Tutorato 2 - Programmazione 2

Esercizio 1

Siano date le seguenti dichiarazioni.

```
interface Frutto {}
class Mela implements Frutto {}
class Pera implements Frutto {}
class Arancia implements Frutto {}
```

Si crei una classe `Cesto` di frutta:

- Deve avere capacità prefissata (costruttore).
- Deve essere possibile "specializzarlo" solo per frutti specifici (utilizzando i type parameters).
- Deve supportare l'operazione `aggiungi` di un frutto (se c'è spazio).
- Deve memorizzare tutti i frutti all'interno della stessa collezione.
- Deve essere possibile iterare gli elementi all'interno del cesto.

Si crei un metodo statico "esterno" `conta` che stampi il numero di `Mela`, `Pera`, `Arancia` all'interno di un `Cesto`.

- Il numero di elementi deve essere calcolato AL VOLO.

Nel `main`:

- Si vada a creare un `Cesto` di `Mela` e lo si riempi con un numero arbitrario di mele.
- Si vada a chiamare `conta` con il `Cesto` appena creato.

Esercizio 2

Siano date le seguenti dichiarazioni:

```
public class Scarpa {
    // Colore RGB
    public Carta(String modello, int taglia) {
        /* impl */
    }
}
```

> ps: è possibile estendere le dichiarazioni

- Si crei una classe `Esposizione` contenente una `List` di `Scarpa` di nome `scarpe`:

    - Deve supportare l'operazione `ordina` (ordine lessicografico per modello, ordine per taglia all'interno di uno stesso modello).
    - Deve supportare l'operazione `compra` che estrae la scatola di `Scarpa` in cima ad un ipotetico stack.
    - Deve supportare l'operazione `esponi` per aggiungere una `Scarpa` all'`Esposizione`.
    - Deve contenere un `isEmpty`.
- Si crei una classe `Magazzino` contenente una mappa `Scarpa` - *n° di scarpe di quel tipo* di nome `scarpe`:
    - Deve supportare l'operazione `aggiungi` che aggiunge una `Scarpa` al magazzino (incrementando il numero di scarpe).
    - Per questioni didattiche DEVE utilizzare un `HashMap`.
    - Deve supportare l'operazione `estrai(Scarpa scarpa)` che "rimuove" (decrementa) la scarpa passata come parametro, se presente.
- Si ridefinisca `Scarpa.toString()` per stampare `<modello> <taglia>`.

Si usi questo main:

```
public class Es2 {

    public static void main(String[] args) {
        final Magazzino magazzino = new Magazzino();
        final Esposizione esposizione = new Esposizione();

        final Random rand = new Random();

        for (char modello = 'A'; modello <= 'C'; ++modello) {
            for (int taglia = 35; taglia <= 47; ++taglia) {
                final int count = rand.nextInt(15);

                for (int i = 0; i < count; ++i) {
                    final Scarpa scarpa = new
Scarpa(String.valueOf(modello), taglia);
                    magazzino.aggiungi(scarpa);
                }

                final Scarpa scarpa = new Scarpa(String.valueOf(modello),
taglia);

                int magazzinoCount = magazzino.scarpe.getOrDefault(scarpa,
0);

                if (magazzinoCount != count) {
```

```
                    System.err.println("Errore Magazzino.aggiungi().
(hash?)");
                }

                for (int i = 0; i < magazzinoCount; ++i) {
                    if (!magazzino.estrai(scarpa)) {
                        System.err.println("Errore Magazzino.estrai()");
                    }

                    esposizione.esponi(scarpa);
                }

                magazzinoCount = magazzino.scarpe.getOrDefault(scarpa, 0);

                if (magazzinoCount != 0) {
                    System.err.println("Errore Magazzino.aggiungi().");
                }
            }
        }

        Collections.shuffle(esposizione.scarpe);

        esposizione.ordina();

        System.out.println(esposizione.scarpe);

    }
}
```