Esercitazione di FdP-B

Lunedì, ore 15:30 - 17:30

Merenda Saverio Mattia, 12/05/2025

Esercizio 1

Simulatore di Feed di Social Network

Interfaccia Utente: definisce un utente del social network

- Un metodo **String getIdUnivoco()** che ritorna l'ID univoco dell'utente.
- Un metodo **String getNomeVisualizzato()** che ritorna il nome visualizzato dall'utente.

Simulatore di Feed di Social Network

Classe UtenteSemplice: Implementazione dell'interfaccia Utente

- Implementa l’interfaccia **Utente**.
- Attributi: **idUnivoco** (String), **nomeVisualizzato** (String).
- Un costruttore che accetta ID e nome.
- Implementazione dei metodi di **Utente**.
- Ridefinizione di **equals** e **hashCode** basati su **idUnivoco**.

Simulatore di Feed di Social Network

Classe astratta Post: rappresenta un post generico nel feed

- Attributi protetti: **idPost** (String, univoco), **autore** (Utente), **timestamp** (java.time.LocalDateTime).
- Un costruttore che accetta ID, autore e timestamp.
- Metodi **getIdPost()**, **getAutore()**, **getTimestamp()**.
- Un metodo astratto **String getContenutoTestuale()**.
- Implementazione dell'interfaccia **Comparable<Post>** e di **compareTo(Post other)**: ordina i post dal più recente al meno recente (basato su timestamp).
- Ridefinizione di **equals** e **hashCode** basati su **idPost**.

Simulatore di Feed di Social Network

Classe TextPost: estende Post per rappresentare un post testuale

- Estende la classe astratta **Post**.
- Attributo: **testo** (String).
- Un costruttore che accetta ID, autore, timestamp e testo.
- Implementazione di **getContenutoTestuale()** che ritorna il testo del post.

Simulatore di Feed di Social Network

Classe astratta MediaPost: estende Post per rappresentare post con contenuti multimediali

- Estende **Post**.
- Attributo: **urlMedia** (String).
- Un costruttore che accetta ID, autore, timestamp e URL del media.
- Metodo astratto **String getUrlMedia()**.

Simulatore di Feed di Social Network

Classe ImagePost: estende MediaPost

- Estende **MediaPost**.
- Un costruttore che accetta ID, autore, timestamp e URL immagine.
- Implementazione di **getUrlMedia()** che ritorna una stringa tipo "[Immagine: url]".

Simulatore di Feed di Social Network

Classe VideoPost: estende MediaPost

- Estende **MediaPost**.
- Attributo: **durataSecondi** (int).
- Un costruttore che accetta ID, autore, timestamp, URL video e durata.
- Implementazione di **getUrlMedia()** che ritorna una stringa tipo "[Video: url (durata)]".

Simulatore di Feed di Social Network

Classe Feed: gestisce una collezione ordinata di Post utilizzando java.util.List

- Attributo: **posts** (una **List<Post>**).
- Un costruttore che inizializza la lista (es. ArrayList).
- Un metodo **void aggiungiPost(Post p)** che aggiunge un post alla lista e la mantiene ordinata per timestamp decrescente.
- Un metodo **List<Post> getPostsRecenti(int N)** che ritorna gli **N** post più recenti.
- Un metodo **List<Post> getPostsByAutore(Utente u)** che ritorna tutti i post di un dato autore, ordinati per timestamp.

Esercizio 2

Piattaforma Corsi Online in Java

Classe astratta MaterialeDidattico: rappresenta un elemento di contenuto generico all'interno di un modulo

- Attributi: **titolo** (String), **idMateriale** (String, univoco).
- Un costruttore che accetta titolo e ID.
- Metodi getter per titolo e ID.
- Un metodo astratto **int getDurataStimataMinuti()**.
- Ridefinizione di equals e hashCode basati su idMateriale.

Piattaforma Corsi Online in Java

Classe Quiz: estende MaterialeDidattico.

