**תיאור מבני הנתונים:**

מבני הנתנוים שלנו מורכב מ 8 עצי AVL שכל אחד מהם אחראי על תפקיד מסויים:

1. All Movies ID: העץ הזה מכיל בתוכות את כל הסרטים במערכת שלנו, והולכים להיות מסודרים לפי מספר ה ID של כל סרט (העץ הזה הולך להכיל את כל הסרטים מכל ה Genres).
2. All Movies Rating: העץ הזה מכיל גם את כל הסרטים במערכת שלנו אבל דרך הסידור היא לפי הדירוג של העץ (ובמקרים שיש את איתו דירוג סידרנו לפי ההנחיות למצב הזה).
3. Comedy/Drama/Action/Fantasy: ארבעת העצים הם כמו העץ הקודם אבל שכל עץ יכיל רק את הסרטים מה Genre המתאים והם מסודרים לפי הדירוג שלהם.
4. All Users: העץ הזה מכיל את כל המשתמשים במערכת שלנו והם מדורגים לפי ה ID שלהם.
5. All Groups: העץ הזה מכיל את כל הקבוצות של משתמשים במערכת שלנו, והקבוצו מדורגות לפי ה ID שלהן.

העצים האלו מבוססים על AVLNodes שבנינו, הנודים שלנו הם גינירים (בנויים עם Templates ) על מנת שיתאימו להרבה סוגים של מידע, הם מקבלים שני דברים:

Key – שהוא אחראי על סידור הנודים בתוך העץ.

Data- שהוא מכיל את המידע בתוך הנוד.

**בנינו Classes שיתאימו למידע שנצרך לשמור ואיך לסדר איתו, תיאור לה Classes שלנו:**

1. **Movies Ranking Key:** הוא מכיל את ה ID, דירוג, צפיות של הסרט, בתוך הקלאס עשינו העמסה של אופירטורים על מנת שנוכל לעשות השוואה בין הסרטים השונים כשההשוואה מתבססת על ה דירוג.
2. **Movie Data:** הוא מכיל את כל המידע על הסרט: ID, דירוג, Genre, צפיות, VIP/Not VIP.

* כל העצים שמכילים את הסרטונים הולכים להכיל את הקלאס של Movie Data וה Key שלהם או שהולך להיות int רגיל במקרה של All Movies ID או הקלאס Movies Ranking Key בשאר העצים של הסרטים.

1. **User Data:** קלאס שמכיל את המידע שלהמשתמשים שלנו, הוא מכיל את ה ID של המשתמש, את הצפיות שלו שעשה לבד בכל Genre בתוך Array בגודל 5 שארבעת התאים הראשונים מתאר Genre מסויים לפי הסדר שנתון לנו בתרגיל הבית, והתא האחרון מתאר את הצפיות בכל סוגי הסרטים לבד, בנוסף לזה יש לו עוד Array בגודל 5 עם אותו מבנה שהולך להחזיק את הצפיות שלו בתוך קבוצה, בנוסף לזה יש לנו פרמטר אם המשתמש

VIP/Not VIP, ומפסר קבוצה במקרה שהוא שייך לאחת (ואם לא המפסר הזה הוא אפס, כי נתון שאין מספר קבוצה 0 לכן השתמשנו במספר הזה כדי לעשות אינדיקציה שהמשתמש לא בקבוצה).

1. **Group Data:** קלאס שמכיל את המידע של הקבוצה, הוא מכיל את המספר של הקבוצה, וגם אם הקבוצה VIP/Not VIP, כמות האנשים בקבוצה, כמות האנשים ה VIP בקבוצה אם יש, מערך בגודל 5 שיכיל את מפסר הצפיות של הקבוצה בכל Genre ובכל הסרטים (אותו מבנה של מערך כמו לפני) , בנוסף לזה הוא מכיל עוד עץ של משתמשים שבתוכו מאחסנים את המשתמשים ששייכים להקבוצה הזאת והם מסודרים לפי ה ID שלהם.

