תאור כללי של המבנה

*(כל אחד מהעצים הבאים נגיש מהשורש* ***או*** *מהעלה הכי קטן...)*

נחזיק עץ AVL שהמיון בו מתבצע על פי מזהה הקורס. כל צומת בעץ מייצג קורס שונה.

כל קורס מחזיק מערך של שיעורים. כל תא במערך מייצג שיעור בעל זמן צפייה מסויים ומצביע לזמן הזה.

בנוסף נחזיק רשימה מקושרת שמייצגת ציר זמן עולה. כל חוליה ברשימה מייצגת זמן שונה.

כל נק' זמן מחזיקה עץ AVL של קורסים עם הרצאות בעלות אותו זמן צפיה, כך שההרצאות גם מוחזקות בעץ AVL.

כל הרצאה "מצביעה בחזרה" אל זמן הצפיה שלה.

נתאר בקצרה את מימוש מבנה הנתונים CoursesManager

המבנה מתבסס על רשימה מקושרת דו-כיוונית, ועל עץ AVL אשר הגישה אליו יכולה להתבצע בזמן (1)O דרך השורש או דרך הצומת עם המפתח המינימלי (לאחר הוספה/הסרה מהעץ נעדכן את המצביע אל הצומת השמאלי ביותר ב-O(logn)).

* **timeAxis** – רשימה מקושרת דו-כיוונית המייצגת את ציר הזמן (ממויינת בסדר עולה\יורד כתלות בכיוון ההסתכלות). הפרש הזמנים בין כל 2 חוליות עוקבות ברשימה גדול או שווה לאחד (ובפרט הפרשי הזמנים אינם שווים). תוכן כל חוליה ברשימה המקושרת:
  + *Time* – חוליה ברשימה מקושרת שמחזיקה:
    - עץ AVL של כל הקורסים אשר להם הרצאה עם זמן הצפייה של החוליה. עבור על קורס נחזיק עץ AVL בו כל הרצאה מצביעה חזרה ל-Time במקום הרלוונטי.
    - ומספר המייצג את זמן הצפיה.
* **courses** – עץ AVL המייצג את כל הקורסים במערכת. במימוש בחרנו להשתמש בגרף לא מכוון עם מצביעים דו-כיווניים. תוכן כל צומת בעץ:
  + *Course* – צומת בעץ שמחזיק
    - מערך של כל ההרצאות שקשורות לקורס של החוליה. כל הרצאה מצביע לזמן הצפיה שלה.
    - ומספר המייצג את סך ההרצאות בקורס.

**פעולות מבנה הנתונים**

**Init()**

* נאתחל עץ AVL ריק courses בזמן O(1).
* נאתחל רשימה מקושרת דו-כיוונית ריקה timeAxis בזמן O(1) במקרה הגרוע.
* נוסיף ל-timeAxis את החוליה עם הזמן t=0, זאת אומרת, נוסיף חוליה לרשימה מקורשת ונאתחל עץ AVL של החוליה ריק בזמן O(1) במקרה הגרוע.

זמן – ביצענו 3 פעולות בזמן O(1) ולכן זמן הריצה הכולל הוא O(1) במקרה הגרוע.

מקום – לא מתבצע שימוש בזיכרון נוסף. (1)O

**AddCourse(DS, CourseID, numOfClasses)**

* נצור Course: נקצה מערך בגודל numOfClasses ונאתחל כל תא להצביע אל החוליה הראשונה ב-timeAxis כלומר ב-time=0. בסיבוכית זמן של O(m) כאשר numOfClasses=m.
* *נוסיף את הקורס ל-DS.courses (עץ ה-AVL שלנו) עם המפתח CourseID בסיבוכיות זמן של הוספה לעץ AVL – O(logn)* במקרה הגרוע*, כאשר n הוא מספר הקורסים.*
* *נצור עץ של ההרצאות:*
  + נצור עץ AVL עם צומת יחיד בזמן O(1)- ונרחיב אותו לעץ מלא בעזרת סיור pre-order, כאשר כל ביקור בצומת מוסיף בן ימני ובן שמאלי. נעשה זאת עד לעומק רקורסיה

. כלומר מספר הביקורים שבצענו הוא ,

כל ביקור בזמן O(1), ולכן סיבוכיות הזמן היא

במקרה הגרוע.

