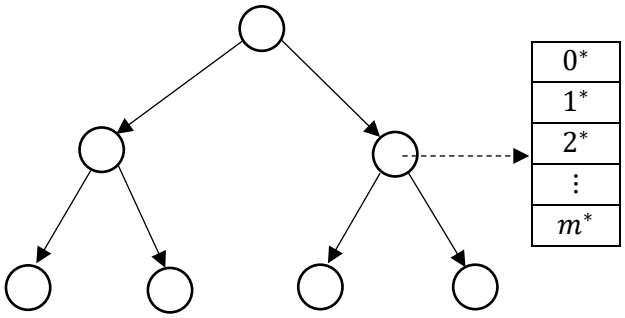
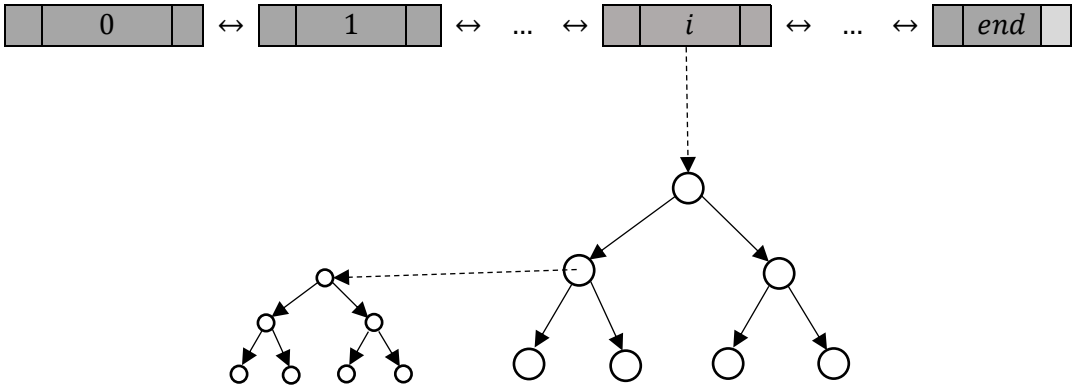
מבנה נתונים

1. $Total\_num\_of\_songs$ – כמות השירים הכוללת של כל האומנים.
2. $musicDict$ – עץ AVL המכיל בתוכו אמנים. מכל אמן יש מצביע למערך בגודל כמות השירים בשם $artistsSongs$ וערך $numOfSongs$ שאומר מה היא כמות השירים של האמן. כל איבר ב $artistsSongs$ מכיל מצביע לאיבר ברשימה המקושרת במבנה $Playbacks$.



3. $Playbacks$ – רשימה מקושרת $numOfPlaybacks$ בה כל איבר מכיל עץ AVL בשם $PlaybacksArtists$ ומספר שלם אי-שלילי i בסדר עולה. כל איבר בעץ $PlaybacksArtists$ מכיל מספר אמן j ועץAVL בשם $PlaybacksSongs$ המכיל את כל השירים של האמן j שהושמעו i פעמים.



תוספות לעצי AVL במבנה הנתונים

עצי AVL בנויים כך שכל איבר מצביע ל 5 איברים:

1. אב
2. בן ימני
3. בן שמאלי
4. איבר הבא אחריו בעץ בסדר עוקב
5. לאיבר הקודם בעץ בסדר עוקב

בנוסף, כל עץ ישמור:

1. מצביע לראש העץ
2. מצביע לראש הרשימה (האיבר הקטן ביותר בעץ)

נקרא למצביעי האיבר הבא, האיבר הקודם וראש הרשימה בשם מצביעי הרשימה של העץ.

נסביר בקצרה איך הוספת מצביעי הרשימה מתבצעים ב $O(1)$ ולכן לא פוגעים ביעילות הפעולות הנעשות על העץ.

