ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

Προγραμματισμός Υπολογιστών με Java
Εαρινό Εξάμηνο 2024-2025
1ο Μέρος Εργασίας

🎬 Σενάριο / Κίνητρο

Μια διαδικτυακή πλατφόρμα επιθυμεί να οργανώσει τις ταινίες του καταλόγου της με στόχο την καλύτερη κατανόηση των προτιμήσεων των χρηστών και τη βελτίωση της εμπειρίας αναζήτησης και σύστασης. Οι χρήστες μπορούν να καταχωρούν αξιολογήσεις και σύντομα σχόλια για τις ταινίες που έχουν παρακολουθήσει.

Γενικότερα, η πλατφόρμα θέλει να απαντά ερωτήματα όπως τα παρακάτω:

- Ποιες είναι οι ταινίες με την υψηλότερη βαθμολογία σε κάθε είδος;
- Πώς μπορεί να εντοπίσει παρόμοιες ή σχετικές ταινίες για προτάσεις;
- Ποιοι χρήστες έχουν αξιολογήσει συγκεκριμένες ταινίες;

Για την υποστήριξη των παραπάνω, η εταιρεία ζητά την ανάπτυξη ενός αντικειμενοστραφούς μοντέλου που να αναπαριστά τις βασικές οντότητες (ταινίες, χρήστες, κριτικές), να επιτρέπει την καταχώρηση και ανάκτησή τους, και να παρέχει λειτουργικότητες αναζήτησης, φιλτραρίσματος και κατάταξης.

Σκοπός

Η παρούσα εργασία έχει ως στόχο την ανάπτυξη ενός αντικειμενοστραφούς συστήματος για την αναπαράσταση, οργάνωση και βασική επεξεργασία πληροφορίας σχετικής με ταινίες και τις κριτικές που λαμβάνουν. Οι φοιτητές καλούνται να σχεδιάσουν και να υλοποιήσουν ιεραρχίες κλάσεων, να αξιοποιήσουν βασικές συλλογές της Java, καθώς και να αναπτύξουν λειτουργικότητες αναζήτησης, φιλτραρίσματος και κατάταξης.

Περιγραφή

Το σύστημα θα αναπαριστά πληροφορία σχετική με ταινίες, κριτικές χρηστών και βαθμολογίες, με σκοπό τη διερεύνηση ερωτημάτων όπως:

- Ποιες είναι οι κορυφαίες ταινίες ανά είδος;
- Ποια είναι η μέση βαθμολογία μιας ταινίας;
- Ποιες ταινίες συνδέονται με κάποια συγκεκριμένη βάσει του είδους;
- Ποιοι χρήστες έχουν αξιολογήσει συγκεκριμένες ταινίες;

Τα δεδομένα θα εισαχθούν προγραμματιστικά, χωρίς χρήση αρχείων.

Απαιτήσεις Σχεδίασης

Η υλοποίηση θα πρέπει να περιλαμβάνει τουλάχιστον τις εξής βασικές κλάσεις:

`Movie`

- Τίτλος
- Έτος κυκλοφορίας
- Είδος/είδη (List<String>)
- Σκηνοθέτης
- Συλλογή κριτικών (List<Review>)
- Συλλογή σχετικών ταινιών (List<Movie>)

`Review`

- Συνδεδεμένος χρήστης (User)
- Βαθμολογία (ακέραιος 1–10)
- Προαιρετικό σχόλιο
- Αναφορά στην ταινία που αξιολογείται

`User`

- Όνομα χρήστη
- Λίστα κριτικών που έχει υποβάλει

Η υλοποίηση μπορεί να επεκταθεί με επιπλέον ιδιότητες (π.χ. γλώσσα, διάρκεια, ηθοποιοί) ή μεθόδους, εφόσον αυτό εξυπηρετεί λειτουργικότητες φιλτραρίσματος, ομαδοποίησης ή προβολής πληροφορίας.

Ενδεικτικές Λειτουργικότητες

- Καταχώρηση νέας ταινίας, κριτικής και χρήστη
- Υπολογισμός μέσης βαθμολογίας ανά ταινία
- Αναζήτηση ταινιών βάσει τίτλου, είδους ή βαθμολογίας
- Εμφάνιση κορυφαίων ταινιών ανά είδος (με τουλάχιστον N κριτικές και μέση βαθμολογία > X)
- Συσχέτιση ταινιών (ίδιο είδος, κοινός σκηνοθέτης κ.ά.)
- Προβολή πληροφοριών με μορφοποιημένο τρόπο μέσω toString().

