מבני נתונים תרגיל רטוב 2
יובל גושן 205810179
תובל גלוון 312419971

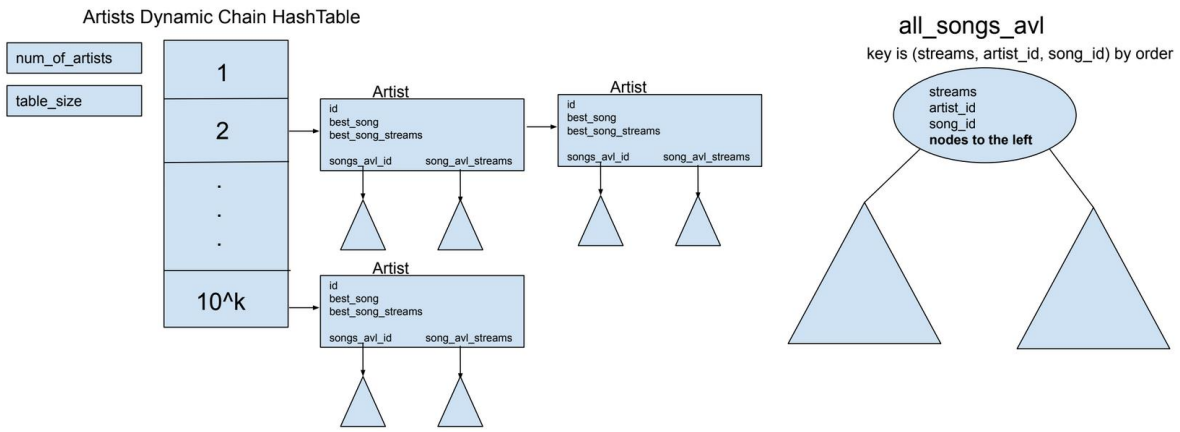
מבנה הנתונים

מבנה הנתונים מתחלק ל2 חלקים עיקריים :

1. טבלת ערבול דינמית – dynamic chained hash table שמכילה את כל האומנים לפי id.
הטבלה מאותחלת בגודל הקבוע 10, וכל פעם שמספר האומנים עולה על גודל הטבלה, נוצרת טבלה חדשה גדולה פי 10, ואליה מועתקים כל האומנים. פעולה זו כולל אתחול הטבלה החדשה לוקחת $O(10n) = O(n)$ זמן, והיא מתבצעת אחרי שהוכנסו לטבלה לפחות n אומנים בזמן ממוצע $O(1)$, לכן סיבוכיות הזמן הממוצע המשוערכת להכנסה לטבלה זו היא $O(1)$, ופקטור העומס בטבלה הוא גם $O(1)$ (ראינו פרטים אלו בהרצאות ובתרגולים) .
כדי לשמור על סיבוכיות מקום כנדרש גם לאחר שהוצאו הרבה אומנים מהטבלה, נדאג להקטין את הטבלה פי 10 בכל פעם שהוצאו אומנים ופקטור העומס קטן מ$\frac{1}{20}$, כלומר נקטין את הטבלה פי 10 והטבלה החדשה תהיה חצי מלאה גם פעולה זו לוקחת $O(1)$ בממוצע משוערך (ראינו זאת בהרצאות ובתרגולים).
הכנסה לטבלה תתבצע לפי פונקציית הערבול : $id\ mod(table\_size)$. המבנה עומד בסיבוכיות הנדרשת תחת הנחת הפיזור האחיד.
הטבלה תכיל מבנה נתונים מטיפוס אומן שיכיל את השדות הבאים :
   א. *artist_id* – מספר האומן – שהוא גם המפתח.
   ב. *best_song* – id של השיר הכי מושמע של אותו אומן
   ג. *best_song_streams* – מספר ההשמעות של השיר הכי מושמע של אותו אומן
   ד. *song_avl_id* – מצביע לעץ *AVL* של השירים של אותו אומן מספר השיר הוא המפתח
   ה. *song_avl_streams* מצביע לעץ *AVL* של השירים של אותו אומן כאשר מספר ההשמעות של שיר הוא המפתח (אם לשני שירים אותו מספר השמעות – אז מספר השיר קובע)

כמובן שבנוסף נשמור שדות *num_of_artists,table_size* למספר האומנים בטבלה וגודל הטבלה.

2. עץ מאוזן של כל השירים במערכת – *all_song_avl* . נשתמש בעץ שמימשנו בתרגיל הקודם וננצל את העובדה שהמפתח הוא מטיפוס גנרי, נגדיר את המפתח כטיפוס בעל 3 שדות – מספר השמעות, מספר אומן ומספר שיר וכך נוכל להכניס לעץ את כל השירים במערכת באופן ממוין כפי שנדרש בתרגיל. יחס הסדר בין מפתחות בעץ מוגדר לפי מספר השמעות, אם לכמה שירים אותו מספר השמעות אז יחס הסדר מוגדר לפי מספר האומן, ובאופן דומה לפי מספר שיר.
כמו כן נוסיף לכל איבר בעץ שדה השומר כמה צמתים משמאלו, ראינו בהרצאה שבאמצעות שמירת מידע זה (אין זה פוגע בסיבוכיות הפעולות על העץ) ניתן להגיע לאיבר המדורג *i* בסדר כרונולוגי ב$O(logn)$.