* + *נבצע סיור post-order (אפשר גם סיור אחר) בסדר הפוך על מנת להסיר את העלים המיותרים בעץ (או לא לעשות כלום במקרה שלא עלה) בסיבוכיות O(1) לביקור בצומת. גם סיור זה מתבצע בזמן כמספר הביקורים בצמתים --* במקרה הגרוע*.*
  + *נבצע סיור in-order על מנת לבצע השמה של כל המפתחות בין 0 לבין numOfClasses לתוך העץ הכמעט שלם שקיבלנו, כך שבהכנסת כל מפתח, נוסיף מצביע לציר הזמן (הרשימה המקושרת שלנו DS.timeAxis) במקום t=0. כל ביקור בצומת לקח O(1) זמן, ולכן שוב הפעולה היא בסיבוכיות כמספר הביקורים בצמתים --* במקרה הגרוע*.*

*סה"כ זמן יצרת עץ ההרצאות הוא כזמן 3 סיורים בעץ עם פעולה שלוקחת (1)O ולכן נקבל O(m)* במקרה הגרוע, *כאשר m = numOfClasses.*

* *ניגש ל"עץ הקורסים" שנמצא ב-time=0. נבצע הוספה של צומת (עץ ההרצאות שיצרנו) עם המפתח CourseID בזמן הוספה לעץ של O(logn).*

זמן – T = O(m)+O(logn)+ O(m)+O(logn) =O(logn+m) במקרה הגרוע.

מקום – אנו משתמשים בזכרון **נוסף** רק בזמן סיורים רקורסיבים בעץ ובזכרון המשמש את מבנה. סיבוכיות הזכרון **הנוסף** היא כעומק הרקורסיה בעת יצירת עץ ההרצאות -- . לכן המקום הנוסף O(logm) ובפרט של O(M+n), כאשר M הוא מספר ההרצאות הכולל במבנה. כמו כן, יצרנו עץ עם m צמתים ובפרט עדיין O(M+n) מכיוון ש-.

**RemoveCourse(DS, courseID)**

* נבצע חיפוש ב-AVL DS.coursesשל הקורס courseID בזמן O(logn) במקרה הגרוע. כך ש-n הוא מספר הקורסים. נזכור ש- כאשר M הוא מספר ההרצאות הכולל, ולכן בפרט זמן הריצה הוא O(logM) במקרה הגרוע. מטרת החיפוש היא לעבור על כל ההרצאות של הקורס.
* נעבור על מערך ההרצאות של הקורס הרלוונטי באופן סידרתי, כך שמכל תא נבקר בחוליה המתאימה ב-timeAxis :
* ניגש דרך המצביע שבתא ה-i אל ציר הזמן DS.timeAxis במקום המתאים (זמן הצפיה של ההרצאה) ב-(1)O. ונגיע לעץ ההקורסים של אותו זמן.
* בעץ ה-AVL אליו הגענו, ניגש לקורס עם המזהה CourseID בזמן (logn)O. הגענו לעץ AVL בו ישנן k­i ההרצאות. נשחרר את כל הצמתים בו בעזרת סיור post-order שמבצע ביקורים בעץ. סה"כ זמן לפעולה הזאת הוא O(logn+ki).
* נסיר את הקורס (עם העץ הריק) בו הרגע טיפלנו מהעץ בחוליה שהגענו אליה מהתא ה-i בזמן O(logn) ובפרט.

נשים לב שמתקיים ולכן עבור כל מערך ההרצאות של הקורס הפעולה תקח זמן של , ולכן בסה"כ O(mlogn+m)=O(mlogn) ובפרט O(mlogM).

* נבצע הסרה (של הצומת שהוא העץ הריק) מעץ ה-AVL של הקורסים DS.courses בסיבוכיות הסרה מעץ AVL – O(logn) במקרה הגרוע. כפי שהראנו מקודם סיבוכיות זו היא גם בפרט היא O(logM).

זמן – T=O(logM)+O(mlogM)+O(logM)=O(mlogM) במקרה הגרוע.

מקום – אנו משתמשים בזכרון **נוסף** רק בזמן סיורים רקורסיבים בעץ. סיבוכיות הזכרון הנוסף היא כעומק הרקורסיה -- . לכן המקום **הנוסף** O(logM) ובפרט של O(M+n).