<u>עדכון מצביעי הרשימה בהכנסת איבר לעץ AVL</u>

1. הכנסת האיבר לעץ (עדיין בלי לאזן)
2. אם האיבר הוא יחיד
   a. נעדכן את כל המצביעים ל $NULL$, וראש הרשימה יהיה האיבר שהוכנס וסיימנו.
3. אם יש עוד איברים בעץ
   a. נחלץ מהאב את המצביעים קדימה ואחורה ונעדכן את המצביעים של האיבר החדש באותו אופן כמו ברשימה מקושרת דו כיוונית רגילה.
   b. אם האיבר שהוכנס הוא הבן הימני
      i. האיבר החדש יכנס "אחרי" האב (במובן של רשימה מקושרת)
   c. אם האיבר שהוכנס הוא הבן השמאלי
      i. האיבר החדש יכנס "לפני" האב (במובן של רשימה מקושרת)
      ii. אם האב היה גם ראש הרשימה אז האיבר החדש הופך לראש הרשימה.

אחרי שינוי המצביעים של הרשימה בעץ נוכל להמשיך כרגיל עם איזון העץ ושינוי מצביעי האב והבנים כרגיל כפי שראינו בהרצאה.

כל הפעולות לוקחות $O(1)$ ולא משנות את הפעולות של ההכנסה של איבר חדש לעץ AVL רגיל.

<u>עדכון מצביעי הרשימה בהוצאת איבר מעץ AVL</u>

1. נבצע את פעולת הוצאת האיבר מהעץ ואיזונו.
2. לפני מחיקת האיבר נוציא אותו מהרשימה בעזרת המצביעים שלו קדימה ואחורה באותו אופן כמו בהוצאת איבר מרשימה מקושרת דו כיוונית (איזון העץ לא משפיע על המצביעים של הרשימה בעץ)
3. נמחק את האיבר.

כל הפעולות לוקחות $O(1)$ ולא משנות את הפעולות של הוצאת איבר מעץ AVL רגיל.

<u>עדכון מצביעי הרשימה בזמן מילוי עץ AVL בעזרת רשימה ממוינת</u>

אחרי שיצרנו את העץ בגודל המתאים כפי שראינו בהרצאה, האיברים מוכנסים לעץ $inorder$ לפי $inorder$ רגיל על עץ AVL. כל איבר שמוכנס נעדכן את המצביע לאיבר הקודם ונעדכן את המצביע קדימה של האיבר הקודם מתחילת הרשימה עד סופה. האיבר הראשון יהיה ראש הרשימה ויצביע אחורה ל $NULL$ והאיבר האחרון ברשימה יצביע ל $NULL$.

עדכון מצביעי הרשימה בכל איבר בעץ לוקח מספר קבוע של פעולות כלומר $O(1)$, כאשר במילוי עץ גם ככה עוברים על כל האיברים ומבצעים מספר קבוע של פעולות כפי שראינו בהרצאה. לכן סיבוכיות הזמן של הפעולה לא משתנה.

<u>מעבר $inorder$ על איברי עץ AVL בעזרת מצביעי הרשימה</u>

נשתמש במצביע לראש הרשימה ובעזרת המצביעים לאיברים שאחרי נוכל לעבור על כל איברי הרשימה.

פעולה זו לוקחת $O(m)$ כתלות בכמות האיברים m עליה נרצה לעבור ללא חשיבות לגודל העץ.

מימושים

בכל הפעולות במקרה של בעיה בהקצאת זיכרון יוחזר $allocation\_error$ ונצא מהפעולה.

$void * Init()$

<u>פעולות</u>

1. אתחול מבנה $musicDict$ ללא אמנים.
2. אתחול מבנה $Playbacks$ עם איבר 0 ועץ ריק.
3. אתחול $Total\_num\_of\_songs$ ל 0.

<u>סיבוכיות זמן</u>

כפי שראינו בהרצאה אתחול כל מבנה לוקח $O(1)$ ולכן סיבוכיות הזמן היא $O(1)$.

$StatusType\ AddArtist(\ void\ *\ DS, int\ artistID, int\ numOfSongs)$

<u>פעולות</u>

1. החזרת $invalid\_input$ אם $artistID <= 0 \mid\mid DS == NULL \mid\mid numOfSongs <= 0$
2. החזרת $failure$ אם האמן כבר קיים (חיפוש האמן בתוך $musicDict$).
3. הגדלת $Total\_num\_of\_songs$ בערך של $numOfSongs$.
4. הכנסת איבר $artistID$ לתוך $musicDict$
5. עדכון $numOfSongs$ של האמן.
6. הוספת מערך $artistsSongs$ לאיבר שהוכנס בסעיף הקודם בגודל $numOfSongs$
7. כל איבר במערך יצביע לאיבר 0 ברשימה $numOfPlaybacks$
8. הכנסת איבר חדש $artistID$ לתוך $PlaybacksArtists$ ברשימה $numOfPlaybacks$ באיבר 0.
9. בתוך האיבר שהוכנס מהסעיף הקודם נבנה את $PlaybacksSongs$ ריק עם כמות צמתים השווה ל $numOfSongs$.
10. נמלא את $PlaybacksSongs$ שיצרנו לפי $numOfSongs$.