Πολυμορφισμός & Επεκτασιμότητα

Η υλοποίηση της εφαρμογής πρέπει να αξιοποιεί τον πολυμορφισμό μέσω ιεραρχιών κλάσεων ή διεπαφών. Συγκεκριμένα:

Υποχρεωτικά:

- Η κλάση Review να υλοποιείται ως **αφηρημένη κλάση** ή διεπαφή, με τουλάχιστον δύο υλοποιήσεις (π.χ. BasicReview, VerifiedReview) που διαφέρουν στη μέθοδο getWeightedRating(). Η τελευταία ορίζεται στην αφηρημένη κλάση Review και υλοποιείται από κάθε υποκλάση με τρόπο που αντανακλά τη «βαρύτητα» ή αξιοπιστία της κριτικής. Χρησιμοποιείται για τον υπολογισμό της μέσης βαθμολογίας μιας ταινίας και επιτρέπει την εφαρμογή πολυμορφισμού. Για παράδειγμα, η BasicReview επιστρέφει την απλή βαθμολογία (return rating) ενώ η VerifiedReview μπορεί να επιστρέφει ενισχυμένη βαθμολογία (π.χ., return (int)Math.round(rating * 1.2)).
- Η κλάση User να μπορεί να επεκτείνεται (π.χ. VerifiedUser, CriticUser) με διαφοροποιημένη συμπεριφορά. Δηλαδή, να μπορεί να κληρονομηθεί, να προσφέρει μεθόδους που μπορούν να επανα-οριστούν (override) στις υποκλάσεις, και να επιτρέπει την αντικατάσταση συμπεριφοράς χωρίς να αλλάζει η βασική δομή
- Οι κλάσεις Movie, Review, User να μπορούν να υλοποιούν κοινή διεπαφή (π.χ. Printable) για κοινή μέθοδο προβολής (printDetails()).

- Να χρησιμοποιηθούν διαφορετικοί **συγκριτές (Comparator<Movie>)** για ταξινόμηση βάσει διαφορετικών κριτηρίων (μέση βαθμολογία, έτος κ.ά.).

Αξιολογείται η κατανόηση της έννοιας του πολυμορφισμού και η ικανότητα σχεδίασης επεκτάσιμων ιεραρχιών.

Οδηγίες

Η εργασία είναι **ομαδική** και μπορεί να υλοποιηθεί από ομάδες **δύο ή τριών φοιτητών**. Παρακάτω, δίνεται ένα βασικό προσχέδιο πηγαίου κώδικα (code template).

Παρακαλούμε ακολουθείστε τις παρακάτω οδηγίες:

1. Η εργασία να παραδοθεί ως Java project με:
   1.1. Τεκμηριωμένο πηγαίο κώδικα
   1.2. `README` με περιγραφή σχεδίασης, λειτουργικότητας και οδηγιών εκτέλεσης
   1.3. Demo στο `Main.java` με αντιπροσωπευτικά παραδείγματα λειτουργίας

2. **Σύνθεση ομάδας**: Τα μέλη μια ομάδας μπορούν να προέρχονται είτε από το τμήμα Α-Λ, είτε από το Μ-Ω, είτε και από τα δύο τμήματα (Α-Λ και Μ-Ω). **Η σύνθεση των ομάδων πρέπει να είναι η ίδια και για τις δύο ομαδικές εργασίες (1ο μέρος και 2ο μέρος).**

3. **Δήλωση ομάδας:** Δεν χρειάζεται να γίνει εκ των προτέρων δήλωση των μελών της ομάδας. Πρέπει όμως να γράψετε ως **σχόλιο**, στην αρχή του αρχείου (**Main**) που περιέχει την κύρια μέθοδο, τα **προσωπικά στοιχεία** (επώνυμο, όνομα, αριθμό μητρώου και διεύθυνση

ηλεκτρονικού ταχυδρομείου) **ΟΛΩΝ ΤΩΝ ΜΕΛΩΝ ΤΗΣ ΟΜΑΔΑΣ**. Το επώνυμο και το όνομα σας παρακαλούμε να το σημειώσετε με λατινικούς χαρακτήρες.

4. **Υλοποίηση εργασίας:** Κατά την υλοποίηση της εργασίας σας, παρακαλούμε να λάβετε υπόψη τα παρακάτω:

   4.1. Ονομάστε την κλάση που περιέχει τη κύρια μέθοδο της εφαρμογής σας **Main** και το αρχείο **Main.java**. Μην ξεχάσετε εδώ να σημειώσετε ως σχόλιο τα προσωπικά στοιχεία όλων των μελών της ομάδας σας.
   4.2. **Μη χρησιμοποιείτε ελληνικούς χαρακτήρες** ούτε ως σχόλια, ούτε ως μηνύματα προς το χρήστη.
   4.3. **Μην ομαδοποιήσετε** τις κλάσεις σας σε **πακέτα κλάσεων**.
   4.4. Ο κώδικας που θα υποβάλετε θα πρέπει να μεταγλωττίζεται και να εκτελείται από τη γραμμή εντολών, **χωρίς τη χρήση κάποιου περιβάλλοντος (IDE).**
   4.5. **Συμπιέστε τα αρχεία** της εφαρμογής σας (**java files**) σε ένα αρχείο με όνομα τους αριθμούς μητρώου των μελών της ομάδας, για παράδειγμα (3230001_3230002_3230003.zip). Υποβάλετε το συμπιεσμένο αρχείο στο eclass.

