

Data Structures Project 3

This project is a movie recommendation system. Processes user data and draws appropriate results.

There is a max heap class. This heap has a Node attribute. There is data and user id attributes in Node class.

```
public class MaxHeap {  
    Node[] heap;  
    int size;  
  
    public MaxHeap(int capacity) {  
        heap = new Node[capacity];  
        size = 0;  
    }  
  
    public boolean isEmpty() {  
        return size == 0;  
    }  
  
    public void insert(double value, int id) {  
        if (size == heap.length) {  
            throw new IllegalStateException(s: "Heap is full. Cannot insert more elements.");  
        }  
  
        Node newNode = new Node(data: value, user_id: id);  
        heap[size] = newNode;  
  
        siftUp(index: size);  
        size++;  
    }  
}
```

The insert method, sorts according to the data of the sent node. The user_id keeps the movie id or userID.

There is extractMax method in MaxHeap class. This method extracts Node that has max data from heap.

```
public Node extractMax() {  
    if (isEmpty()) {  
        throw new IllegalStateException(s: "Heap is empty. Cannot extract maximum element.");  
    }  
  
    Node max = heap[0];  
    heap[0] = heap[size - 1];  
    size--;  
    siftDown(index: 0);  
  
    return max;  
}
```

There is a MovieRecommendationSystem class. This class has methods of transferring the sent file to the array.

```
public class MovieRecommendationSystem {

    public int[][] readUserMovieMatrixFromFile(String fileName) {
        int[][] kullaniciFilmMatrisi = null;
        try (BufferedReader br = new BufferedReader(new FileReader(fileName))) {
            List<int[]> rows = new ArrayList<>();
            String line;
            while ((line = br.readLine()) != null) {
                String[] values = line.split(regex: ",");

                int[] row = new int[values.length];
                for (int i = 0; i < values.length; i++) {

                    row[i] = Integer.parseInt(values[i]);
                }
                rows.add(e: row);
            }
            kullaniciFilmMatrisi = new int[rows.size()][];
            for (int i = 0; i < rows.size(); i++) {
                kullaniciFilmMatrisi[i] = rows.get(index: i);
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
        return kullaniciFilmMatrisi;
    }
}
```

This method compares the target user with all the users in the table and assigns the cosine similarity value to the heap. Then, the cosine values in the heap are extracted with the extract method and the most matching users in the table with the target user are written to the String text variable.

```
private void btn_recommendationActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String text = "";

    heap = new MaxHeap(capacity: movieMatris.length);
    int x_gui = Integer.parseInt(s: label_x.getText());
    int k_gui = Integer.parseInt(s: label_k.getText());

    int[] targetUser = createUserArray(matrix: targetUserMatris, cmb_users.getSelectedIndex() + 1);
    //heap.insertHeap(targetUser);
    for (int i = 1; i < movieMatris.length; i++) {
        double cosine = m.cosineSimilarity(kullanici1: targetUser, kullanici2: createUserArray(matrix: movieMatris, row: i));
        heap.insert(value: cosine, id: i);
    }

    ArrayList<Integer> users_id = new ArrayList<>();
    for (int i = 0; i < x_gui; i++) {
        Node maxNode = heap.extractMax();
        double maxNode.data;
        int maxID = maxNode.user_id;
        text += "Id = " + maxID + " Cosine : " + maxNode.data + "\n";
        users_id.add(e: maxID);
    }

    text += "-----\n";
}
```

This loop sends an array list of the ids of the most matched users and this array sends all the movie scores and movie ids of each of the users in the list to the heap. Afterwards, the scores in this heap are extracted with the extract method and the favorite movies of the users are written to the text variable. Then it is transferred to the text area component with the set text method.

```

text += "-----\n";
for (int i = 0; i < users_id.size(); i++) {
    for (int j = 1; j < movieMatrix.length; j++) {
        if(movieMatrix[j][0] == users_id.get(index:i)){
            MaxHeap users_movies = new MaxHeap(capacity:movieMatrix[0].length);
            for (int k = 1; k < movieMatrix[j].length - 1; k++) {
                if (k == 0) {
                    System.out.println(x: movieMatrix[j].length);
                }
                users_movies.insert(movieMatrix[j][k], id: k);
            }
            text += "ID = " + users_id.get(index: i) + " Best Movies" + "\n";
            for (int k = 0; k < k_gui; k++) {
                int index = users_movies.extractMax().user_id;
                for (int l = 1; l < movies.length; l++) {
                    if(Integer.parseInt(movies[l][0]) == index){
                        text += "Movie " + (k+1) + ": " + movies[l][1] + "\n";
                    }
                }
            }
            break;
        }
    }
    text += "-----\n";
}

text area.setText(t: text);