<u>סיבוכיות זמן</u>

1. $O(1)$ - כמות קבועה של פעולות.
2. $O(\log n)$ – כפי שראינו בהרצאה (חיפוש בעץAVL).
3. $O(1)$ - כמות קבועה של פעולות.
4. $O(\log n)$ – כפי שראינו בהרצאה (הכנסה לעץ AVL) וכפי שהוסבר במבנה הנתונים.
5. $O(1)$ - כמות קבועה של פעולות.
6. $O(m)$ – כפי שראינו בהרצאה (אתחול מערך)
7. $O(m)$ – נבצע m פעולות שכל אחת מהן לוקחות $O(1)$
8. $O(\log n)$ – כפי שראינו בהרצאה (הכנסה לעץ AVL).
9. $O(m)$ – כפי שראינו בהרצאה (בניית עץ AVL ריק בגודל m)
10. $O(m)$ – מעבר על כל איבר מספר מוגבל של פעמים בהכנסה כפי שראינו בהרצאה (מילוי עץ AVL בעזרת מערך ממוין) וכפי שהוסבר במבנה הנתונים.

כל הפעולות בוצעו זו אחר זו בנפרד לכן

$$O(\log n) + O(m) + O(m) + O(\log n) + O(m) + O(m) = O(m + \log n)$$

$StatusType\ RemoveArtist(\ void\ *\ DS, int\ artistID)$

<u>פעולות</u>

1. החזרת $invalid\_input$ אם $artistID <= 0 \mid\mid DS == NULL$.
2. החזרת $failure$ אם האמן לא קיים (חיפוש האמן בתוך $musicDict$).
3. הקטנת $Total\_num\_of\_songs$ בערך של $numOfSongs$ של האמן.
4. מעבר על המערך $artistsSongs$ של האמן ולכל איבר נבצע

a. שימוש במצביע כדי להגיע לאיבר ב $Playbacks$ שבו מצוינת כמות ההשמעות של השיר. אם המצביע $NULL$ אז לעבור לאיבר הבא עד סוף המערך ושלבים $b - e$ לא מתבצעים.
b. בתוך האיבר ב $Playbacks$ למצוא את האמן $artistID$ בתוך $PlaybacksArtists$.
c. מחיקת $PlaybacksSongs$ המתאים. בכל מחיקה של שיר מ $PlaybacksSongs$ נעביר את המצביע המתאים מתוך $artistsSongs$ להיות $NULL$.
d. מחיקת איבר האמן מתוך $PlaybacksArtists$.
e. אם באיבר $numOfPlaybacks$ לא נותרו עוד אמנים אז נמחק אותו.

5. מחיקת המערך $artistsSongs$ של האמן.
6. מחיקת האמן מתוך $musicDict$

<u>סיבוכיות זמן</u>

1. $O(1)$ - כמות קבועה של פעולות.
2. $O(\log n)$ – כפי שראינו בהרצאה (חיפוש בעץ AVL)
3. $O(1)$ - כמות קבועה של פעולות.
4. $O(m)$ – מעבר על כל אחד מאיברי המערך בגודל m פעם אחת. כל אחד מהאיברים עובר את הפעולות הבאות:
   a. $O(1)$ – אם האיבר $NULL$ או לא.
   b. $O(\log n)$ – כפי שראינו בהרצאה (חיפוש בעץ AVL)
   c. $O(k)$ – מחיקה כפי שראינו בהרצאה וכפי שהוסבר במבנה הנתונים. במקרה הגרוע ביותר נצטרך למחוק $PlaybacksSongs$ המכיל את כל השירים של אותו אמן. כאשר k זו כמות השירים בעץ בו מופיע השיר אותו אנו רוצים למחוק (סך כל ה k שווים ל m)
   d. $O(\log n)$ – כפי שראינו בהרצאה (מחיקה מעץ AVL) וכפי שהוסבר במבנה הנתונים.
   e. $O(1)$ – כפי שראינו בהרצאה (מחיקה מרשימה מקושרת)
5. $O(m)$ – כפי שראינו בהרצאה (מחיקת מערך)
6. $O(\log n)$ – כפי שראינו בהרצאה (מחיקה מעץ AVL) וכפי שהוסבר במבנה הנתונים..

