Esercizio: Classe Ingrediente e classe Ricetta

In questa esercitazione creerete due semplici classi Java, Ingrediente e Ricetta, per modellare ricette culinarie. Alla fine dell'esercitazione dovrete fornire i file sorgente (Ingrediente.java, Ricetta.java, Main.java) e una breve dimostrazione in main che mostri il funzionamento dei metodi richiesti

Classe Ingrediente

Attributi privati

- nome — nome dell'ingrediente
- quantita — quantità (double) (es. 200) — quantità nelle unità indicate
- unita — unità di misura (String), es. "g", "ml", "pz"
- caloriePerUnita — calorie per unità (double)
- costoPerUnita — costo per unità (double)
- vegetariano — boolean che indica se l'ingrediente è vegetariano

Costruttori

- **Costruttore principale**: inizializza tutti gli attributi tramite parametri.
- **Costruttore di copia**: crea un nuovo Ingrediente copiando tutti i campi da un altro Ingrediente.

Metodi

- **Getter e setter** per tutti gli attributi.
    - I setter devono validare i valori sensati: quantita >= 0, caloriePerUnita >= 0, costoPerUnita >= 0, nome e unita non vuoti.
- **toString()**: restituisce una stringa leggibile dell'ingrediente (es. "200 g Farina — 364 kcal/100g — €0.50/unit — Vegetariano: true").
- **scalaQuantita(double fattore)**: moltiplica quantita per fattore (utile per scalare le dosi).
- **calorieTotali()**: restituisce quantita * caloriePerUnita.
- **costoTotale()**: restituisce quantita * costoPerUnita.

Classe Ricetta

Attributi privati

- nome — nome della ricetta
- tempoPreparazione — tempo in minuti (int)
- difficolta — "Facile", "Media" o "Difficile" (String)
- porzioni — numero intero di porzioni (int)
- ingrediente1, ingrediente2, ingrediente3 — tre oggetti Ingrediente

Costruttori

- **Costruttore principale**: riceve tutti i parametri necessari (compresi tre oggetti Ingrediente) e inizializza gli attributi.
- **Costruttore di copia**: riceve un'altra Ricetta e crea una copia completa (deep copy): copia i campi semplici e crea nuove istanze Ingrediente per ingrediente1..3.

Metodi richiesti

1. **Getter e setter** per tutti gli attributi.
    - I setter devono validare: porzioni > 0, tempoPreparazione > 0, difficolta tra i valori ammessi.
2. **toString()**: restituisce una descrizione leggibile della ricetta, includendo nome, porzioni, tempo, difficoltà e la descrizione dei tre ingredienti.
3. **scalaDosi(int nuovePorzioni)**: scala **in place** (modifica la ricetta) tutte le quantità degli ingredienti in modo proporzionale rispetto alle porzioni attuali e aggiorna porzioni.

    - Calcolo del fattore: fattore = nuovePorzioni / (double) porzioniAttuali.
4. **calcolaCalorieTotali()**: calcola e restituisce la somma delle calorieTotali() dei tre ingredienti (valore **per l'intera ricetta**).
5. **stimaCostoTotale()**: calcola e restituisce la somma dei costoTotale() dei tre ingredienti (valore **per l'intera ricetta**).
6. **isVegetariana()**: restituisce true se **tutti e tre** gli ingredienti hanno vegetariano == true.
7. **sostituisciIngrediente(int numeroIngrediente, Ingrediente nuovoIngrediente)**: sostituisce ingrediente1/2/3 con nuovoIngrediente. Restituisce true se numeroIngrediente è valido (1..3), false altrimenti.
8. **confrontaDifficolta(Ricetta altra)**: confronta la difficoltà secondo l'ordine Facile < Media < Difficile e restituisce: "Più difficile", "Meno difficile" o "Stessa difficoltà".
9. **adattaRicettaPerOspiti(int numOspiti)**: crea e restituisce **una nuova Ricetta** (non modifica l'originale) con dosi scalate per numOspiti porzioni. Implementazione: creare una copia tramite costruttore di copia, chiamare scalaDosi(numOspiti) sulla copia e restituirla.