**WatchClass(DS, courseID, ClassID, time)**

* נבצע חיפוש ב-AVL DS.coursesשל הקורס coursed בזמן O(logn) במקרה הגרוע. כך ש-n הוא מספר הקורסים. נזכור ש- כאשר M הוא מספר ההרצאות הכולל, ולכן בפרט זמן הריצה הוא O(logM) במקרה הגרוע. מטרת החיפוש היא למצא את classID של courseID.
* הגענו לצומת של הקורס המתאים. ניגש לתא ה-classID בזמן O(1). המטרה היא לגשת למבציע שיש באותו תא במערך.
* דרך המצביע נגיע אל ציר הזמן DS.timeAxis (הרשימה המקושרת) במקום שמייצג את זמן הצפיה המתאים, נסמנו , ב-O(1) ושם ניגש לעץ הקורסים של אותו הזמן כדי לגשת לקורס courseID בסיבוכיות O(logn), ובפרט O(logM).
* הגענו לעץ ההרצאות המתאים. נסיר את ההרצאה עם classID. זמן ההסרה הוא הסרה מ-AVL ב- O(logM) במקרה הגרוע.
* נתקדם לכל היותר t=time צעדים ברשימה המקושרת המייצגת את ציר הזמן כך שנגיע אל זמן t0+t. נבצע זאת באופן הבא:

בצעד ה-i (חוליה אחרי חוליה החל מ-t0):

* + אם ti<t+t0 התקדם ל-ti+1 הבא בזמן של O(1).
  + אם ti=t+t­0 אז הוסף לעץ המשוייך ל-ti את הקורס (אם לא קיים כבר) בזמן של O(logn) בפרט O(logM) במקרה הגרוע. כעת נוסיף לקורס (עץ AVL של ההרצאות) את ההרצאה המשוייכת לזמן ti בסיבוכיות O(logM).
  + אם ti>t+t0 אז הוסף חוליה לרשימה של ציר הזמן לפני החוליה ti. בחוליה זו נשים את הזמן t+t0 ונוסיף עץ עם צומת יחיד courseID. ההוספה לאמצע של רשימה דו-כיוונית עם מצביע לחוליה ספציפית מתבצעת בזמן (1)O, ויצירת עץ עם צומת בודד לוקחת גם (1)O. כעת נוסיף לקורס צומת יחיד (את classID) בזמן O(log1) כלומר O(1). סה"כ O(1).

במקרה הגרוע לולאה זו רצה t צעדים ומבצעת מקסימום פעם אחת שתי הוספות ל-AVL ב-O(logM). לכן במקרה הגרוע פעולה זו לוקחת O(logM)+O(t)=O(logM+t).

זמן – T=O(logM)+O(logM)+O(logM+t) = O(logM+t)

מקום – לא מתבצע שימוש בזכרון נוסף. (1)O

**TimeViewed(DS, courseID, ClassID, timeViewed)**

* כמו בפונקציות הקודמות נמצא את הקורס בעץ ה-AVL DS.courses בזמן O(logn) כאשר n זה מספר הקורסים. הגענו לעץ AVL של הרצאות.
* ניגש אל התא המתאים במערך ההרצאות של הקורס בזמן O(1), ודרך המצביע בתא זה ניגש אל הזמן המתאים ברשימה המקושרת של ציר הזמן גם בזמן (1)O. סה"כ O(1).
* נחזיר ל- timeViewed את זמן הצפיה שמצאנו (אותו אנחנו מחזיקים בכל חוליה ברשימה). פעולה שלוקחת (1)O.

זמן – T=O(logn)+O(1)+O(1)=O(logn)

מקום – לא מתבצע שימוש בזכרון נוסף. (1)O

**GetMostViewedClasses(DS, numOfClasses, courses, classes)**

*נגדיר סיור climbingInOrder באופן הבא:*

*בדומה לסיור in-order רגיל נעבר על צמתי העץ מהמפתח הקטן ביותר למפתח הגדול ביותר. ההבדל הוא: שאנו ניגשים דרך מצביע אל העלה השמאלי ביותר במקום אל השורש בזמן O(1). בכל צומת החל מלמטה ועד השורש נבצע את הפעולה על הצומת עצמו, ואז נקרא לסיור in-order רגיל על תת-העץ הימני שלו עם אותה פעולה. לאחר מכן נטפס למעלה אל ההורה.*

*סיבוכיות climbingInOrder:*

*משום שביצענו סיורי in-order רגילים על כל תת-עץ, במצטבר הזמן של הסיור החדש הוא עדיין*

*O(n).*

אבחנה: לביקור הראשון בצומת נגיע לאחר O(1) במקום לאחר זמן של O(logn). כאשר n זה מספר הצמתים בעץ בו מסיירים בסיור in-order מטפס.