- Estende **MaterialeDidattico**.
- Attributi: **numeroDomande** (int), **tempoLimiteMinuti** (int).
- Un costruttore che accetta titolo, ID, numero domande e tempo limite.
- Implementazione di **getDurataStimataMinuti()** che ritorna **tempoLimiteMinuti**.

Piattaforma Corsi Online in Java

Classe LezioneVideo: estende MaterialeDidattico.

- Estende **MaterialeDidattico**.
- Attributi: **urlVideo** (String), **durataEffettivaMinuti** (int).
- Un costruttore che accetta titolo, ID, URL e durata.
- Implementazione di **getDurataStimataMinuti()** che ritorna **durataEffettivaMinuti**.

Piattaforma Corsi Online in Java

Classe Modulo: rappresenta un modulo didattico contenente vari materiali.
Implementa Comparable<Modulo>

- Attributi: **nomeModulo** (String), **ordine** (int), **contenuti** (una List<MaterialeDidattico>).
- Un costruttore che accetta nome e ordine, inizializza la lista dei contenuti.
- Un metodo **void aggiungiContenuto(MaterialeDidattico md)** che aggiunge materiale alla lista.
- Un metodo **int getDurataTotaleModuloMinuti()** che somma le durate stimate di tutti i contenuti.
- Implementazione di **compareTo(Modulo other)** basata sull'ordine.

Piattaforma Corsi Online in Java

Classe Corso: rappresenta un corso online, composto da moduli.

- Attributi: **titoloCorso** (String), **idCorso** (String), **docente** (String), **moduli** (una List<Modulo>, mantenuta ordinata per ordine).
- Un costruttore che accetta titolo, ID e docente, inizializza la lista dei moduli.
- Un metodo **void aggiungiModulo(Modulo m)** che aggiunge un modulo mantenendo l'ordine.
- Un metodo **int getDurataTotaleCorsoMinuti()** che somma le durate totali dei moduli.
- Ridefinizione di **equals** e **hashCode** basati su **idCorso**.

Piattaforma Corsi Online in Java

Classe Piattaforma: gestisce l'insieme dei corsi e le iscrizioni degli studenti

- Attributi: **corsiDisponibili** (una **Map<String, Corso>**, dove la chiave è **idCorso**), iscrizioni (una **Map<String, Set<String>>**, dove la chiave è **idStudente** e il valore è un **Set** di **idCorso** a cui è iscritto).
- Un costruttore che inizializza le mappe.
- I metodi **void aggiungiCorso(Corso c)** e **Corso getCorso(String idCorso)**.
- Un metodo **void iscriviStudente(String idStudente, String idCorso)** che aggiunge l'iscrizione. Lancia **NoSuchElementException** se il corso non esiste.
- Un metodo **Set<Corso> getCorsiStudente(String idStudente)** che ritorna l'insieme dei corsi a cui uno studente è iscritto.
- Un metodo **Set<String> getStudentiIscritti(String idCorso)** che ritorna l'insieme degli ID degli studenti iscritti a un corso.

Esercizio 3

Sistema Prenotazione Voli in Java

Classe Aeroporto: rappresenta un aeroporto

- Attributi: **codiceIATA** (String, es. "MXP"), **nomeCompleto** (String, es. "Milano Malpensa"), **citta** (String).
- Un costruttore che accetta codice IATA, nome e città.
- Metodi getter per tutti gli attributi.
- Ridefinizione di **equals** e **hashCode** basati su **codiceIATA**.

Sistema Prenotazione Voli in Java

Classe Volo: rappresenta un volo di linea. Implementa Comparable<Volo>

- Attributi: **numeroVolo** (String, es. "AZ123"), **aeroportoPartenza** (Aeroporto), **aeroportoArrivo** (Aeroporto), **orarioPartenza** (java.time.LocalDateTime), **orarioArrivo** (java.time.LocalDateTime), **capacitaPosti** (int), **postiPrenotati** (una Set<String> contenente i numeri dei posti già assegnati, e.g., "12A", "3B").
- Un costruttore che accetta numero volo, aeroporti, orari e capacità. Inizializza **postiPrenotati** come un **HashSet** vuoto.
- Metodi getter per tutti gli attributi.
- Un metodo **int getPostiDisponibili()** che ritorna capacitaPosti.
- Un metodo **boolean prenotaPosto(String numeroPosto)** che aggiunge il **numeroPosto** al **Set postiPrenotati** se disponibile e valido (es. non già presente). Restituisce true se la prenotazione del posto ha successo, false altrimenti.
- Un metodo **void cancellaPrenotazionePosto(String numeroPosto)** che rimuove il posto dal **Set**.
- Implementazione di **compareTo(Volo other)**: ordina i voli per **orarioPartenza** crescente.
- Ridefinizione di **equals** e **hashCode** basati su **numeroVolo**.

