Esercitazione per i capitoli 14-16

Creare un workspace "CognomeMatricola", ad esempio "Rossi1234", con un progetto omonimo. Implementare la classe "Magazzino" nel progetto, fungendo da contenitore per prodotti alimentari ed elettronici.

Classe Magazzino. Contiene istanze di differenti tipi di prodotti ed è così definita:

- Attributi:
  - prodotti: contenitore per memorizzare vari tipi di prodotti. **Non sono ammessi prodotti che abbiano lo stesso *codice Prodotto***.
- Operazioni:
  - boolean **aggiungiProdotto**(Prodotto prodotto): *true* se aggiunge un prodotto al contenitore, *false* altrimenti. Non c'è un limite al numero di prodotti.
  - boolean **cancellaProdottiByCosto**(double costo): cancella tutti i prodotti il costo è inferiore o uguale a quello specificato in input; *true* se almeno un prodotto è stato cancellato, *false* altrimenti.
  - ElencoProdotti **getProdottiCostoSuperiore**(double costo): restituisce l'elenco dei prodotti il cui costo è uguale o superiore al valore di *costo*. L'elenco deve essere immutabile.
  - Prodotto **searchByCodice**(int codiceProdotto): esegue la ricerca di un prodotto mediante il codiceProdotto; se il prodotto non è presente restituisce un'eccezione personalizzata denominata *ProdottiException*.
  - ElencoProdotti **getProdottiByNome**(String nomeProdotto): restituisce l'elenco dei prodotti ordinato per nome del prodotto. L'elenco deve essere immutabile.
  - *ElencoProdotti* **getProdottiByCostoDecr**(String nomeProdotto): restituisce l'elenco dei prodotti ordinato per costo decrescente del prodotto. L'elenco deve essere immutabile.

Qualunque tipo di prodotto sarà caratterizzato da:

- Attributi: nome (String), costo (double), codiceProdotto (numero intero identificativo di un prodotto).
- Operazioni:
  - **Prodotto**(String nome, double costo, int codiceProdotto): costruttore per inizializzare gli attributi.
  - double **calcolaCosto**(): calcola il costo del prodotto.
  - String **toString**(): rappresentazione testuale di un prodotto. In questo metodo si usi StringBuilder per ottimizzare la concatenazione di stringhe.

I **prodotti alimentari** si caratterizzano per avere:

- Attributi: energia (int), tempoConservazione (int).
- Metodi:

- **ProdottoAlimentare**(String nome, double costo, int energia, int tempoConservazione, int codiceProdotto): costruttore per inizializzare gli attributi.
    - double **calcolaCosto**(): il costo del prodotto è dimezzato se tempoConservazione è uguale o inferiore a 2.

I **prodotti elettronici** si caratterizzano per avere:
- Attributi: garanzia (int).
- Metodi:
    - **ProdottoElettronico**(String nome, double costo, int garanzia, int codiceProdotto): costruttore per inizializzare gli attributi.
    - double **calcolaCosto**(): se la garanzia è superiore a 12 il costo del prodotto aumenta del 10%.

Eccezione personalizzata denominata ProdottiException

Classe per gestire errori legati ai requisiti informativi dei prodotti. Tale classe ha i seguenti metodi:

- public **ProdottiException**(String message): costruttore che accetta un messaggio di errore; utile quando si voglia personalizzare un messaggio di errore per tale tipologia di eccezioni.
- *enne* metodi statici, uno per ogni caso specifico di eccezione rilevata per i prodotti come, ad esempio:
    - public static void **costoNegativo**() throws ProdottiException: metodo statico per creare e sollevare un'eccezione quando il costo è negativo. Solleva eccezione ProdottiException con il messaggio appropriato per il costo negativo (esempio: “Il costo non può essere negativo”).

Requisiti informativi dei prodotti

- Il *costo* non può mai essere negativo. Se negativo viene sollevata un’eccezione ProdottiException.
- *Energia* non può mai essere negativo. Se negativo viene sollevata un’eccezione ProdottiException.
- La *garanzia* è un valore compreso tra 2 e 24, se non appartiene a tale intervallo gli viene assegnato 2 (valore di default).
- Il *tempo di conservazione* è un valore almeno pari ad 1, se inferiore gli viene assegnato 1 (valore di default).
- L’attributo *nome,* obbligatorio per ogni prodotto. Se il nome non è opportunamente avvalorato, ad esempio se è null oppure ha lunghezza inferiore a 3 allora viene sollevata un’eccezione *ProdottiException*.

NOTA BENE

Per ogni classe si aggiungano tutte i metodi get() funzionali alla realizzazione dei metodi sopra definiti.

Esecuzione del TEST

In una classe di Test, denominata *TestMagazzino*, si eseguano i seguenti scenari di esecuzione.

Scenario 1: Aggiunta di Prodotti

- Si crei un magazzino vuoto.
- Si aggiungano i seguenti prodotti:
    1. alimentare ("Pasta") con costo 2.5, energia 150, tempoConservazione 5 e codiceProdotto 1.
    2. elettronico ("Laptop") con costo 1200, garanzia 24 e codiceProdotto 2.
    3. alimentare ("Frutta") con costo 3.0, energia 120, tempoConservazione 3 e codiceProdotto 3.
    4. elettronico ("Smartphone") con costo 800, garanzia 12 e codiceProdotto 2 (stesso codice del Laptop).
    5. elettronico “ab” con costo 800, garanzia 12 e codiceProdotto 4.
    6. alimentare ("Pasta") con costo 7.8, energia 220, tempoConservazione 6 e codiceProdotto 8.
- Si stampi il risultato dell'aggiunta di ciascun prodotto (*true* indica che il prodotto è stato aggiunto con successo).

Scenario 2: Ricerca di un Prodotto per Codice

- Si cerchi un prodotto nel magazzino utilizzando il codice identificativo (ad esempio, codice 1, codice 7).
- Se il prodotto viene trovato, si stampi il dettaglio del prodotto. Se non viene trovato, si gestisca un'eccezione e si stampi il motivo.

Scenario 3: Cancellazione di Prodotti per Costo

- Si tenti di cancellare i prodotti con costo inferiore o uguale a 2.0 (nessun prodotto viene cancellato). Si stampi a video l’esito.
- Si tenti di cancellare i prodotti con costo inferiore o uguale a 1.5 (almeno un prodotto viene cancellato). Si stampi a video l’esito.

Scenario 4: Elenco di Prodotti per Costo Superiore

- Si ottenga un elenco di prodotti con costo superiore a 1.0 e si stampi l'elenco riportando il dettaglio per ogni prodotto.

Scenario 5: Elenco di Prodotti per Nome

- Si ottenga un elenco di prodotti con il nome "Pasta" ordinato per nome e si stampi l'elenco riportando nome e costo di ogni prodotto.

Scenario 6: Elenco di Prodotti per Costo Decrescente

- Si ottenga un elenco di prodotti ordinato per costo decrescente e si stampi l'elenco riportando codice e costo di ogni prodotto.