

# 03FYZ TECNICHE DI PROGRAMMAZIONE

## Esercitazione di Laboratorio 03 es.1 – 23 Marzo 2016

---

Obiettivi dell'esercitazione:

- Utilizzo del pattern MVC
  - Utilizzo di HashSet
  - Introduzione alla complessità
- 

### ESERCIZIO 1

Scopo dell'esercitazione: Facendo riferimento all'applicazione del Laboratorio 2, confrontare le prestazioni di ricerca utilizzando diverse strutture dati: HashSet, ricerca su Lista, query su database.

Funzionamento previsto per la versione finale (completa) dell'applicazione: fare riferimento al testo del Laboratorio 2 <http://goo.gl/DTHdpK>

**Esercizio 1.1** Partendo dall'applicazione sviluppata nel Laboratorio 2, utilizzare come struttura dati per memorizzare le parole un *HashSet*. Confrontare le prestazioni di ricerca con le strutture dati utilizzati nello scorso laboratorio.

**Esercizio 1.2** Modificare il dizionario italiano, utilizzando il database "dizionario.sql".

Per importare il database, avviare il database locale *XAMPP*; successivamente lanciare il programma *HeidiSQL* e selezionare l'opzione *File -> Carica file SQL*. Selezionare il file *dizionario.sql* (scaricato dal sito del corso, sezione Materiale, Data-Sets), eseguire la query direttamente (senza caricare nell'editor i dati) per importarlo e cliccare sul tasto *Aggiorna*, per permettere la corretta visualizzazione all'interno dell'elenco del database appena caricato.

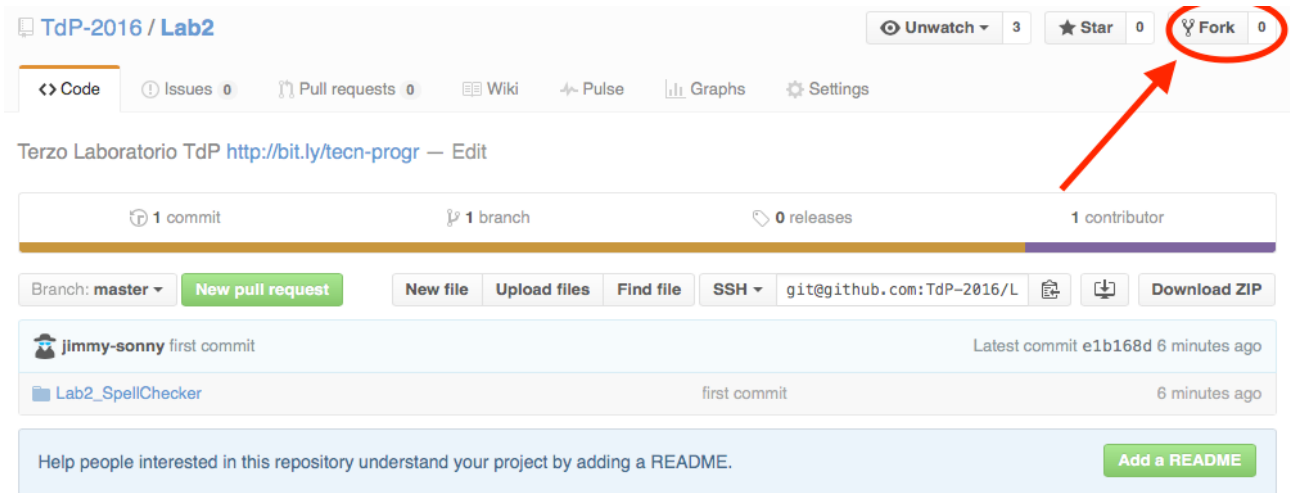
Utilizzare una query per salvare il contenuto dell'intero dizionario all'interno di una lista in Java. Effettuare la ricerca dicotomica sulla lista.

**Esercizio 1.3** Modificare la funzione di ricerca delle parole, facendo delle query direttamente sul dizionario. Confrontare le prestazioni con i metodi precedenti.

## 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 terzo laboratorio:  
<https://github.com/TdP-2016/Lab3>
- 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 account 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/Lab3>
- 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 03 es.2 – 23 Marzo 2016

---

### Obiettivi dell'esercitazione:

- Utilizzo del pattern MVC
  - Introduzione a JDBC
  - Introduzione al Pattern DAO
- 

### **ESERCIZIO 2**

Scopo dell'esercitazione: Realizzare in linguaggio Java un'applicazione dotata di interfaccia grafica che permetta la gestione degli studenti e dei corsi di ateneo.

