

# Esercizi

- Operatori
- Decisioni
- Loop
- Stringhe e array
- ...

# Operatori

- `speed()`
  - Distanza e tempo → velocità media
- `distance()`
  - Due punti  $(x_0, y_0)$  e  $(x_1, y_1)$  in un piano → la loro distanza
- `engineCapacity()`
  - Alesaggio e corsa in mm, numero cilindri → cilindrata in cm cubi
- `digitSum()`
  - Intero → Somma delle sue cifre
- `score()`
  - Punto  $(x, y)$  → In base alla distanza dal centro, punteggio freccetta  $[1, 5, 10] \rightarrow [10, 5, 1, 0]$

# Decisioni

- `checkSign()` // un intero → “positive”, “negative”, o “zero”
- `isOdd()` // un intero → true se dispari
- `asWord()`
  - Un intero, se  $[0..9]$  → “zero”, “one” ... “nine”, altrimenti “other”
- `vote()`
  - Un voto in  $[0..100]$  → F  $\leq 50$ , E in  $(50, 60]$ , D in  $(60, 70]$ , C in  $(70, 80]$ , B in  $(80, 90]$ , A  $> 90$
- `isLeapYear()` // Intero → true se anno bisestile
- `sort()` // tre numeri → un array con i tre numeri ordinati

# Loop

- `sum()`
  - somma tutti i valori in `[first, last]` (o zero), p.es:  $(1, 3) \rightarrow 6$  e  $(3, 1) \rightarrow 0$
- `sumEven()`
  - somma tutti i numeri pari nell'intervallo
- `sqrt()`
  - Calcolo della radice quadrata usando il metodo di Newton  $x = 1; x = x/2 + z/(2x)$
  - Due versioni: con epsilon predefinito (0.001) e specificato dal chiamante
- Per un (piccolo) intero, scrivere metodi che calcolino:
  - il fattoriale
  - il numero di Fibonacci (0, 1, 1, 2, 3, 5, 8, ...)
  - la tavola pitagorica (ritornata come array bidimensionale)

# String e array

- reverse() // per una stringa
  - Copia ribaltata
- isPalindrome()
  - È un palindromo?
- removeVowels()
  - Stringa → copia senza vocali
- bin2dec()
  - Dalla stringa rappresentazione binaria di un intero al suo valore. “11010” → 26
- reverse() // per un array
  - Copia ribaltata
  - Alternativa: in place
- average()
  - Calcolo della media
- max()
  - Trova il massimo

# Altri

- isPrime()
- isArmstrong()
  - [https://it.wikipedia.org/wiki/Numero\\_di\\_Armstrong](https://it.wikipedia.org/wiki/Numero_di_Armstrong)
- isPangram()
  - <https://it.wikipedia.org/wiki/Pangramma>
- hammingDistance()
  - [https://it.wikipedia.org/wiki/Distanza\\_di\\_Hamming](https://it.wikipedia.org/wiki/Distanza_di_Hamming)
- acronym() // Model View Controller → MVC
- yahtzee(int[] dice, Category cat)
  - <https://it.wikipedia.org/wiki/Yahtzee>

# Numeri perfetti

[https://it.wikipedia.org/wiki/Numero\\_perfetto](https://it.wikipedia.org/wiki/Numero_perfetto)

- `isPerfect()` // perfetto
  - La somma dei divisori propri è uguale al numero:  $1 + 2 + 3 == 6$
- `isAbudant()` // abbondante
  - La somma è maggiore:  $1 + 2 + 3 + 4 + 6 > 12$
- `isDeficient()` // difettivo
  - La somma è minore:  $1 + 2 + 4 < 8$

# Extra

- `binarySum()` // “10” + “11” = “101” (len left == len right)
- `mergeSorted()` // [1,2,3,4], [3,4,5,8] → [1,2,3,3,4,4,5,8] –  $O(n)$
- `getSingle()` // [1,4,2,3,3,2,1] → 4 –  $O(n \lg n)$ ,  $O(n)$
- `hasOnlyUnique()` // “hello” → false, “helo” → true –  $O(n)$
- `isAnagram()` // “baba”, “abba” → true –  $O(n)$
- `duplicates(array)` → solo gli elementi duplicati –  $O(n^2)$
- `singles(array)` → solo gli elementi unici
- `median(array)` → mediano di un array –  $O(n \lg n)$ ,  $O(n)$