* נגדיר k=0 בשביל לתת אינדקס למערכי הפלט.
* נגדיר stepsLeft = numOfClasses.
* ניגש לסוף הציר הזמן (זנב הרשימה המקושרת הד-כיוונית timeAxis).

עד כאן ביצענו פעולות בזמן O(1).

* כל עוד stepsLeft>0, נבצע ***בכל חוליה ti*** ברשימה המקושרת מהסוף להתחלה:
* אם stepsLeft>0 נפעיל את פונקצית העזר climbingInOrder על העץ של הקורסים המשוייך ל-ti כך ש***בכל ביקור בצומת*** (הצומת הוא עץ ההרצאות):
  + - (אם stepsLeft>0) נפעיל climbingInOrder על עץ ההרצאות כך שבכל ביקור בצומת (הרצאה) נוסיף את ההרצאה ל-classes[k] ואת הקורס ל-courses[k].
    - לאחר כל הוספת צמד של הרצאה וקורס למערכי הפלט נעדכן בהתאם את מספר הצעדים שנותרו... stepsLeft 🡨(stepsLeft-1) ונקדם את k🡨k+1
  + אם stepsLeft=0 במהלך ריצת ה"לולאה החיצונית"  **או** סיורי ה" inOrder הפנימיים", נפסיק את המעבר על ציר הזמן. כלומר הוספנו numOfClasses פלטים כמבוקש.

נשים לב: אם בתחילת סיור climbing in-order מספר הצעדים p שנותרו לבצע קטן מ-logm )כאשר m זה מספר הצמתים בעץ בו מסיירים( זמן הריצה הינו o(p) בניגוד לסיור רגיל בו זמן הריצה היה נשאר באופן לא יעיל O(logm+p)=O(logm).

כל גישה לתחילת סיור(על הקורסים או על ההרצאות) היא בזמן O(1). כל ביקור בצומת הוא בזמן O(1). לכן סה"כ כמות הפעולות שעשתה הלולאה היא O של מספר הביקורים בכל צמתי ההרצאות. ביצענו לכל היותר numOfClasses ביקורים. לכן, סיבוכיות פעולה זו היא O(m) כאשר m הוא numOfClasses. (בביקור אחד בצומת של הרצאה אנו כוללים גם זמן הגישה לקורס בO(1) וגם את הגישה לרשימה ב-(1)O ).

* במידה וסיימנו לעבור על כל הרשימה, אבל stepsLeft>0 נחזיר FAILURE. כלומר ביקשו יותר הרצאות מכמות ההרצאות שיש במבנה. אחרת נחזיר SUCCESS.

זמן – T=O(m)+O(1)=O(m)

מקום – אנו משתמשים בזכרון **נוסף** רק בזמן סיורים רקורסיבים בעץ. סיבוכיות הזכרון הנוסף היא כעומק הרקורסיה -- . לכן המקום **הנוסף** O(logM) ובפרט של O(M+n).

**Quit(DS)**

* ניגש לרישמה של ציר הזמן DS.timeAxis, ותא אחר תא ונבצע:
  + נבצע סיור post-order על עץ ה-AVL של חוליה כך שבכל ביקור בצומת נהרוס את עץ ההרצאות גם בעזרת post-order. כלומר ביקרנו ב-m+n צמתים סה"כ. כל ביקור בזמן O(1). סה"כ O(n+m).
  + נסיר את החוליה ברשימה מקושרת ב-O(1).

סה"כ זמן פעולה זו היא O(n+m).

* נבצע סיור post-order על n הקורסים ב-DS.courses ובכל צומת i נהרוס את המערך בגודל ki של ההרצאות. מתקיים ולכן סה"כ ביצענו הכל בזמן O(n+m).

זמן – T=O(n+m)+O(n+m)=O(n+m)

מקום – אנו משתמשים בזכרון **נוסף** רק בזמן סיורים רקורסיבים בעץ. סיבוכיות הזכרון הנוסף היא כעומק הרקורסיה -- . לכן המקום **הנוסף** O(logM) ובפרט של O(M+n).