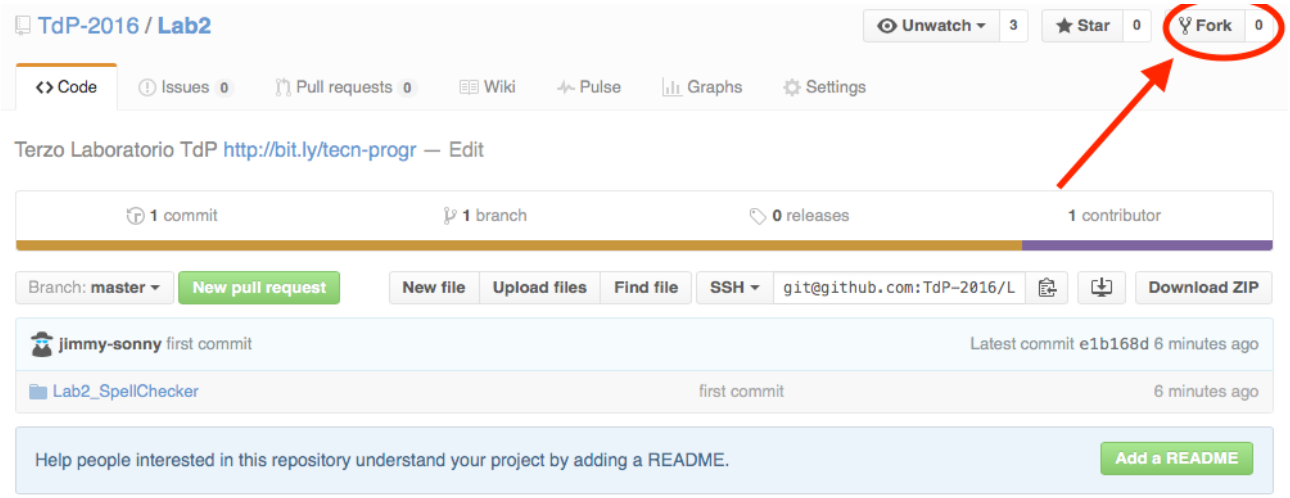
03FYZ TECNICHE DI PROGRAMMAZIONE
Istruzioni per effettuare il fork di un repository GitHub

- Effettuare il login su GitHub utilizzando il proprio username e password.
- Aprire il repository su GitHub relativo all'ottavo laboratorio:

  https://github.com/TdP-2017/Lab08

  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-2017!), ad esempio:

  https://github.com/my-github-username/Lab08

  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 – 3 maggio 2017

Obiettivi dell'esercitazione:

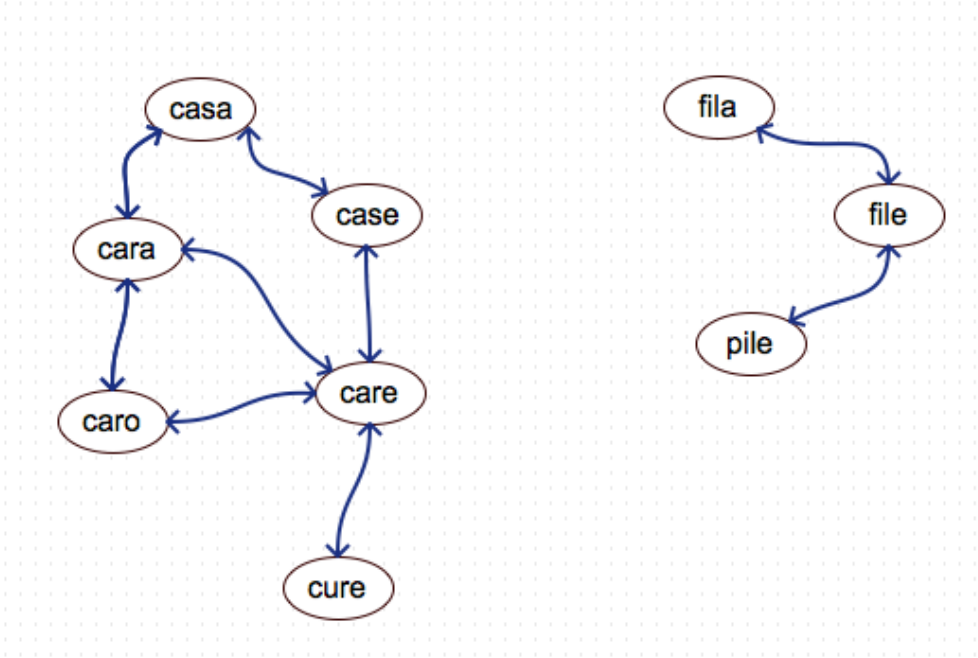
- Introduzione ai Grafi
- Utilizzo della libreria JGraphT

ESERCIZIO 1

Partendo dal programma svolto nel settimo laboratorio (Lab07), aggiungere la funzionalità "trova tutti i vicini" a partire dalla parola inserita. Modificare l'interfaccia grafica, inserendo un nuovo *Button*, la classe *Controller* e *Model* per supportare la nuova funzionalità. Facendo click sul bottone *TrovaTuttiVicini* viene restituita la lista di tutti i nodi raggiungibili nel grafo a partire dal vertice selezionato, che coincide con la componente connessa del grafo relativa alla parola scelta.

Esempio:
Dato il seguente dizionario di parole di **4 lettere**: [casa, case, cara, care, caro, cure, fila, pila, pile]
si ottiene il grafo in *Fig 1*.

