

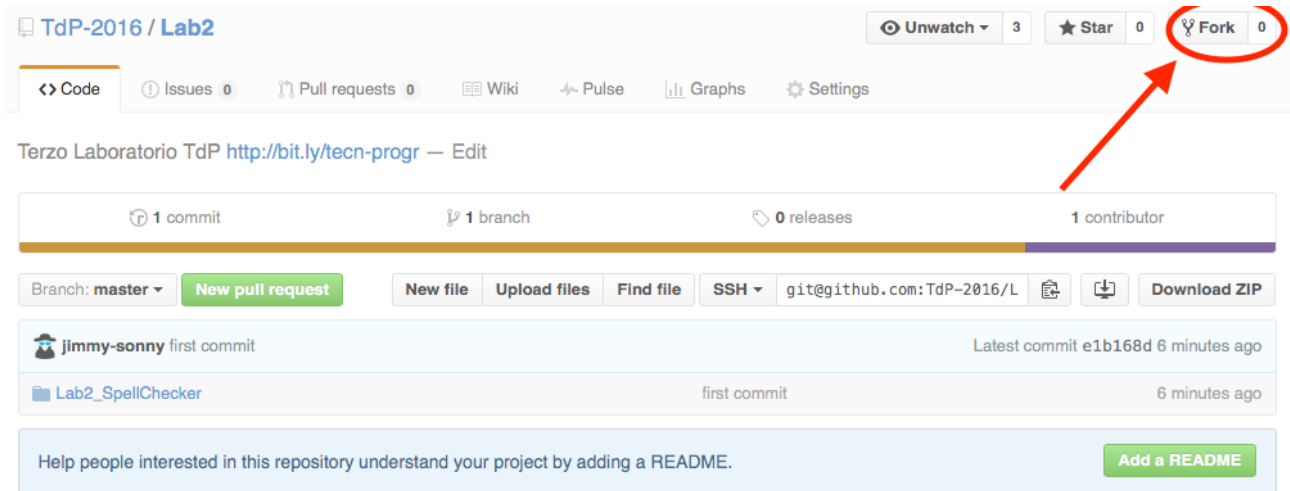
## 03FYZ TECNICHE DI PROGRAMMAZIONE

### 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 sesto laboratorio:

<https://github.com/TdP-2017/Lab06>

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-2017!), ad esempio:

<https://github.com/my-github-username/Lab06>

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 06 – 12 aprile 2017

Obiettivi dell'esercitazione:

- Apprendere il meccanismo della ricorsione
- Utilizzo del pattern MVC e DAO
- Utilizzo di JDBC

### Estratto dal tema d'esame 03/07/2014

Il noto sito di previsioni meteorologiche “ilMeteo” (<http://www.ilmeteo.it/>) mette a disposizione, gratuitamente, un archivio storico delle principali variabili climatiche nelle città italiane, con cadenza diurna.

Le informazioni relative alla situazione meteorologica sono rappresentate nella base dati avente la struttura schematizzata a fianco. Il database è denominato ‘meteo’ e contiene un’unica tabella, chiamata ‘situazione’.

Le informazioni presenti nella base dati fornita sono relative alle città di *Torino*, *Milano* e *Genova*, e coprono tutto e solo l’anno 2013.

Si intende costruire un’applicazione JavaFX che permetta di interrogare tale base dati, e calcolare informazioni a proposito dell’andamento climatico.

L’applicazione dovrà svolgere le seguenti funzioni:

1. Permettere all’utente di scegliere un mese dell’anno (valore intero tra 1 e 12) e visualizzare il valore dell’umidità media per quel mese in ciascuna delle città presenti nel database.
2. Risolvere il seguente problema di ottimizzazione **mediante un algoritmo ricorsivo**:

Sapendo che nel database sono presenti 3 città, supponiamo che un tecnico debba compiere delle analisi tecniche della durata di un giorno in ciascuna città. Le analisi hanno un costo per ogni giornata, determinato dalla somma di due contributi: un fattore costante (di valore 100) ogniqualvolta il tecnico si deve spostare da una città ad un’altra in due giorni successivi, ed un fattore variabile direttamente proporzionale all’umidità della città nel giorno considerato. Si trovi la sequenza delle città da visitare nei primi 15 giorni del mese selezionato, tale da minimizzare il costo complessivo rispettando i seguenti vincoli:

