03FYZ TECNICHE DI PROGRAMMAZIONE

Esercitazione di Laboratorio 06 es. 01 – 20 Aprile 2016

Obiettivi dell'esercitazione:

- Apprendere il meccanismo della ricorsione

ESERCIZIO 1

Scopo dell'esercitazione: **Analizzare su CARTA**, utilizzando lo schema seguente, l'algoritmo ricorsivo per risolvere il gioco del Sudoku.

Descrizione del problema: Nel gioco del Sudoku viene proposta una *griglia* di 9×9 celle, ciascuna delle quali può contenere un numero da 1 a 9, oppure essere vuota. La griglia è suddivisa in 9 righe orizzontali, 9 colonne verticali e in 9 "sottogriglie" di 3×3 celle contigue. Lo scopo del gioco è quello di riempire le caselle bianche con numeri da 1 a 9 in modo tale che in ogni riga, in ogni colonna e in ogni regione siano presenti tutte le cifre da 1 a 9, quindi senza ripetizioni.

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

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:
  (nb. Alcuni potrebbero essere non necessari)

  - **A** – Condizione di terminazione
  - **B** – Generazione di una nuova soluzione
  - **C** – Filtro sulla chiamata ricorsiva
  - **D** – Backtracking
  - **E** – Sequenza di istruzioni da eseguire sempre

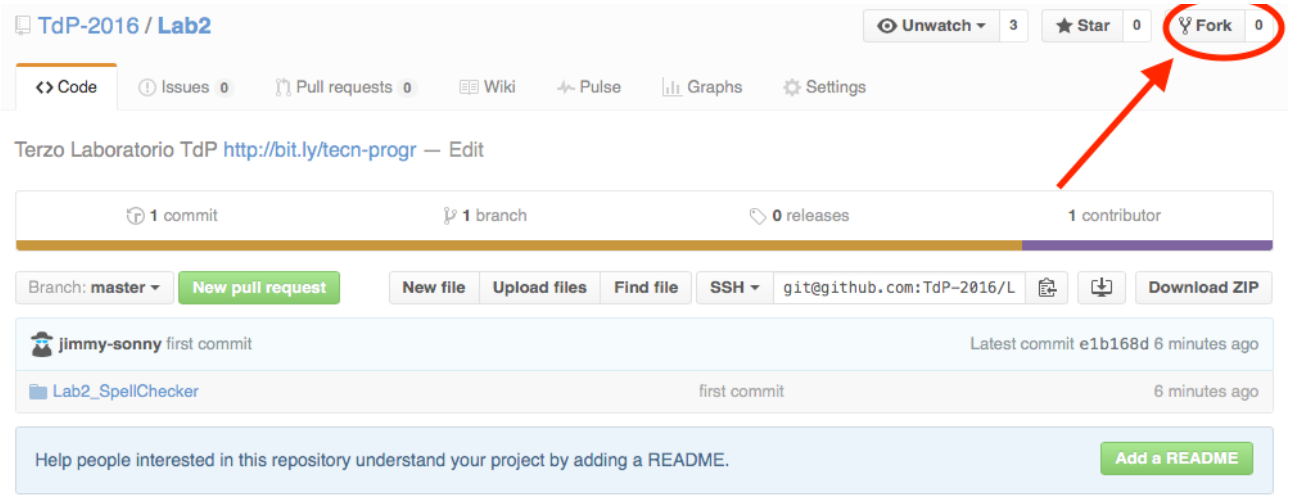
Fig. 1

```
// 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;
  }

}
```

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 settimo laboratorio:

  https://github.com/TdP-2016/Lab6
- 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/Lab6
- 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 es. 02 – 20 Aprile 2016

Obiettivi dell'esercitazione:

- Apprendere il meccanismo della ricorsione

ESERCIZIO 2

Scopo dell'esercitazione: Realizzare in linguaggio Java un'applicazione JavaFx (come quella mostrata in *Fig.2*) per risolvere il gioco del "Sudoku" utilizzando un algoritmo ricorsivo.

Funzionamento previsto dell'applicazione: Facendo click sul pulsante "Genera sudoku" l'applicazione crea una nuova griglia Sudoku secondo il livello di difficoltà selezionato nel menù a tendina. Cliccando su "Risolvi Sudoku" l'applicazione utilizza un algoritmo ricorsivo per risolvere la griglia proposta e stampa la soluzione nell'interfaccia.

Nota: Per agevolare lo sviluppo dell'applicazione, sia l'interfaccia grafica che il codice per la generazione di una nuova griglia Sudoku sono già presenti nel progetto iniziale.

Utilizzare la funzione `void printMatrixOnScreen(int[][] matrix)` per aggiornare le labels dell'interfaccia grafica.

Fig. 2