במעבר על $artistsSongs$ נעבור על אותו שיר אם הגענו אליו במעבר על כל המערך, אם עברנו עליו והוא שווה ל $NULL$ וכאשר נשנה את ערכו ל $NULL$. רק כאשר נעבור עליו והוא לא יהיה $NULL$ אז נבצע עליו פעולות נוספות.
כל שיר ימחק מ $PlaybacksSongs$ כלשהו פעם אחת בלבד.

$$O(\log n) + O(m) \cdot O(\log n) + O(m) + O(\log n)$$

כפי שראינו בכיתה

$$= O(m \log n)$$

$StatusType\ AddToSongCount(\ void\ *\ DS, int\ artistID, int\ songID)$

<u>פעולות</u>

1. החזרת $invalid\_input$ אם
$DS == NULL\ ||\ songID < 0 ||\ songID >= numOfSongs || artistID <= 0$
2. החזרת $failure$ אם האמן לא קיים (חיפוש $artistID$ בתוך $musicDict$)
3. מתוך $artistsSongs$ המתאים נשתמש במצביע במיקום $songID$ כדי לעבור לאיבר ברשימת $numOfPlaybacks$
4. בתוך $PlaybacksArtists$ המתאים נחפש את $artistID$
5. בתוך $PlaybacksSongs$ נמחק את האיבר המתאים ל $songID$
6. אם הוא האיבר היחיד בתוך $PlaybacksSongs$
   a. נמחק את העץ ואת האיבר המתאים בתוך $artistsSongs$
7. אם $numOfPlaybacks$ במיקום הנוכחי הכיל רק את האיבר $artistsSongs$ והוא אינו שווה לאפס
   a. נמחק אותו
8. אם האיבר הבא ברשימת $numOfPlaybacks$ גדול ביותר מאחד מערך האיבר הנוכחי
   a. נכין איבר חדש ב $numOfPlaybacks$ שערכו גדול ב 1 מערך האיבר הנוכחי ונמצא מיד אחריו
9. נעבור לאיבר הבא ברשימת $numOfPlaybacks$

10. אם $PlaybacksArtists$ לא קיים או לא מכיל את $artistID$
    a. אז נוסיף אותם
11. אם $PlaybacksSongs$ לא קיים או לא מכיל את $songID$
    a. נוסיף אותם
12. בתוך $artistsSongs$ נשנה במיקום $songID$ את המצביע כך שיצביע לאיבר הבא ברשימת $numOfPlaybacks$

<u>סיבוכיות זמן</u>

1. $O(1)$ – מספר פעולות קבועות שלוקחות $O(1)$
2. $O(\log n)$ – כפי שראינו בהרצאה (חיפוש בעץ AVL)
3. $O(1)$ – מספר פעולות קבועות שלוקחות $O(1)$
4. $O(\log n)$ – כפי שראינו בהרצאה (חיפוש בעץ AVL)
5. $O(\log m)$ – כפי שראינו בהרצאה (מחיקה מעץ AVL) וכפי שהוסבר במבנה הנתונים.
6. $O(\log n)$ – כפי שראינו בהרצאה (מחיקה מעץ AVL) וכפי שהוסבר במבנה הנתונים.
7. $O(1)$ – כפי שראינו בהרצאה (מחיקה מרשימה מקושרת)
8. $O(1)$ – כפי שראינו בהרצאה (הוספת איבר במקום נתון לרשימה מקושרת)
9. $O(1)$ – מספר פעולות קבועות שלוקחות $O(1)$
10. $O(\log n)$ – כפי שראינו בהרצאה (הוספה לעץ AVL) וכפי שהוסבר במבנה הנתונים.
11. $O(\log m)$ – כפי שראינו בהרצאה (הוספה לעץ AVL) וכפי שהוסבר במבנה הנתונים.
12. $O(1)$ – מספר פעולות קבועות שלוקחות $O(1)$