Funzionamento previsto nella versione finale: L'applicazione permette di *cercare* informazioni relative ad un corso e ad uno studente. Inoltre consente di *iscrivere* uno studente già presente nel database ad un corso. Il menù a tendina dell'interfaccia grafica contiene l'elenco dei corsi erogati dall'ateneo. Un campo vuoto permette di *non* selezionare un corso specifico. Fornire le seguenti funzionalità:

- Data la matricola di uno studente, utilizzare il pulsante adiacente (icona verde) per completare in automatico il nome e cognome di uno studente. (Esempio Fig. 1)
- Selezionato un corso, facendo click sul pulsante *Cerca*, vengono visualizzati tutti gli studenti iscritti a quel corso. (Esempio Fig. 2)
- Data la matricola di uno studente (nessun corso selezionato), facendo click sul pulsante *Cerca*, controllare se è presente nel database ed in caso visualizzare tutti i corsi ai quali lo studente è iscritto. Se non è presente, visualizzare un messaggio di errore. (Esempio Fig. 3)
- Selezionato un corso ed inserita la matricola di uno studente, facendo click su *Cerca*, è possibile sapere se lo studente specificato è iscritto a quel corso. (Esempio Fig. 4)
- Selezionato un corso ed inserita la matricola di uno studente, facendo click sul pulsante *Iscrivi*, è possibile iscrivere lo studente a quel corso. (Esempio Fig. 5)
- Utilizzare il pulsante *Reset* per cancellare il contenuto di tutti i campi di ricerca.

- Esercizio 2.1** Realizzare un'interfaccia grafica con *JavaFx* simile al mockup mostrato in Fig 1. I campi *Nome* e *Cognome* sono dei *TextField* non editabili. Il pulsante *Reset* permette di cancellare il contenuto di tutti i campi.
- Dopo aver caricato il database *iscritticorsi.sql*, popolare il menù a tendina con tutti i corsi disponibili. Inseire anche un campo vuoto da utilizzare quando non si vuole selezionare nessun corso in particolare.
- Fare uso dei pattern MVC e DAO, come spiegato a lezione.
- Esercizio 2.2** Implementare la funzionalità di completamento automatico: data la matricola di uno studente, facendo click sul pulsante adiacente, completare in automatico il nome e cognome di uno studente. (Esempio Fig. 1)
- Esercizio 2.3** Implementare la ricerca degli studenti iscritti ad un corso: selezionato un corso dal menù a tendina, facendo click sul pulsante *Cerca*, vengono visualizzati tutti gli studenti iscritti a quel corso. (Esempio Fig. 2)
- Esercizio 2.4** Implementare la ricerca dei corsi a cui è iscritto uno studente: data la matricola (nessun corso selezionato), facendo click sul pulsante *Cerca*, controllare se è presente nel database ed in caso visualizzare tutti i corsi ai quali lo studente è iscritto. Se non è presente, visualizzare un messaggio di errore. (Esempio Fig. 3)
- Esercizio 2.5** Implementare la funzionalità di ricerca di uno studente iscritto ad un corso: selezionato un corso ed inserita la matricola di uno studente, facendo click su *Cerca*, è possibile sapere se lo studente specificato è iscritto a quel corso. (Esempio Fig. 4)
- Esercizio 2.6** Implementare la funzione di iscrizione di uno studente ad un corso: selezionato un corso ed inserita la matricola di uno studente, facendo click sul pulsante *Iscrivi*, è possibile iscrivere lo studente a quel corso. (Esempio Fig. 5)
- Esercizio 2.7** Controllare di aver gestito tutte le possibili condizioni di errore. (Es. studente non presente, studente già iscritto ad un corso, corso senza iscritti, studente senza corsi, caratteri non permessi nel campo matricola dello studente.)

Fig. 1

Lab3\_SegreteriaStudenti

Corso:

Studente:

Fig. 2

Lab3\_SegreteriaStudenti

Corso:

Studente:

146101	VIGGIANO	CLAUDIO	CIN1T3
148072	VANNINI	ALBERTO	LOP1T4
151528	LOPEZ	FABIO	GESZT4
154817	RUSSO	FRANCESCA	ORG1T4
155459	ROBERTO	MARCO	ORG1T4
155793	BARUCCO	ANDREA	ORG1T4

Fig. 3

Lab3\_SegreteriaStudenti

Corso:

Studente:   Nome

02CIXPG	8	Sistemi informativi aziendali	pd: 1
02PBVPG	8	Analisi dei sistemi economici	pd: 1
09AQGPG	8	Economia aziendale	pd: 1

Fig. 4

Lab3\_SegreteriaStudenti

Corso:

Studente:   Nome

James Gosling (001001) non è iscritto al corso "Diritto commerciale"

Fig. 5

Lab3\_SegreteriaStudenti

Corso:  ▼

Studente:

James Gosling (001001) è stato iscritto al corso di "Diritto commerciale"