**הסבר להפעולות המבוקשות עם ניתוח סיבוכיות זמן:**

**\*\* בכל הפונקציות שיש התעסקות עם זיכרון דינמי בדקנו אם הכל עבר כפי שרוצים ואם לא החזרנו StatusType של Allocation\_error**

1. **Streaming\_database():**

הפעולה הזאת מאתחלת המבני שלנו של השמונה עצים, כל העצים ריקים ואתחות כל עץ הוא בסיבוכיות .

1. **Virtual ~streaming\_database()**
2. **Add\_movie:**

בפונקציה הזאת מקבלים מספר של סרט, את ה Genre שלו, צפיות ואם הוא VIP, דבר ראשון בודקים אם ה Input חוקי או לא, אחרי זה מחפשים על הסרט בהעץ הגדול All Movies ID , מכיוון שהעץ הזה הוא AVL לכן כפי שלמדנו הסיבכיות המקסימלית הולכת להיות כאשר K הוא מפסר הסרטים במערכת, אם מוציאם אז שולחים Failure, אחרת עושים insert לתוך העץ של כל הסרטים וגם להעץ של ה Genre המתאימה של הסרט, אחרי הוספת הנוד החדש מתחילים לבדוק איזות של המערכת כך שנשאר בעץ AVL , נתחיל לעלות בנודים ונעדכן את הגבהים שלהם, במקרה שנמצא נוד לא מאוזן נבצע את ה rotation המתאים לפי המקרה (כפי שלמדנו) ,פעולת הסיבוב היא לכן במקרה הגרוע החזרת האיזון תעלה לנו , ואם צריך גם מעדכנים את המצביע שלהנוד הכי גדול בעץ במקרה שסרט הזה הוא עכשיו הכי גדול, במקרה הגרוע הפעולות האלו הולכים לעלות לנו , ולכן ביחד כל הפעולה של הוספת סרט היא בעל סיבוכיות של

והפונקציה הזאת תחיזור StatusType לפי אם הצלחנו או לא (במקרה של לא החזרנו את הערך המתאים כפי שהוסבר בתרגיל הבית), והפונקציה הזאת תחזיר StatusType שהוא Success אם הצלחנו, Failure אם לא הצלחנו, Invalid input אם מקבלים input לא חוקי.

1. **Removie\_movie:**

הפונקציה הזאת מקבלת מספר סרטון, בודקת שהוא חוקי אחרי זה מחפשת עליו בהעץ הגדול של הסרטים, אם מוצאים איתו, נילך לעץ הסרטים עם ה Genre המתאימה לסרט המטרה , מחפשים איתו שם, אחרי זה מוחקים איתו משם, כאשר מוחקים יש לנו 3 מקרים, אם הוא LEAF אז רק נמחק, אם יש לו בן אחד אז נעלה את הבן שלו למקומו, ואם יש לו 2 בנים אז צריך שנמצא לו מחליף ועשינו את זה על ידי הליכה פעם אחד ימינה אחרי זה רק שנאלה עד שנגייע ל LEAF ונחיף אותה בהנוד שנצרך למחוק ואחרי זה נמחק את סרט המטרה, הפעולה הזאת של החיפוש והמחקיה במקרה הכי גרוע תעלה לנו כאשר K הוא מפסר הסרטים במערכת שלנו( זה יקרה כאשר כל הסרטים מאיתו סוג), אחרי זה נתחיל לעלות בנודים ונעדכן את הגובה שלהם ובמקרה שהשתבש האיזון נבצע הסיבוב המתאים, ובמקרה הגרוע זה הולך לעלות לנו , אחרי זה נחזור להעץ של כל הסרטים וגם נמחק את הסרט משם באיתו דרך כפי שעשינו לפני, כל הפעולות האלה יעלו לנו במקרה הגרוע שזה סיבכיות זמן של , הפונקציה הזאת תחזיר StatusType כפי שהוסבר בפונקציה לפני.

