VETTORI E LISTE

Angelo Di Iorio

Università di Bologna

Vettori in Java

- Un array (vettore) è un contenitore:
  - di dimensione fissa
  - di elementi dello stesso tipo

- Per inizializzare un vettore in Java:

```
Integer[] integers = new Integer[10];
Integer[] integers = {10, 2, 4, 5, 6};
String[] words = {“cane”, “gatto”, “tacchino”};
```

- La classe *java.util.Arrays* (statica) espone metodi per accedere e manipolare vettori

Esercizio 0

- Scrivere una classe Java per manipolare vettori (di interi), che espone i seguenti metodi:
  - `min(Integer[] integers)`: calcola il minimo valore nel vettore in input
  - `isOrdered(Integer[] integers)`: verifica se il vettore in input è ordinato

Esercizio 1

- Scrivere una classe Java per modellare e gestire una mensola di libri
- Per questo esercizio:
  - tutti i libri hanno la stessa dimensione
  - la capacità della mensola è fissata
  - è possibile mettere/prendere un libro in/da una data posizione

Esercizio 1

- La classe `Mensola` espone i metodi per eseguire le seguenti operazioni:
  - inizializzare una mensola di capacità `c`
  - contare quanti libri ci sono sulla mensola
  - verificare se la mensola è vuota (piena)
  - verificare se la posizione `p` è libera
  - prendere il libro nella posizione `p`
  - mettere un libro nella posizione `p`
  - stampare i titoli dei libri sulla mensola (nell'ordine in cui sono)

Esercizio 2

- Scrivere una classe Java per modellare e gestire una mensola di CD
- Per questo esercizio:
  - tutti i CD hanno la stessa dimensione
  - la capacità della mensola è fissata
  - è possibile mettere/prendere un CD in/da una data posizione

- E se volessimo usare la mensola per VHS, DVD o altri oggetti?

Esercizio 3 (sul libro Es. 2.10)

- Sia dato un vettore di *n* elementi che possono assumere solo i tre valori VERDE, BIANCO e ROSSO.
- Si vuole ordinare il vettore in tempo lineare in modo che tutti i “verdi” precedano tutti i “bianchi” e tutti i “bianchi” precedano tutti i “rossi”.
- Le uniche operazioni ammesse sono l’esame di un elemento e lo scambio di due elementi, dati i loro indici (non si possono usare vettori di appoggio)

Esercizio 3 (sul libro Es. 2.10)

```
ordinaBandiera(integer[] B, integer n)
    integer k ← 1
    integer j ← n
    while k ≤ n and B[k] = VERDE do k ← k + 1
    while j ≥ 1 and B[j] = ROSSO do j ← j − 1
    integer i ← k
    while i ≤ j do
        if B[i] = ROSSO then
            B[i] ↔ B[j]
            j ← j − 1
        else if B[i] = VERDE then
            B[i] ↔ B[k]
            k ← k + 1
            i ← i + 1
        if B[i] = BIANCO then
            i ← i + 1
```

LISTE

Esercizio 4

- Implementare una classe Java per gestire una sequenza di interi
- Realizzare la sequenza tramite una *lista bidirezionale senza sentinella*

Esercizio 4

- La classe espone i metodi per eseguire le seguenti operazioni:
  - costruttore `List()` // inizializza lista vuota
  - `Position insertAtPosition(Position p, Integer i)`
    - aggiunge l'intero `i` alla lista come predecessore di `p`;
    - se `p==null` aggiunge il valore in fondo alla lista
    - ritorna il puntatore all'elemento appena aggiunto
  - `remove(Position p)` // rimuove l'elemento in posizione `p`
  - `printAll()` // scandisce la lista e stampa tutti i valori in ordine