- Nei primi 15 giorni del mese, tutte le città devono essere visitate
- In nessuna città si possono trascorrere più di 6 giornate (anche non consecutive)
- Scelta una città, il tecnico non si può spostare prima di aver trascorso 3 giorni consecutivi.

situazione	
Localita	VARCHAR(50)
Data	DATE
Tmedia	INT(11)
Tmin	INT(11)
Tmax	INT(11)
Puntorugiada	INT(11)
Umidita	INT(11)
Visibilita	INT(11)
Ventomedia	INT(11)
Ventomax	INT(11)
Raffica	INT(11)
Pressioneslm	INT(11)
Pressionemedia	INT(11)
Pioggia	INT(11)
Fenomeni	VARCHAR(50)

**Continua nella pagina seguente**

## ESERCIZIO 1

**Analizzare su CARTA** l'algoritmo ricorsivo per trovare la migliore sequenza di città che il tecnico deve visitare. Utilizzare lo schema in *Figura 1* e le successive domande per impostare l'algoritmo ricorsivo.

```
// Struttura di un algoritmo ricorsivo generico
void recursive (... , level) {
    // E -- sequenza di istruzioni che vengono eseguite sempre
    // Da usare solo in casi rari (es. Ruzzle)
    doAlways();

    // A
    if (condizione di terminazione) {
        doSomething;
        return;
    }

    // Potrebbe essere anche un while ()
    for () {

        // B
        generaNuovaSoluzioneParziale;

        if (filtro) { // C
            recursive (... , level + 1);
        }

        // D
        backtracking;
    }
}
```

*Figura 1 Struttura base algoritmo ricorsivo*

Rispondere alle seguenti domande:

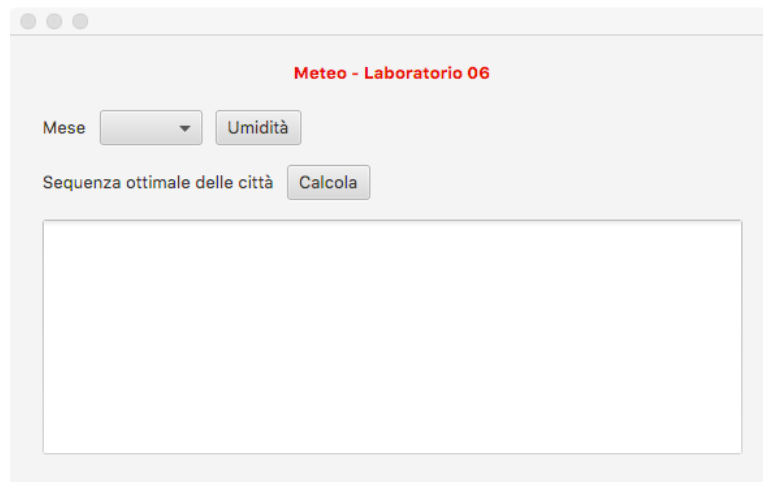
- Cosa rappresenta il "livello" nel mio algoritmo ricorsivo?
- Com'è fatta una soluzione parziale?
- Come faccio a riconoscere se una soluzione parziale è anche completa?
- Data una soluzione parziale, come faccio a sapere se è valida o se non è valida?  
(nb. magari non posso)
- Data una soluzione completa, come faccio a sapere se è valida o se non è valida?
- Qual è la regola per generare tutte le soluzioni del livello+1 a partire da una soluzione parziale del livello corrente?
- Qual è la struttura dati per memorizzare una soluzione (parziale o completa)?
- Qual è la struttura dati per memorizzare lo stato della ricerca (della ricorsione)?
- Sulla base dello schema presentato in *Fig. 1*, completare i blocchi (alcuni potrebbero essere non necessari)
  - o **A** – Condizione di terminazione
  - o **B** – Generazione di una nuova soluzione
  - o **C** – Filtro sulla chiamata ricorsiva
  - o **D** – Backtracking
  - o **E** – Sequenza di istruzioni da eseguire sempre

## ESERCIZIO 2

Dopo aver fatto il *fork* del progetto relativo a questo laboratorio, realizzare in linguaggio Java un'applicazione dotata di interfaccia grafica, Figura 2, che implementi i punti 1 e 2 dell'esercizio proposto.

L'applicazione va sviluppata seguendo il pattern MVC e il pattern DAO per l'accesso al database.

*Al fine di semplificare lo svolgimento, alcune classi e l'interfaccia grafica sono già fornite.*



*Figura 2: Interfaccia grafica Laboratorio 06*