מימוש הפעולות

נסביר בקצרה את המימוש של כל פעולה ומדוע היא עומדת בסיבוכיות

$init(\ )$

פעולה	סיבוכיות **במקרה הגרוע**
1. נאתחל טבלת ערבול, שהיא למעשה מערך של מצביעים לאומנים, המערך בהתחלה ריק ובגודל הקבוע 10	$O(1)$
2. נאתחל עץ AVL ריק עבור כל השירים	$O(1)$

סה"כ קיבלנו סיבוכיות $O(1)$

$AddArtist(ArtistId)$

פעולה	סיבוכיות **במקרה הממוצע**
1. נחשב לאן להכניס את האומן בטבלת הערבול ע"י: $Location = ArtistId\ mod(table\_size)$ ונכניס את האומן לרשימה המקושרת באותו מיקום	$O(1)$
2. נעדכן את מספר האומנים בטבלה ונבדוק את פקטור העומס – אם מספר האומנים קטן מגודל הטבלה סיימנו, אחרת נמשיך	$O(1)$
3. אם מספר האומנים גדול מגודל הטבלה, כלומר פקטור העומס גדול מ1, נקצה מערך חדש גדול פי 10, נחשב את פונקציית הערבול עבור כל האומנים ונכניס אותם למערך החדש	$O(10n) = O(n)$

פעולה מספר 3 תתבצע לפחות אחרי n הכנסות כל פעם, לכן על סדרה שלn הכנסות יתבצעו סדר גודל של n פעולות בממוצע ולכן הסיבוכיות הממוצעת באופן משוערך היא O(1).

$RemoveArtist(ArtistId)$

פעולה	סיבוכיות **במקרה הממוצע**
1. נמצא את התא במערך בו נמצא האומן באמצעות פונקציית הערבול שהוגדרה בשאלה הקודמת, ונמצא אותו בתוך הרשימה המקושרת שבאותו תא במערך	$O(1)$
2. נסיר את האומן מהרשימה (אם הוא קיים ואין לו שירים)	$O(1)$
3. נבדוק את פקטור העומס, אם הוא קטן מ0.05, כלומר אם מספר האומנים קטן פי 20 לפחות מגודל הטבלה. אם לא נסיים, אם כן נמשיך לפעולה הבאה	$O(1)$
4. אם פקטור העומס קטן, נקצה מערך חדש קטן פי 10 מהנוכחי ונעתיק אליו את כל האומנים מהמערך הקודם, נשחרר את המערך הקודם מהזכרון	$O\left(\frac{n}{10}\right) = O(n)$

פעולה מספר 4 תתבצע לפחות אחרי $\frac{n}{2}$ הוצאות כל פעם, לכן על סדרה שלn הוצאות יתבצעו סדר גודל של n פעולות בממוצע ולכן הסיבוכיות הממוצעת באופן משוערך היא O(1).

$AddSong(ArtistId, SongId)$

פעולה	סיבוכיות **במקרה הממוצע**
1. נמצא את האומן ברשימת הערבול של האומנים	$O(1)$
2. נכניס לעץ שני עצי השירים שלו את השיר עם אפס השמעות (המפתח הוא מספר השיר)	$O(logm)$ כאשר m הוא מספר השירים אצל אותו אומן, שהוא קטן או שווה למספר השירים הכולל במערכת ולכן גם: $O(\log n)$

3. נכניס איבר עם הערכים : 0 השמעות, מספר אומן, מספר שיר לעץ AVL של כל השירים במערכת	$O(logn)$ למעשה כאן זו הסיבוכיות גם במקרה הגרוע

קיבלנו מספר סופי של פעולות בסיבוכיות ממוצעת $O(logn)$

$\boldsymbol{RemoveSong(ArtistId, SongId)}$

פעולה	סיבוכיות **במקרה הממוצע**
1. נוציא את השיר מהעץ של כל השירים	$O(logn)$ למעשה כאן זו הסיבוכיות גם במקרה הגרוע
2. נמצא את האומן ברשימת הערבול של האומנים	$O(1)$
3. נוציא מעץ השירים (של אותו אומן) שממוין לפי id את השיר, וכך נדע גם כמה השמעות היו לאותו שיר	$O(logm)$ כאשר m הוא מספר השירים אצל אותו אומן, שהוא קטן או שווה למספר השירים הכולל במערכת ולכן גם : $O(\log n)$
4. נוציא אותו גם מהעץ שממוין לפי מספר השמעות (שמרנו את מספר ההשמעות של אותו שיר בסעיף הקודם	_____"_____
5. אם זה היה השיר הכי מושמע במערכת, נעדכן את השיר המקסימלי מהעץ שירים שממוין לפי מספר השמעות (הימני ביותר)	$O(logm)$ כאשר m הוא מספר השירים אצל אותו אומן, שהוא קטן או שווה למספר השירים הכולל במערכת ולכן גם : $O(\log n)$ (הגעה לאיבר המקסימלי בעץ AVL)