5. **Υποβολή εργασίας**: **Κάθε μέλος** της ομάδας **υποβάλλει ατομικά** ένα **αντίγραφο** της εργασίας στο **eclass**. Δηλαδή, αν για παράδειγμα μια ομάδα αποτελείται από τρεις φοιτητές τότε και οι τρεις πρέπει να υποβάλλουν την – ίδια – εργασία στο eclass (ο κάθε ένας στο τμήμα που ανήκει Α-Λ ή Μ-Ω). Σε κάθε αντίγραφο της εργασίας, παρακαλούμε να αναφέρονται τα προσωπικά στοιχεία όλων των μελών της ομάδας.

Τέλος, υπενθυμίζεται ότι έχετε το δικαίωμα να **διατηρήσετε βαθμό εργασιών που τυχόν έχετε υλοποιήσει σε προηγούμενο ακαδημαϊκό έτος**. Στην περίπτωση αυτή, δεν χρειάζεται να κάνετε τώρα κάτι. Απλά πρέπει να το **σημειώσετε στο γραπτό σας**, στο τελικό διαγώνισμα του Ιουνίου 2025 ή/και του Σεπτεμβρίου 2025.

CODE TEMPLATE

```
// Movie.java
import java.util.*;

public class Movie implements Printable {
    private String title;
    private int year;
    private List<String> genres;
    private String director;
    private List<Review> reviews;
    private List<Movie> relatedMovies;

    public Movie(String title, int year, List<String> genres, String director) {
        this.title = title;
        this.year = year;
        this.genres = new ArrayList<>(genres);
        this.director = director;
        this.reviews = new ArrayList< Review >();
        this.relatedMovies = new ArrayList< Movie >();
    }

    public void addReview(Review r) {
        reviews.add(r);
    }

    public void addRelatedMovie(Movie m) {
        relatedMovies.add(m);
    }

    public double getAverageRating() {
        if (reviews.isEmpty()) return 0;
        int total = 0;
        for (Review r : reviews) {
            total += r.getWeightedRating();
        }
        return (double) total / reviews.size();
    }

    public void printDetails() {
        System.out.println("Title: " + title);
        System.out.println("Year: " + year);
        System.out.println("Genres: " + genres);
        System.out.println("Director: " + director);
        System.out.println("Average Rating: " + getAverageRating());
    }

    // Getters and other utility methods can be added here
}

// Review.java
public abstract class Review implements Printable {
    protected User user;
    protected int rating;
    protected String comment;
    protected Movie movie;

    public Review(User user, int rating, String comment, Movie movie) {
        this.user = user;
        this.rating = rating;
```

```
        this.comment = comment;
        this.movie = movie;
    }

    public abstract int getWeightedRating();

    public void printDetails() {
        System.out.println(user.getUsername() + " rated " + movie.getTitle() + "
with " + rating + "/10");
        if (comment != null && !comment.isEmpty())
            System.out.println("Comment: " + comment);
    }
}

// BasicReview.java
public class BasicReview extends Review {
    public BasicReview(User user, int rating, String comment, Movie movie) {
        super(user, rating, comment, movie);
    }

    public int getWeightedRating() {
        return rating;
    }
}

// User.java
import java.util.*;

public class User implements Printable {
    protected String username;
    protected List<Review> reviews;

    public User(String username) {
        this.username = username;
        this.reviews = new ArrayList<String>();
    }

    public void addReview(Review r) {
        reviews.add(r);
    }

    public String getUsername() {
        return username;
    }

    public void printDetails() {
        System.out.println("User: " + username);
        System.out.println("Reviews submitted: " + reviews.size());
    }
}

// Printable.java
public interface Printable {
    void printDetails();
}

// Main.java
import java.util.*;

public class Main {
    public static void main(String[] args) {
```

```
        Movie m = new Movie("Inception", 2010, List.of("Sci-Fi", "Action"),
"Christopher Nolan");
        User u = new User("alice");

        Review r = new BasicReview(u, 9, "Mind-blowing!", m);
        m.addReview(r);
        u.addReview(r);

        m.printDetails();
        System.out.println();
        u.printDetails();
        System.out.println();
        r.printDetails();
    }
}
```