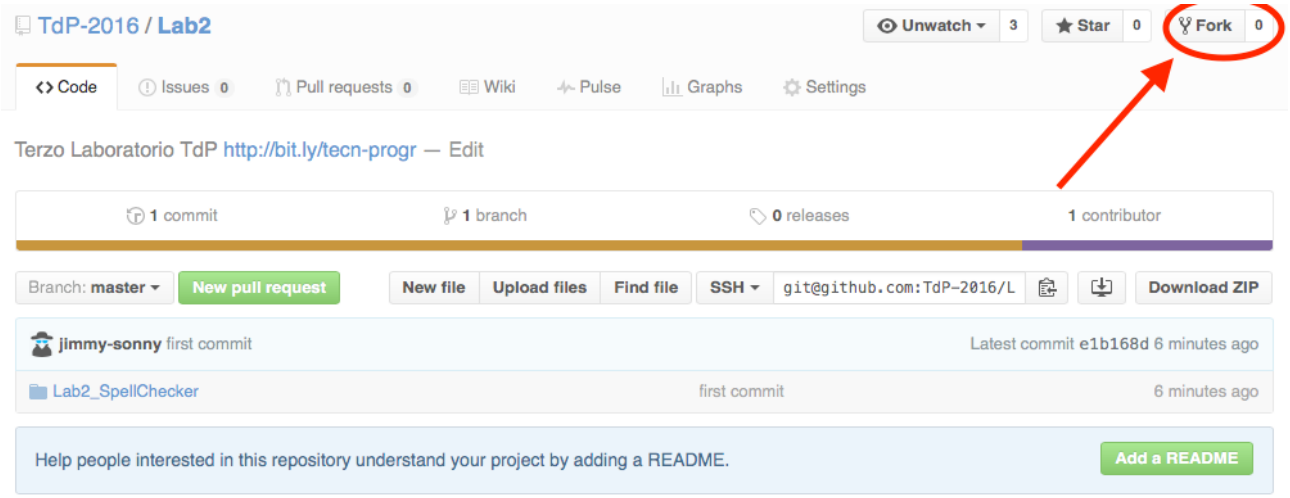
03FYZ TECNICHE DI PROGRAMMAZIONE
Istruzioni per effettuare il fork di un repository GitHub

- Effettuare il login su GitHub utilizzando il proprio username e password.
- Aprire il repository su GitHub relativo al nono laboratorio:

  https://github.com/TdP-2016/Lab8
- Utilizzare il pulsante *Fork* in alto a destra per creare una propria copia del progetto.



L'azione di Fork crea un nuovo repository nel proprio accout GitHub con una copia dei file necessari per l'esecuzione del laboratorio.

- Aprire Eclipse, andare su *File -> Import*. Digitare *Git* e selezionare *Projects from Git -> Next -> Clone URI -> Next*.
- Utilizzare la URL del **proprio** repository che si vuole clonare (**non** quello in TdP-2016!), ad esempio:

  https://github.com/my-github-username/Lab8
- Fare click su *Next*. Selezionare il branch (*master* è quello di default) fare click su *Next*.
- Selezionare la cartella di destinazione (quella proposta va bene), fare click su *Next*.
- Selezionare *Import existing Eclipse projects*, fare click su *Next* e successivamente su *Finish*.
- Il nuovo progetto Eclipse è stato clonato ed è possibile iniziare a lavorare.
- A fine lavoro ricordarsi di effettuare Git commit e push, utilizzando il menù *Team in Eclipse*.

ATTENZIONE: solo se si effettua Git **commit** e successivamente Git **push** le modifiche locali saranno propagate sui server GitHub e saranno quindi accessibili da altri PC e dagli utenti che ne hanno visibilità.