```

In this method, movies are sent to a node array and the given scores are recorded. These scores are then recorded in an array. This array is compared with all users in the list and the users with the highest cosine value are printed on the screen.

```

// TODO add your handling code here:
String text = "";

Node[] movieID = new Node[5];
movieID[0] = new Node(data:Double.parseDouble(s: txt_movie1.getText()), user_id: switchCase(index: cmb_movie1.getSelectedIndex()));
movieID[1] = new Node(data:Double.parseDouble(s: txt_movie2.getText()), user_id: switchCase(index: cmb_movie2.getSelectedIndex()));
movieID[2] = new Node(data:Double.parseDouble(s: txt_movie3.getText()), user_id: switchCase(index: cmb_movie3.getSelectedIndex()));
movieID[3] = new Node(data:Double.parseDouble(s: txt_movie4.getText()), user_id: switchCase(index: cmb_movie4.getSelectedIndex()));
movieID[4] = new Node(data:Double.parseDouble(s: txt_movie5.getText()), user_id: switchCase(index: cmb_movie5.getSelectedIndex()));

int [] newUser = new int[movieMatrix[1].length];

for (int i = 0; i < newUser.length; i++) {
    boolean bool = false;
    for (int j = 0; j < movieID.length; j++) {
        if(i == movieID[j].user_id){
            newUser[i] = (int) movieID[j].data;
            bool = true;
            break;
        }
    }
    if(!bool) {
        newUser[i] = 0;
    }
}

MaxHeap h = new MaxHeap(capacity:movieMatrix.length);

for (int i = 1; i < movieMatrix.length; i++) {
    int[] arr = createUserArray(matrix: movieMatrix, row: i);
    double cosine = m.cosineSimilarity(kullanici1: newUser, kullanici2: arr);
    h.insert(value: cosine, id: i);
}

```

The output of the project is below.

The screenshot shows a Java Swing window titled "Movie Recommendation System". It has a light gray background and standard window controls (minimize, maximize, close) in the top right corner.

Top Section:

- Target User:** A dropdown menu showing "User 5".
- X:** A text input field containing the number "4".
- K:** A text input field containing the number "3".
- Get Recommendation:** A button.

Output Area (Top):

Id = 391 Cosine : 0.07567180561089365
Id = 553 Cosine : 0.06951247259605346
Id = 151 Cosine : 0.0650472071839427
Id = 132 Cosine : 0.06095614063117699

ID = 391 Best Movies
Movie 1: Dead Presidents (1995)
Movie 2: Dunston Checks In (1996)
Movie 3: Thirty-Two Short Films About Glenn Gould (1993)

ID = 553 Best Movies

Input Section (Bottom Left):

Georgia (1995)	5
Underneath (1995)	5
Frenzy (1972)	3
Clueless (1995)	1
Frenzy (1972)	4

Get Recommendation: A button.

Output Area (Bottom Right):

En çok eşleşen kullanıcılar.
ID = 332 cosine = 0.14623249474027014
ID = 479 cosine = 0.09558988911273404
ID = 385 cosine = 0.09419553888434772
ID = 26 cosine = 0.07484811885651198
ID = 584 cosine = 0.060663650634803626
ID = 364 cosine = 0.058952864412411494
ID = 525 cosine = 0.05737910526322038
ID = 182 cosine = 0.055901699437494734
ID = 397 cosine = 0.04709686404196994
ID = 486 cosine = 0.040422604172722164

ALİ HAYDAR OSMANOĞLU

2121221033