Sistema Prenotazione Voli in Java

Interfaccia Persona: definisce una persona generica

- Un metodo **String getId()** che ritorna un identificativo univoco.
- Un metodo **String getNomeCompleto()** che ritorna il nome completo.

Sistema Prenotazione Voli in Java

Classe Passeggero: implementa Persona

- Implementa **Persona**.
- Attributi: **codiceFiscale** (String), **nome** (String), **cognome** (String).
- Un costruttore che accetta il codice fiscale, nome e cognome.
- Implementazione di **getId()** (ritorna **codiceFiscale**) e **getNomeCompleto()** (ritorna "nome cognome").
- Ridefinizione di **equals** e **hashCode** basati su **codiceFiscale**.

Sistema Prenotazione Voli in Java

Classe Prenotazione: rappresenta una prenotazione effettuata da un passeggero per un volo

- Attributi: **idPrenotazione** (String, univoco), **volo** (Volo), **passeggero** (Passeggero), **postoAssegnato** (String), **dataOraPrenotazione** (java.time.LocalDateTime).
- Un costruttore che accetta ID prenotazione, volo, passeggero, posto assegnato.
- Metodi getter per tutti gli attributi.
- Ridefinizione di **equals** e **hashCode** basati su **idPrenotazione**.

Sistema Prenotazione Voli in Java

Classe SistemaPrenotazioni (1/2): gestisce l'insieme dei voli disponibili e delle prenotazioni effettuate

- Attributi: **voliProgrammati** (una Map<String, Volo>, chiave: numeroVolo), **prenotazioniEffettuate** (una Map<String, Prenotazione>, chiave: idPrenotazione).
- Un costruttore che inizializza le mappe.
- I metodi **void aggiungiVolo(Volo v)** e **Volo getVolo(String numeroVolo).**
- Un metodo **Prenotazione creaPrenotazione(String numeroVolo, Passeggero p, String postoDesiderato)** che cerca il volo tramite **numeroVolo** e lancia **NoSuchElementException** se non trovato.
  - Verifica se ci sono posti disponibili sul volo (`getPostiDisponibili() > 0`). Se non ci sono, lancia una **IllegalStateException.**
  - Tenta di prenotare il **postoDesiderato** sul volo. Se fallisce (posto non valido o già occupato), lancia una **IllegalArgumentException.**
  - Genera un ID univoco per la prenotazione.
  - Crea un nuovo oggetto **Prenotazione.**
  - Aggiunge la prenotazione alla mappa **prenotazioniEffettuate.**
  - Restituisce la prenotazione creata.

Sistema Prenotazione Voli in Java

Classe SistemaPrenotazioni (2/2): gestisce l'insieme dei voli disponibili e delle prenotazioni effettuate

- Un metodo **boolean cancellaPrenotazione(String idPrenotazione)**:
  - Trova la prenotazione tramite **idPrenotazione**. Se non trovata, ritorna **false**.
  - Rimuove la prenotazione dalla mappa **prenotazioniEffettuate**.
  - Libera il posto sul relativo volo.
- Un metodo **List<Prenotazione> getPrenotazioniVolo(String numeroVolo)** che ritorna una lista di tutte le prenotazioni per un dato volo.
- Un metodo **List<Volo> cercaVoli(String codiceIATA_Partenza, String codiceIATA_Arrivo, java.time.LocalDate data)** che ritorna una lista di voli disponibili per una data rotta aerea in una data specifica, ordinati per orario di partenza.