

Laboratorio 4

Ripasso della classe Scanner

All'inizio del codice java, prima della dichiarazione `public class Nome`, inseriamo:

```
import java.util.Scanner;
```

Nel metodo che usa l'input, dichiariamo:

```
Scanner input = new Scanner(System.in);
```

La classe Scanner suddivide le linee di testo inserite in input in linee o token (parole delimitate da spazi), e ha i seguenti metodi:

```
int value = input.nextInt(); // legge il token successivo come numero
```

```
double value = input.nextDouble(); // successivo token come numero reale
```

```
String line = input.nextLine(); // ritorna una riga di testo
```

```
String word = input.next(); // ritorna il token successivo
```

Esercizio 1

Creare una classe **EserciziWhile.java** che contenga un main in cui:

1. Dato un intero n letto da input, si stampino a video tutti gli interi da 0 a n e poi da n a 0. Si usino sia il while che il for
2. Successivamente si stampino a video tutti gli interi dispari da 0 a n
3. Infine si stampi a video il fattoriale di n. Per trovare il valore si usi prima il while e poi il for. Si ricorda che il fattoriale di n è pari a: $1 * 2 * \dots * (n-1) * n$.

Esercizio 2

Scrivere un programma **MediaInput.java** che:

1. Legge da stdin una sequenza di interi positivi terminata da 0
2. Quando l'utente inserisce il numero 0, il programma interrompe la lettura degli input, stampa la media (come int) su stdout e termina
3. Lo 0 finale non fa parte della sequenza di interi su cui si calcola la media.

Verificare il programma con i seguenti input:

- 4 8 10 2 0
- 3 5 0
- 0

Esercizio 3

Scrivere un programma **DivisoriInput.java** che legge un intero da stdin e scrive su stdout tutti i suoi divisori.

Esercizio 4

Scrivere un programma **ParitaInput.java** che legge una sequenza di interi letti da stdin (terminati da uno 0 che viene escluso dalla sequenza) e alla fine della sequenza stampa:

- "Tutti i numeri inseriti sono dispari" se tutti i numeri inseriti sono dispari.
- "Tutti i numeri inseriti sono pari" se tutti i numeri inseriti sono pari.
- "Ci sono sia numeri pari che dispari" se compaiono sia numeri pari e dispari nella sequenza
- "Non hai inserito numeri" se la sequenza di numeri è vuota.

Verificare il programma con i seguenti input (per testare tutta la casistica):

- 2 4 6 0
- 9 3 7 5 0
- 1 2 3 4 0

- 0

Esercizio 5

Scrivere un programma **Floyd.java** che legge in input un numero N e stampa le prime N righe del triangolo di Floyd. Il triangolo di Floyd è costituito dai numeri naturali scritti in modo consecutivo, per riempire le righe con 1,2,3,... valori. Ad esempio per N=5 l'output deve essere:

```
1
2 3
4 5 6
7 8 9 10
11 12 13 14 15
```

Esercizio 6

Scrivere un programma **Quadrato.java** che legge in input un numero N e stampa a video un quadrato di N*N caratteri tra '*', '\' e ':' seguendo questo pattern (esempio per N=6):

```
\:~::~
*\:~::~
**\:~::~
***\:~::~
****\:~::~
*****\
```

Suggerimento: si ricorda che `System.out.print()` non va a capo, a differenza di `System.out.println()`.
Suggerimento: fare due cicli annidati, uno esterno per le righe, ed uno o più cicli interni per i caratteri sulla stessa una riga.

Esercizio 7

Scrivere un programma **Fibonacci.java** che legge in input un numero N e stampa a video i primi N numeri della successione di Fibonacci. Ad, esempio, per N=10, l'output atteso è:

```
0 1 1 2 3 5 8 13 21 34
```

Per semplicità assumere che $N \geq 2$.

Si ricorda che la successione di Fibonacci parte da due numeri 0 e 1, ed ogni elemento successivo della successione è ottenuto come somma dei due elementi precedenti.

Suggerimento: partite da due numeri $n=0$ ed $m=1$ e stampateli. Poi ad ogni iterazione aggiornali in modo che:

$$n' = m, \quad m' = n + m$$

e stampare m' .

Domanda: Per fare questo aggiornamento serve una variabile temporanea?

Esercizio 8

Scrivere un programma **MinMax10.java** che chiede all'utente 10 interi e successivamente stampa a video la differenza tra il minimo e il massimo.

- Si usi un ciclo for chiedendo a ogni iterazione di inserire un intero.
- Si usino variabili opportune per il minimo e per il massimo, inizializzate usando `Integer.MAX_VALUE` e `Integer.MIN_VALUE`.

Esercizio 9

Aprire la classe **AritNatIter.java**, che di partenza contiene i seguenti metodi:

1. Il metodo *somma* di due interi ottenuti usando solo il successore: Ad esempio, $s = s + 1$ è ammesso, ma non lo è $s = s + m$, se m diverso da 1.
2. Il metodo *moltiplicazione* di due interi m e n . Non si usa $m * n$ ma si richiama opportunamente un numero adeguato di volte *somma*. Si usi un `while` prima con indice crescente e poi decrescente

Si provi il corretto funzionamento dei metodi richiamandoli opportunamente nel main dove si possono testare proprietà numeriche come ad esempio l'associatività dell'addizione.

Completare la classe **AritNatIter.java** con i seguenti metodi.

1. un metodo *max* che dati due interi restituisca il massimo
2. un metodo *sommatoria* che, dato un intero n , restituisca la sommatoria $1+2+\dots+n$
3. un metodo *fattoriale* che, dato un intero n , restituisca il fattoriale di n
4. un metodo *potenza* che, dati due interi x e y , calcoli x elevato ad y usando la moltiplicazione

IMPORTANTE: Per ciascuno dei metodi aggiunti, **scrivere un codice di test nel main che lo richiama e ne verifica il funzionamento su uno o più casi.**

Esercizio 10 (opzionale)

Scrivere un programma **Freccia.java** che legge in input un numero N e stampa a video $(2*N-1)$ righe secondo il seguente pattern (esempio per $N=5$):

```
**
. **
.. **
... **
.... **
... **
.. **
. **
**
```

Esercizio 11 (opzionale)

Scrivere un programma **SommeCifre.java** che legge in input un numero N compreso tra 1 e 9, e calcola la somma di N numeri:

$1 + 22 + 333 + 4444 + 55555 + \dots$

Stampare un messaggio di errore se N non è compreso tra 1 e 9.

Suggerimento: scrivere un metodo *numeroNcifre* che prende in input un argomento n e ritorna il numero intero ottenuto da n volte la cifra n . Per ottenere tale numero, usare un ciclo che moltiplica per 10 e somma n .