVTimeSeriesFile` ma al suo stesso "livello" nel file), di nome `compute_avg_monthly_difference`, che verrà usata così:

```
compute_avg_monthly_difference(time_series, first_year, last_year)
```

dove il primo input è la time series da processare, mentre `first_year` e `last_year` corrispondono rispettivamente agli estremi dell'intervallo di anni che vogliamo considerare. Entrambi gli estremi, quindi sia il primo che l'ultimo anno dell'intervallo, devono essere compresi nella valutazione. La funzione dovrà ritornare (tramite un `return`) in output una lista di 12 elementi, dove ogni elemento rappresenta la differenza media nel numero mensile di passeggeri tra un anno e il seguente, ad esempio:

```
[
  16.25,
  16,
  23,
  ...
]
```

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:

- 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, ma assumiamo di avere almeno una misurazione all'anno.
- Nel caso in cui per un mese di un anno non ci siano misurazioni, decidiamo semplicemente di non considerare quel mese di quell'anno nel calcolo della differenza media. Quindi:
  - se consideriamo un intervallo di due anni, per il mese con la misurazione mancante verrà tornato come valore finale 0;
  - se consideriamo un intervallo di più di due anni e per un mese abbiamo meno di due misurazioni, verrà tornato 0 come valore finale per quel mese;
  - se consideriamo un intervallo di più di due anni, calcoliamo la differenza media tra le misurazioni di quel mese per gli altri anni, ignorando la misurazione mancante.
- Attenzione che gli estremi dell'intervallo di tempo dati come input alla funzione devono essere validi valori: per esempio se vogliamo considerare un intervallo di tempo dal 1950 al 1952 chiameremo la funzione `compute_avg_monthly_difference(time_series, "1950", "1952")`, dove `first_year` e `last_year` devono essere passati alla funzione come stringhe e devono essere presenti nel file CSV, altrimenti va alzata un'eccezione.
- I valori di che leggete dal file CSV sono da aspettarsi di tipo intero positivo. 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 `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 commit hash da valutare su un repository GitHub, entrambi mandati via mail a stefano.russo@gmail.com, 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!*

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.