

Programmazione a oggetti - Java

Esercitazione 4

Contatti:

Prof. Angelo Gargantini – angelo.gargantini@unibg.it

Dott.ssa Silvia Bonfanti – silvia.bonfanti@unibg.it

Classi e oggetti

Esercizio 24

- Scrivere una classe Dipendente. Ciascun dipendente ha un nome (di tipo stringa) e uno stipendio (di tipo double). Scrivere un costruttore con due parametri (nome e stipendio) e i metodi per conoscere nome e stipendio.
- Aggiungere un metodo incrementaStipendio(double percentuale), che incrementi lo stipendio del dipendente secondo una certa percentuale.
- Fare una classe di prova contenente il metodo main per testare i metodi della classe Dipendente

Esercizio 25

- Creare una classe `Studente` in cui si crea uno studente e si implementano le seguenti operazioni (metodi):
 - Aggiungi voto esame in posizione X
 - Calcola media voti esami sostenuti
 - Modifica/Inserisci paese di residenza

Ogni studente è costituito da:

- Nome e cognome
 - Un array di N voti (inizialmente tutti gli elementi dell'array sono 0)
 - Paese di residenza
- Fare una classe di prova contenente il metodo `main` per testare i metodi della classe `Studente`; alcuni studenti vengono creati con la residenza nota, altri la residenza non è nota al momento della creazione (usa due costruttori)

Esercizio 26

- Creare una classe Automobile che ha un ID e il numero di km percorsi. Implementare le seguenti operazioni (metodi):
 - isMaggioreKm: data l'auto corrente confrontarla con quella passata come parametro; ritorna true se i km dell'auto corrente sono maggiori di quella passata come parametro, false altrimenti.
 - isMinoreKm: data l'auto corrente confrontarla con quella passata come parametro; ritorna true se i km dell'auto corrente sono minori di quella passata come parametro, false altrimenti.
- Fare una classe di prova contenente il metodo main per testare i metodi della classe Automobile.
- Creare un array di automobili e ordinarle in ordine crescente di km utilizzando l'algoritmo BubbleSort. Stampare gli ID delle auto prima l'ordinamento e dopo l'ordinamento (verificare che siano ordinati correttamente - per verificare se scambiare o no due auto utilizzare il metodo isMaggioreKm definito nella classe Automobili).

Esercizio 27

- Scrivere un programma per la gestione di un acquario. Un acquario è costituito da vasche identificabili da un nome. In ogni **Vasca** sono contenuti un certo numero di Pesci (utilizza `arrayList/Sequenza/Vector`). Ogni **Pesce** è qualificato da un identificativo alfanumerico, dall'età e dalla profondità. Implementa i seguenti metodi nella classe **Vasca**:
 - `inserisciPesce`: inserire un nuovo pesce nella vasca
 - `stampaPesci`: stampare i pesci presenti nella vasca
 - `eliminaPesce`: elimina un certo pesce dato l'identificativo.
- Definisci anche una classe per l'oggetto **Pesce**
- Testa i metodi in una classe di prova

Esercizio 28

- Scrivi un programma in grado di gestire le ordinazioni ai tavoli di un ristorante. Nel ristorante sono disposti dei tavoli, ciascuno identificato da un id numerico e da un numero massimo di coperti. Per ciascun **Tavolo** devono essere memorizzati i piatti consumati (utilizza `ArrayList/Sequenza/Vector` inizialmente vuota). Ogni **Piatto** è identificato da un id, dalla quantità e dal suo prezzo. Inoltre ogni tavolo contiene una lista (inizialmente vuota) di persone sedute al tavolo. Ogni **Persona** è identificata da nome e cognome. Nella classe `Tavolo` implementa i metodi per aggiungere le persone, aggiungere i piatti ordinati, rimuovere i piatti non più desiderati (passa l'id come parametro del metodo), calcolare il totale del tavolo e calcolare il prezzo medio a persona.