

# Esercitazione\*

2014

## Esercizio 1: ricorsione

Scrivete dei metodi (uno per funzione) che calcolino le seguenti funzioni in modo ricorsivo:

1. *Somma di naturali.* Dato  $n \in \mathbb{N}$ , sia  $n'$  il suo successore. Ogni naturale maggiore di 0 è successore di qualche altro naturale. Date le seguenti regole:

- $n + m' = (n + m)'$
- $n + 0 = n$

implementare in modo ricorsivo la somma di interi.

2. *Quadrato di un naturale.* Dato  $n \in \mathbb{N}$  si calcoli  $n^2$  sfruttando le seguenti proprietà:

- $n^2 = ((n - 1) + 1)^2 = (n - 1)^2 + 2(n - 1) + 1$  se  $n > 0$
- $0^2 = 0$

3. *Massimo di un array di interi.* Trovare il massimo di un array di interi sfruttando il fatto che il massimo di un array con un elemento è l'elemento stesso mentre il massimo di un array di  $n$  elementi è il massimo tra il primo elemento ed il massimo del rimanente array di  $n - 1$  elementi.

4. *Somma di  $n$  interi.* Dato un array di  $n$  interi si implementi la somma di tutti gli elementi dell'array sapendo che la somma di un array di un elemento è l'elemento stesso e che la somma degli elementi di un array di  $n$  elementi è il primo elemento sommato alla somma dei restanti  $n - 1$ .

5. *Elevamento a potenza.* Dati due interi positivi  $a$  e  $b$  sfruttare le seguenti proprietà:

- $a^b = a \cdot a^{b-1}$  se  $b > 0$ .
- $a^0 = 1$  altrimenti.

per implementare in modo ricorsivo l'elevamento a potenza.

## Esercizio 2: il gioco dell'impiccato

Lo scopo di questo esercizio è l'implementazione del gioco dell'impiccato. Per l'implementazione sono forniti due file `dizionario.txt` e `Dizionario.class`. Mantenete questi file nella directory contenente i vostri file class. Avrete quindi a disposizione un metodo `Dizionario.getParola()` che ritorna una stringa di testo (tipo `String`) scelta a caso in un dizionario di oltre  $10^5$  parole.

All'avvio il programma deve indicare al giocatore il numero di errori che ha ancora a disposizione (in partenza il giocatore ha a disposizione 10 errori) e visualizzare una schermata come la seguente

---

\*Materiale originale a cura di Luca Manzoni e Stefano Beretta.

- - - - -

Inserisci un carattere:

Dopo l'inserimento di un carattere<sup>1</sup> il programma deve ristampare la stringa con scoperti i caratteri fino a quel momento indovinati. In caso il carattere scelto dall'utente sia già stato inserito in precedenza il programma deve chiederne un altro in input senza contare eventuali errori.

In caso di vittoria il programma deve stampare una scritta di congratulazioni al giocatore mentre in caso di sconfitta deve scrivere quale era la parola da indovinare.

Nella realizzazione del programma dovreste creare i seguenti metodi, i cui argomenti e tipi di ritorno non sono interamente specificati ma devono essere scelti a seconda dell'implementazione:

- un metodo chiamato `inserita` che ritorna `true` se il carattere fornito come argomento è già stato inserito in precedenza e `false` altrimenti.
- Un metodo chiamato `printWord` che data la stringa da cercare ed eventuali altri argomenti stampa la stringa con il carattere underscore (`_`) al posto dei caratteri non ancora scoperti.

L'implementazione può definire anche altri metodi a seconda delle scelte implementative che vengono compiute.

**Facoltativo.** Al termine della partita invece di chiudere il programma si chieda all'utente se vuole continuare o no e, nel caso scelga di continuare, avviare un'altra partita.

**Facoltativo.** Far scegliere all'avvio della partita il livello di difficoltà: facilissimo (12 errori), facile (10 errori), medio (7 errori), difficile (5 errori) e difficilissimo (3 errori).

**Suggerimenti.** Oltre alla stringa da cercare bisogna sempre tenere traccia di altre due informazioni:

- I caratteri inseriti fino a quel momento (ad esempio utilizzando una stringa od un array di caratteri).
- I caratteri della stringa che sono stati trovati (ad esempio utilizzando una seconda stringa od un array di booleani che dica quali caratteri considerare ma queste sono solo due scelte tra le molte possibili).

---

<sup>1</sup>Per l'inserimento dei caratteri utilizzare `Savitchin.readNonwhiteChar()`