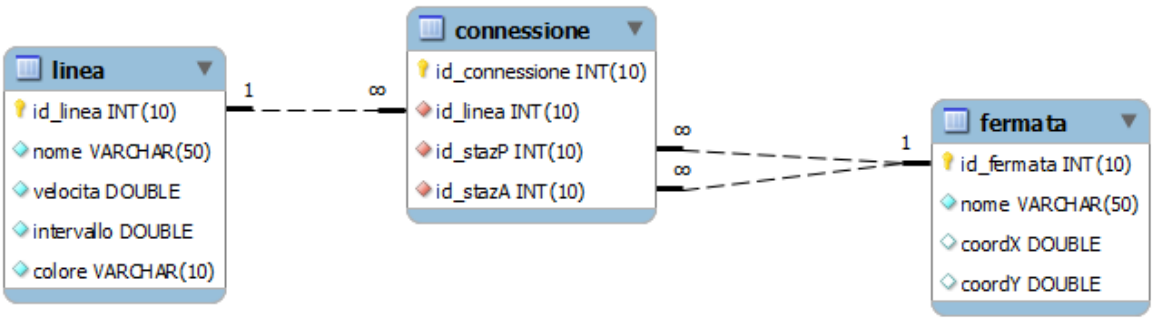
03FYZ TECNICHE DI PROGRAMMAZIONE
Esercitazione di Laboratorio 08 – 4 maggio 2016

Obiettivi dell'esercitazione:

- Esercizi sui Grafi Pesati
- Utilizzo della libreria JGraphT
- Cammini minimi

Si consideri la rete del sistema metropolitano della città di Parigi [GitHub - DB/metro-Paris.pdf]. Tale rete è codificata nel data-set **metroparis.sql** [GitHub - DB/metroparis.sql]. In *Fig 1* è mostrata la struttura delle tabelle.

Fig 1



ESERCIZIO 1:

Scopo dell'esercitazione: Realizzare un'applicazione in JavaFX che permetta di calcolare il tempo di percorrenza tra due stazioni della rete metropolitan di Parigi.

Funzionamento previsto dall'applicazione: L'utente seleziona da un menù a tendina una stazione di partenza ed una di arrivo. Facendo click sul pulsante "*Calcola percorso*", il programma visualizza in un'area di testo sottostante le fermate intermedie ed il tempo di percorrenza totale.

Esercizio 1.1 Realizzare un'interfaccia grafica con *JavaFx* simile a quella mostrata in *Fig 2*. Strutturare il programma secondo i pattern MVC e DAO come spiegato a lezione.

Esercizio 1.2 Utilizzare il file **metroparis.sql** [GitHub - DB/metroparis.sql] per ottenere le informazioni su linee, connessioni e fermate del sistema metropolitano di Parigi. Fare uso di tutti gli elementi della tabella *fermata* per popolare i due menù a tendina.

Esercizio 1.3 Utilizzando la libreria *JGraphT* creare un grafo non orientato pesato in cui ogni stazione rappresenta un vertice. Un arco collega due stazioni vicine (utilizzare le informazioni della tabella *connessione*). Il peso dell'arco è dato dal tempo di percorrenza tra le due stazioni. Utilizzare l'equazione del moto rettilineo uniforme per calcolarla.

Consiglio #1: per calcolare la distanza è possibile utilizzare la libreria **simplelatlng-1.3.0.jar** già inclusa nel progetto, facendo uso della seguente istruzione:

```
distance(new LatLng(x1, y1), new LatLng(x2, y2), LengthUnit.KILOMETER)
```

Esercizio 1.4 Implementare le funzionalità "*Calcola percorso*" calcolando il cammino minimo tra la stazione di partenza e di arrivo selezionate. Visualizzare tutte le fermate intermedie ed il tempo di percorso totale (assumendo una sosta di 30 secondi per ogni fermata).

Fig 2

ESERCIZIO 2:

Scopo dell'esercitazione:
Migiorare l'applicazione realizzata nel primo esercizio considerando anche il tempo di attesa nel caso in cui si debba cambiare linea. Per semplicità considerare sempre il caso peggiore in cui si attende l'intero *intervallo* indicato nella tabella *Linea*.

Esercizio 2.1 Utilizzando la libreria *JGraphT* creare un grafo orientato pesato in cui ogni stazione rappresenta un vertice ed un arco orientato collega due stazioni vicine (utilizzare le informazioni della tabella *connessione*).

