# Esercitazione 1 La Ricorsione Lineare

Corso di Fondamenti di Informatica II BIAR2 (Ing. Informatica e Automatica) e BSIR2 (Ing. dei Sistemi)  $A.A.\ 2010/2011$ 

29 ottobre 2010

#### Sommario

Scopo di questa esercitazione è risolvere problemi facendo uso di metodi ricorsivi.

### 1 Inverti Lista

Si vuole creare un metodo per la inversione degli elementi di una lista utilizzando esclusivamente il list iterator associato.

Esempio. Supponendo come input la lista contenente i valori (10,20,5,6,8) si dovrà modificare la lista in modo da ottenere i valori (8,6,5,20,10).

Programma Java. Scrivere una classe Java contenente il metodo

public static void inverti (ListIterator < Integer > iteratore)

Il metodo prende come parametro un iteratore associato ad una lista e inverte la posizione di tutti gli elementi della lista. Il metodo deve essere ricorsivo.

## 2 Problema delle K Regine

Si consideri il problema di disporre K regine in una scacchiera  $K \times K$  in modo che non ci siano 2 regine sotto minaccia. Due regine sono sottominaccia se si trovano sulla stessa riga, sulla stessa colonna o sulla stessa diagonale.

Esempio. Nella scacchiera di Fig. 1:

- R1 e R2 sono sotto minaccia perché fanno parte della stessa diagonale
- R2 e R4 sono sotto minaccia perché appartenenti alla stessa riga

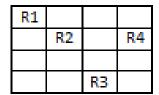


Figura 1: Una semplice scacchiera  $4 \times 4$ .

La Classe Scacchiera. Per poter rappresentare una scacchiera  $K \times K$  con K regine viene fornita la classe Scacchiera che permette di rappresentare una scacchiera con K regine e  $K \times K$  celle. Ogni regina è posizionata in una colonna e non possono esserci due regine sulla stessa colonna. E' invece possibile impostare il valore di riga associato ad una regina con il metodo impostaRegina o controllare se due regine si minacciano con il metodo sottoMinaccia. Segue descrizione dei principali metodi della classe Scacchiera.

public void impostaRegina(int numeroRegina, int numeroRiga)
 throws IndexOutOfBoundsException

Imposta la regina specificata sul numero di riga indicato

#### Parametri.

- numeroRegina: numero regina (tra 0 e k-1)
- numeroRiga: numero riga (tra 0 e k-1)

#### Throws.

• IndexOutOfBoundsException se il numero di regina o di riga non sono corretti

```
public boolean sottoMinaccia (int numRegina1, int numRegina2)
```

Ritorna true se la regina numRegina1 minaccia la regina numRegina2

#### Parametri.

- numeroRegina1: numero regina (tra 0 e k-1)
- numeroRegina2: numero regina (tra 0 e k-1)

```
public String toString()
```

Stampa la scacchiera su console

#### Overrides.

• metodo toString della classe Object

#### Programma Java. Scrivere il metodo

```
public static Scacchiera risolvi(int k)
```

Il metodo prende come parametro un intero K e restituisce una configurazione delle K regine tale che non ci siano regine che si minacciano oppure segnala il fatto che non sia possibile trovare nessuna configurazione che soddisfi il problema. Il metodo deve essere ricorsivo.

### 3 Permutazioni

Consideriamo un alfabeto di simboli di cardinalità N. Si vogliono generare tutte le possibili N! permutazioni degli N simboli dell'alfabeto.

Link Wikipedia alla voce Permutazione

**Esempio.** Consideriamo l'alfabeto  $\{a,b,c\}$  con N=3. In questo caso vogliamo generare tutti gli angrammi abc, acb, bac, bca, cab, cba.

Programma Java. Scrivere una classe Java contenente il metodo

```
public static void permutazioni (char[] alfabeto)
```

Il metodo prende come parametro un alfabeto di simboli e stampa in output tutte le permutazioni ottenute a partire dall'alfabeto dato.

# 4 Disposizioni con ripetizione (Per casa)

Consideriamo un alfabeto di simboli di cardinalità N e un intero K. Si vogliono generare tutte le possibili  $N^K$  disposizioni di lunghezza K con ripetizione degli N simboli dell'alfabeto.

Link Wikipedia alla voce Disposizione

**Esempio.** Consideriamo l'alfabeto  $\{a,b,c\}$  con N=3 e consideriamo K=2. In questo caso vogliamo generare tutte le sequenze aa, ab, ac, ba, bb, bc, ca, cb, cc.

Programma Java. Scrivere una classe Java contenente il metodo

```
public static void disposizioni (char[] alfabeto, int K)
```

Il metodo prende come parametro un alfabeto di simboli e stampa in output tutte le disposizioni di lunghezza K con ripetizioni ottenute a partire dall'alfabeto dato.

Suggerimento. Non utilizzare direttamente la funzione disposizioni per la ricorsione ma costruirne una privata con gli opportuni parametri.