Fig 1



- Inserendo la parola **casa**, si ottiene la seguente lista di nodi raggiungibili:
  [cara, case, care, caro, cure]

- Inserendo la parola **fila**, si ottiene la seguente lista di nodi raggiungibili:
  [file, pile]

Continua nella pagina seguente

Nota: per trovare la lista di tutti i vicini di un vertice (quindi la componente connessa relativa al vertice selezionato) è necessario effettuare una visita del grafo utilizzando una delle seguenti tecniche:

- Sfruttando i metodi esposti dalla libreria JGraphT (*BreadthFirstIterator*, *DeepFirstIterator*)
- Implementando manualmente un algoritmo ricorsivo (per la visita in profondità)
- Implementando manualmente un algoritmo iterativo

La versione iterativa utilizza due liste, quella dei nodi *Visitati* e quella dei nodi *daVisitare*. Si inizia inserendo la parola scelta nella lista *daVisitare*. L'algoritmo continua fino a quando la lista dei nodi *daVisitare* non si svuota. Ad ogni passo si estrae un nodo dalla lista *daVisitare* e si inseriscono tutti i nodi vicini a quello estratto (a meno di quelli già visitati) nella lista *daVisitare*. Infine, il nodo estratto viene inserito nella lista dei *Visitati*.

Si consiglia di provare almeno due metodi alternativi.

ESERCIZIO 2

Dopo aver effettuato il fork del progetto relativo all'ottavo laboratorio (Lab08), scrivere un'applicazione JavaFX che permetta di interrogare il database *countries.sql* e restituire informazioni sui confini via terra tra stati. L'interfaccia grafica, *Fig.2*, viene fornita nel progetto di base.

L'applicazione dovrà svolgere le seguenti funzioni:

a. Permettere all'utente di inserire un anno (nel range 1816 - 2016) nell'apposita casella di testo e di richiedere il calcolo dei confini facendo click sul bottone *Calcola confini*

b. Creare un grafo che rappresenti la situazione dei confini mondiali nell'anno specificato dall'utente. Le relazioni di confine da considerare sono tutte quelle in cui campo 'year' sia minore o uguale dell'anno specificato. I vertici del grafo rappresentano le nazioni presenti in quell'anno, e gli archi (non orientati e non pesati) rappresentano un confine via terra ('conttype'=1)

c. Stampare l'elenco degli stati, indicando per ciascuno il numero di stati confinanti

d. Stampare il numero di componenti connesse nel grafo.

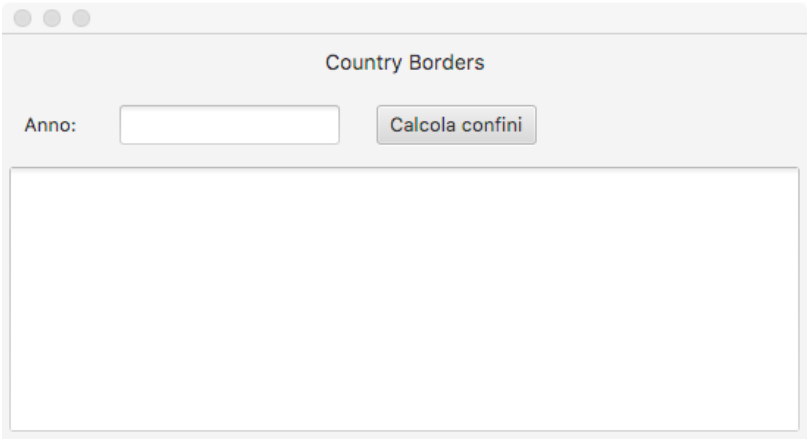
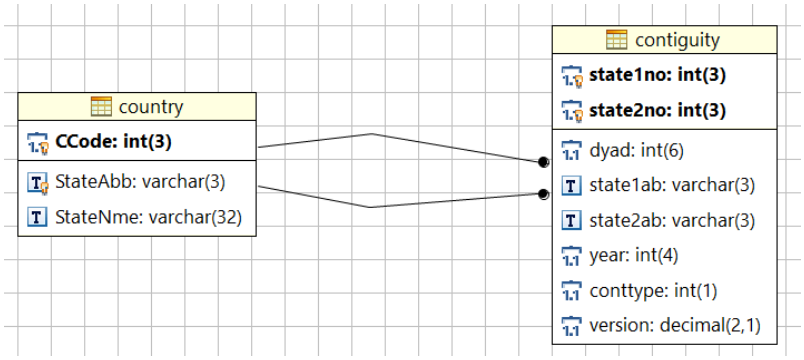


Fig. 2

Continua nella pagina seguente

Descrizione del database "country borders":



La tabella *country* riporta la lista di tutte le nazioni, ciascuna identificata da un numero, da un'abbreviazione univoca di 3 lettere e dal nome completo. La tabella *contiguity* rappresenta la presenza di un confine, per ogni coppia di stati ('state1no', 'state2no'), a partire dall'anno 'year' (anni compresi tra il 1816 ed il 2006). Esistono tipi diversi di confine, ma considerare esclusivamente il confine via terra ('conttype'=1).
[Nota: la tabella *contiguity2006* non deve essere utilizzata]

I dati sono estratti dal progetto "The Correlates of War" http://www.correlatesofwar.org/

Nota: per trovare il numero di componenti connesse di un grafo si può sfruttare la classe di JGraphT *ConnectivityInspector*. http://jgrapht.org/javadoc/org/jgrapht/alg/ConnectivityInspector.html