Esercizio 2.2 Implementare le funzionalità "*Calcola percorso*" calcolando il cammino minimo tra la stazione di partenza e di arrivo selezionate. Visualizzare tutte le fermate intermedie ed il tempo di percorso totale (assumendo una sosta di 30 secondi per ogni fermata).
Nel caso in cui sia necessario, indicare il cambio della linea. (Esempio in *Fig 3*)

Consiglio #2: Per la creazione del nuovo grafo fare riferimento al disegno in *Fig 4*. Nel caso in cui una la stazione sia un punto di collegamento tra più linee, bisogna inserire nel grafo più vertici, uno relativo ad ogni linea della stazione. Gli archi orientati che collegano la stessa stazione su diverse linee sono i tempi di attesa (*intervallo*). Gli archi orientati tra stazioni della stessa linea sono i tempi di collegamento, come nel caso dell'esercizio precedente.

Fig 3

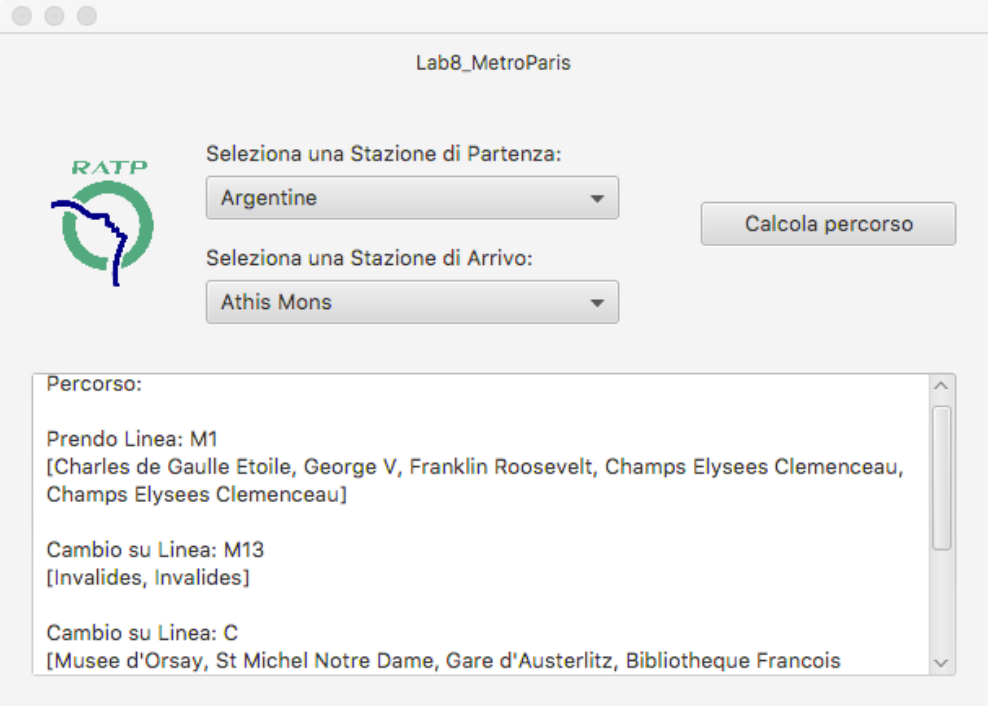
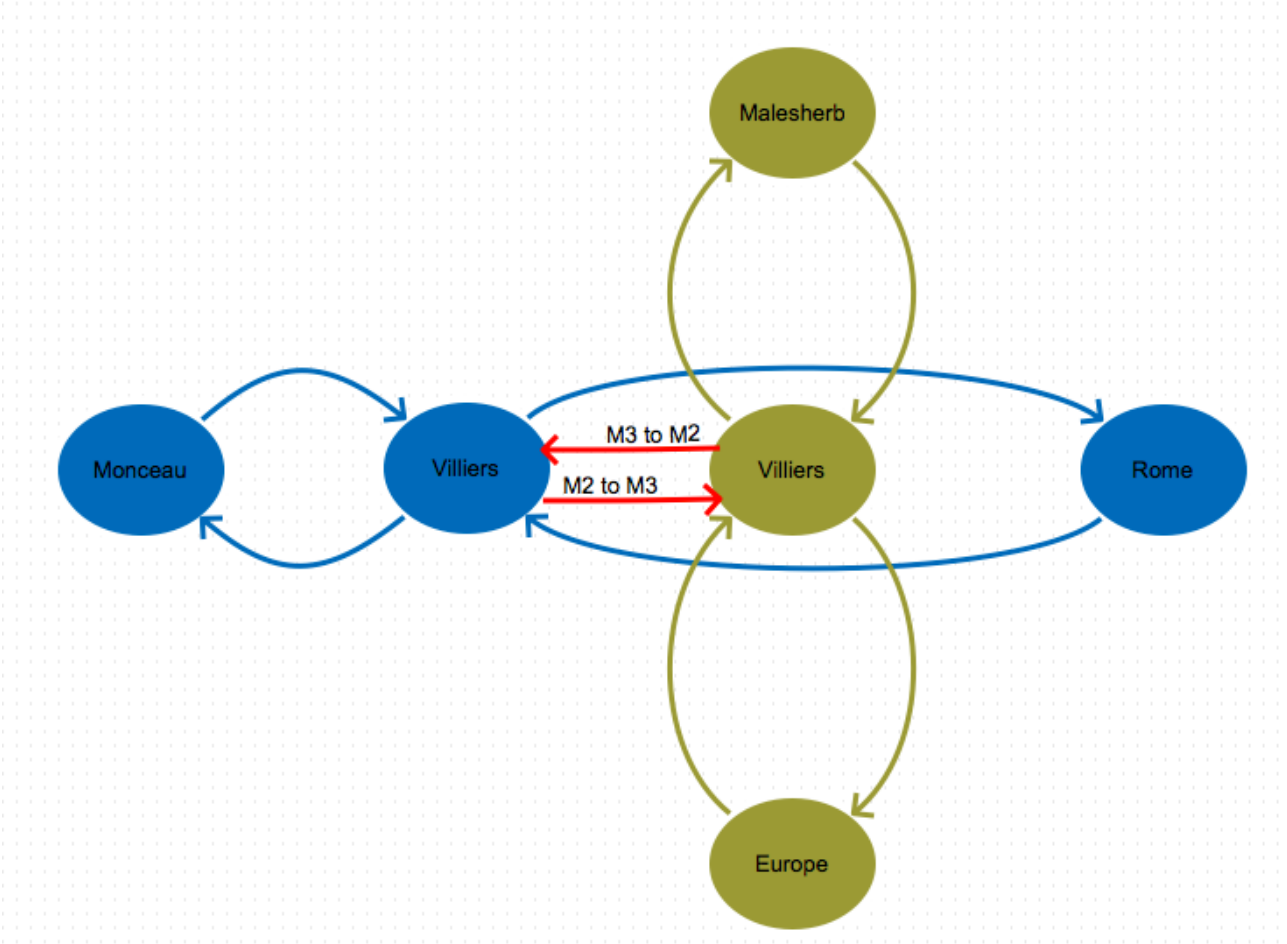

Lab8_MetroParis
RATP
Seleziona una Stazione di Partenza:
Argentine
Seleziona una Stazione di Arrivo:
Athis Mons
Calcola percorso
Percorso:
Prendo Linea: M1
[Charles de Gaulle Etoile, George V, Franklin Roosevelt, Champs Elysees Clemenceau, Champs Elysees Clemenceau]
Cambio su Linea: M13
[Invalides, Invalides]
Cambio su Linea: C
[Musee d'Orsay, St Michel Notre Dame, Gare d'Austerlitz, Bibliotheque Francois

Fig 4


Malesherb
Monceau
Villiers
M3 to M2
M2 to M3
Villiers
Rome
Europe