קיבלנו מספר סופי של פעולות בסיבוכיות ממוצעת $O(logn)$

$\boldsymbol{AddToSongCount(ArtistId, SongId, count)}$

פעולה	סיבוכיות **במקרה הממוצע**
1. נוציא את השיר מהעץ של כל השירים	$O(logn)$ למעשה כאן זו הסיבוכיות גם במקרה הגרוע
2. נכניס מחדש לעץ כל השירים עם מספר השמעות גבוה בcount	$O(logn)$ למעשה כאן זו הסיבוכיות גם במקרה הגרוע
3. נמצא את האומן ברשימת הערבול של האומנים	$O(1)$
4. ניגש לשיר בעץ השירים של אותו אומן הממוין לפי מספר שיר	$O(logm)$ כאשר m הוא מספר השירים אצל אותו אומן, שהוא קטן או שווה למספר השירים הכולל במערכת ולכן גם : $O(\log n)$
5. ,נשמור את מספר ההשמעות הישן עבור השיר במשתנה עזר ונעדכן את מספר ההשמעות של אותו שיר באותו עץ	$O(1)$
6. נוציא אותו מהעץ שממוין לפי מספר השמעות (שמרנו את מספר ההשמעות של אותו שיר בסעיף הקודם	$O(logm)$ כאשר m הוא מספר השירים אצל אותו אומן, שהוא קטן או שווה למספר השירים הכולל במערכת ולכן גם : $O(\log n)$
7. נכניס מחדש עם מספר השמעות גבוה בcount	_____"_____
8. אם לשיר זה יותר השמעות מהשיר הכי מושמע עבור אותו אומן, נעדכן את השיר הזה להיות הכי מושמע עבור אותו אומן	$O(1)$

קיבלנו מספר סופי של פעולות בסיבוכיות $O(logn)$

$GetArtistBestSong(ArtistId,* SongId)$

פעולה	סיבוכיות **במקרה הממוצע**
1. ניגש לאומן ברשימת הערבול	$O(1)$
2. נחזיר את השדה השומר את מספר השיר הכי מושמע	$O(1)$

$GetRecommendedSongByPlace(rank,* ArtistId,* SongId)$

עלינו למצוא בעץ שמכיל את כל השירים את האיבר המדורג במקום הr. כדי לעשות זאת בסיבוכיות הנדרשת ננצל את העובדה שאנו שומרים בכל צומת את מספר הצמתים שמתחתיה בצד שמאל.

פעולה	סיבוכיות **במקרה הגרוע**
1. נתחיל מהשורש	
2. אם יש לו בדיוק r בנים שמאליים מצאנו את האיבר המדורג r, נחזיר אותו	סה"כ נרד לכל היותר את גובה העץ לכן הסיבוכיות היא $O(logn)$
3. אם יש לו יותר מr בנים שמאליים, נעבור לבן השמאלי שלו ונחזור לסעיף 2	
4. אם יש לו פחות מr בנים שמאליים, נעבור לבן הימני שלו ונחסיר מr את מספר הבנים השמאליים של הצומת	

$Quit(\ )$

פעולה	סיבוכיות **במקרה הגרוע**
1. מוחקים את כל האומנים מהרשימת ערבול לפי הסדר, ומכל אומן את השירים שלו. (מחיקת 2 עצי AVL בגודל m)	$O(m+n)$ לכל היותר n אומנים וm שירים
2. מחיקת העץ all_songs_avl, שמכיל m שירים	$O(m)$ מחיקת עץ AVL עם m איברים
3. שחרור מצביע למבנה הנתונים כולו	$O(1)$

סיבוכיות המקום

בכל רגע נתון אנו מחזיקים את הנתונים הבאים:

1. עץ AVL עם צומת עבור כל שיר במערכת, כל צומת מכיל מספר קבוע של שדות ולכן סה"כ $O(n)$ מקום
2. מערך (טבלת ערבול) של מצביעים לאומן בגודל החסום ע"י מספר האומנים מלמעלה ומספר האומנים חלקי 20 מלמטה (כשמספר האומנים קטן מגודל הטבלה חלקי 20 אנו מקטינים אותה). סה"כ $O(m)$ מקום
3. מבנה עבור כל אומן המכיל מספר קבוע של שדות, חוץ מעצי השירים שלו אליהם נתייחס בסעיף הבא סה"כ $O(m)$ מקום
4. כל אומן מכיל 2 עצי AVL שבהם מופיע כל שיר של אותו אומן פעם אחת, כמובן שכל שיר משוייך לאומן אחד לכן כל שיר מופיע פעמיים נוספות בעץ AVL ותופס עוד גודל קבוע. סה"כ כל השירים יחד תופסים עוד $O(m)$ מקום

קיבלנו סיבוכיות מקום כוללת של $O(m+n)$