1. **Add\_user:**

הפונקציה הזאת מוסיפה משתמש חדש למערכת שלנו, היא מתנהגת אותו דבר כמו הוספת סרט חדש והפעם יש לנו רק עץ אחד שנצרך להוסיף לו, סדר הפעילות כמו שהוסבר לעיל ולכן במקרה הגרוע הפעולה הזאת תעלה לנו כאשר n הוא מפסר המשתמשים במערכת שלנו שכולם בתוך עץ ה AVL של המשתמשים, הפונקציה הזאת תחזיר StatusType כפי שהוסבר בפונקציה לפני.

1. **Remove\_user:**

הפונקציה האזת מוחקת את המשתמש מהמערכת שלנו, היא מחפשת על המשתמש בהעץ של המשתמשים אם היא מוציאה איתו בודקים אם הוא שייך לקבוצה, אם כן שייך אז ניקח את מספר הקבוצה שלו ונלך לחפש עליו בעץ הקבוצות ומשם נמחק את המשתמש המקבוצה ונעדכן את מצב הקבוצה, ואחרי זה נמחק את המשתמש מעץ המשתמשים, מכיוון שכל משתמש יכול להיות שייך לקבוצה אחד לכן מספר הקבוצות המקסימלי שיכול להיות הוא n לכן כל הפונקציה שלנו היא בעלת סיבוכיות זמן של , (כאשר מוחקים נעשה את זה כפי שהוסבר שמוחקים סרט) , הפונקציה הזאת תחזיר StatusType כפי שהוסבר בפונקציה לפני.

1. **Add \_group:**

הפונקציה האלו כמו הוספה ו של משתמש אבל הפעם ה דאטא היא קלאס של group , והיא בעלת סיבוכיות זמן של ומחזירה StatusType כפי שהוסבר בפונקציה לפני.

1. **Remove\_group:**

מחיקת קבוצה היא כמו מחיקת משתמש אבל בנוסף לזה לפני מחיקת הקבוצה נעבור על כל המשתמשים בה ונעדכן שהם עכשיו לא שייכים לקבוצה, נכול לעשות את זה כי יש לנו מצביעים על המידע של כל משתמש והמידע הזה הוא גם המידע שמוצבע עליו מתוך העץ של המשתמשים (ולכן "גם ישנה שם"), סביוכיות הזמן של הפןנקציה הזאת היא הסיבוכיות הרגילה של מחיקה בעת AVL כמו שהיסברנו לפני בנוסף לזה יש צורך לעבור על משתמש מתשמש בתוך הקבוצה לכן סכ"כ יהי לנו כאשר m הוא מפסר הקבוצות במערכת שלנו, ומחזירה StatusType כפי שהוסבר בפונקציה לפני.

1. **Add\_user\_to\_group:**

הפונקציה מקבל כ input את מספר המשתמש ומפסר קבוצה, בודקת שהם חוקים, אחרי זה היא מחפשת על המשתמש בעץ המשתמשים ובודקת שהוא לא שייך לקבוצה, אם הוא לא שייך לקבוצה היא מחפשת על הקבוצה בעץ הקבוצות ומוסיפה איתו להקבוצה (בתוך הקבוצה יש לנו עץ שמכיל מצבעים להמשתמשים ששייכים לקבוצה הזאת), במקרה שהמשתמש כבר שייך לקבוצה או שהקבוצה לא קיימת הפונקציה מחזירה Failure , ואם הצלחנו להוסיף המשתמש לקבוצה תחזיר success, בנוסף דאגנו לבדוק אם מצב הקבוצה השתנה מ not vip ל vip,

הסיבוכיות שלנו היא החיפוש על המשתמש בעץ המשתמשים,חיפוש על הקבוצה בעץ הקבוצות ולעושת insert להמשתמש לתוך עת המשתמשים השייכים לקבוצה, ומקרה הגרוע זה יעלה לנו

כאשר m מספר הקבוצות ו n מספר המשתמשים זה סיבוכיות זמן של , ומחזירה StatusType כפי שהוסבר בפונקציה לפני.