כל הפעולות נעשות זו אחר זו בנפרד לכן

$$O(\log n) + O(1) + O(\log n) + O(\log m) + O(\log n) + O(1) + O(1) + O(1) + O(\log n) + O(m) + O(1) = O(\log n + \log m)$$

$StatusType\ NumberOfStreams(\ void\ *\ DS, int\ artistID, int\ songID, int\ *\ streams)$

<u>פעולות</u>

1. החזרת $invalid\_input$ אם $DS == NULL\ ||\ songID < 0 ||\ songID >= numOfSongs || artistID <= 0$
2. החזרת $failure$ אם האמן לא קיים (חיפוש $artistID$ בתוך $musicDict$)
3. מתוך $artistsSongs$ המתאים נשתמש במצביע במיקום $songID$ כדי לעבור לאיבר ברשימת $numOfPlaybacks$
4. נחזיר את הערך באיבר המוצבע ברשימה $numOfPlaybacks$

<u>סיבוכיות זמן</u>

1. $O(1)$ – מספר פעולות קבועות שלוקחות $O(1)$
2. $O(\log n)$ – כפי שראינו בהרצאה (חיפוש בעץ AVL)
3. $O(1)$ – מספר פעולות קבועות שלוקחות $O(1)$
4. $O(1)$ – מספר פעולות קבועות שלוקחות $O(1)$

כל הפעולות נעשות זו אחר זו בנפרד לכן

$$O(1) + O(\log n) + O(1) + O(1) = O(\log n)$$

$StatusType\ GetRecommendedSongs(\ void\ *\ DS, int\ numOfSongs, int\ *\ artists, int\ *\ songs)$

<u>פעולות</u>

1. החזרת $invalid\_input$ אם $numOfSongs <= 0\ ||\ DS == NULL$.
2. החזרת $failure$ אם $numOfSongs \leq Total\_num\_of\_songs$
3. נאתחל $counter$ להיות 0
4. מאיבר end ברשימת $numOfPlaybacks$ נעבור אחורה עד לאיבר 0 כולל
    a. נעבור על $PlaybacksArtists$ בסדר עולה בעזרת מצביעי הרשימה

i. נעבור על $PlaybacksSongs$ בסדר עולה בעזרת מצביעי הרשימה
1. נכניס את $artistID$ לתוך $artists$ לפי ערכו של ה $counter$
2. נכניס את $songID$ לתוך $songs$ לפי ערכו של ה $counter$
3. נגדיל את $counter$ באחד
4. אם $counter == numOfSongs$
   a. נעצור את הפונקציה ונחזיר את $artists$ ואת $songs$
5. נעבור למיקום הבא ב $artists$ וב $songs$

<u>סיבוכיות זמן</u>

מכיוון שכל המעברים נעשים בעזרת מצביעי הרשימה (כלומר בצורה של רשימה מקושרת) סיבוכיות הזמן של המעבר על האיברים תלויה רק בכמות האיברים עליה נרצה לעבר. נעבור על כמות איברים m (גודל $numOfSongs$) מספר קבוע של פעמים בעזרת מספר קבוע של פעולות לכן

$$O(m)$$

$void\ Quit(\ void\ **\ DS)$

<u>פעולות</u>

1. מעבר על $musicDict$ בסדר עולה
   a. מחיקת $artistsSongs$
   b. מחיקת איבר האמן המתאים בתוך $musicDict$
2. מעבר על $numOfPlaybacks$ בסדר עולה מ 0 עד end
   a. מעבר על $PlaybacksArtists$ בסדר עולה
      i. מעבר על $PlaybacksSongs$ בסדר עולה
         1. מחיקת איברי $PlaybacksSongs$
      ii. מחיקת איברי $PlaybacksArtists$
   b. מחיקת איברי $numOfPlaybacks$

<u>סיבוכיות זמן</u>

1. $O(m+n)$ - מעבר על n אמנים ועל m שירים וביצוע מספר קבוע של פעולות
2. $O(m+n)$ - ערך מקסימאלי של $n+m$ איברים עליהם נבצע מספר קבוע של פעולות

כל פעולה נעשית זו אחר זו ללא קשר אחת לשנייה לכן

$$O(m+n)+O(m+n)=O(m+n)$$