**גיליון רטוב מספר 1 – חלק יבש  
  
מגישים:   
נועם יעקבי - 316600782   
עידו קולמן - 316465202   
  
  
תיאור מבני הנתונים:  
  
CourseManager**   
מבנה הנתונים הראשי שלנו נקרא CourseManager ומטרתו אחסון מידע על קורסים, השיעורים ששייכים לכל קורס וזמני הצפייה של כל שיעור.  
מבנה זה מכיל שני עצי AVL, רשימה מקושרת דו-כיוונית, מצביע וקאונטר.  
מבנה הנתונים שלנו שומר את הקורסים בעצי AVL, עץ חיפוש בינארי שמתקן עצמו על מנת לשמור על גובה נמוך, כאשר כל קורס שומר במערך מידע על כל השיעורים השייכים אליו.  
בעץ אחד שומרים קורסים שיש להם לפחות שיעור אחד עם זמן צפייה 0 ובעץ השני כאלה שכבר אין להם.  
כמו כן מבנה הנתונים שומר ברשימה המקושרת איברים כמספר זמני הצפייה השונים שקיימים במערכת, כאשר כל איבר ב-times שומר עץ AVLהמכיל את כל השיעורים בעלי זמן הצפייה שמתאים למזהה של times ששומר עץ זה, וכן רשימה מקושרת ממוינת של שיעורים המכילה את אותם שיעורים כמו בעץ.  
כל האיברים בעצים וברשימה המקושרת הם מאותו טיפוס על מנת לאפשר מעבר קל בין המבנים, כאשר בכל מבנה נשתמש אך ורק בשדות הרלוונטיים לאיבר המתאים.  
  
**עץ AVLבשם** **courses:**   
מכיל את הקורסים שאין להם אף שיעור בעל זמן צפייה 0,   
ממוין לפי המזהה של כל צומת שהוא ה-ID של הקורס.  
כל צומת מכילה:

* ***id*** id - של הקורס.
* ***parent***– מצביע לאיבר האב בעץ.
* ***left*** – מצביע לבן השמאלי בעץ, בהכרח בעל id קטן יותר מאשר איבר זה.
* ***right*** – מצביע לבן הימני בעץ, בהכרח בעל id גדול יותר מאשר איבר זה.
* ***count*** – כמות השיעורים השייכים לקורס.
* ***classes***– מצביע לתחילת מערך באורך *count* של השיעורים השייכים לקורס, כאשר כל איבר במערך הוא מצביע לשיעור שה- idשלו זהה למקום שלו במערך, וכן כל איבר הוא גם צומת בעץ AVL המתאים לזמן הצפייה שלו, וגם איבר ברשימה מקושרת עבור אותו זמן צפייה.
* ***first\_in\_time0*** – nullptr עבור העץ courses, אין שיעורים בעלי זמן צפייה 0.
* ***last\_in\_time0*** – nullptr עבור העץ courses, אין שיעורים בעלי זמן צפייה 0.

**עץ AVLבשם** **courses\_in\_time0:**   
מכיל את הקורסים שיש להם לפחות שיעור אחד בעל זמן צפייה 0,   
ממוין לפי המזהה של כל צומת שהוא ה-ID של הקורס.  
זהה לעץ AVL בשם courses למעט שני השדות:

* ***first\_in\_time0*** – מצביע לשיעור בעל ה-id הנמוך ביותר שזמן הצפייה שלו הוא עדיין 0,   
  נשמר על מנת לשפר יעילות בפעולת הכנסת קורס למבנה הנתונים.
* ***last\_in\_time0*** – מצביע לשיעור בעל ה-id הגבוה ביותר שזמן הצפייה שלו הוא עדיין 0,   
  נשמר על מנת לשפר יעילות בפעולת הכנסת קורס למבנה הנתונים.

**רשימה מקושרת דו-כיוונית בשם :times**  
רשימה מקושרת בה כל איבר מתאים לזמן צפייה שהוכנס,   
הרשימה מסודרת לפי זמן צפייה הולך וגדל כאשר האיבר הראשון יהיה זמן צפייה שהוא 0.המזהה של כל איבר הוא זמן הצפייה המתאים לאיבר זה,   
כל איבר מכיל גם עץ AVLשל כל השיעורים בעלי זמן הצפייה המתאים למזהה של איבר זה, וגם רשימה מקושרת של אותם שיעורים הממוינים לפי ה- idשלהם.  
כל איבר מכיל:

* ***id*** – זמן הצפייה המתאים לאיבר זה.
* ***next*** – האיבר הבא ברשימה, כאשר בהכרח ה-id שלו גבוה יותר,   
  כלומר זמן הצפייה המתאים לו גבוה יותר.
* ***prev*** – האיבר הקודם ברשימה, כאשר בהכרח ה-id שלו נמוך יותר,   
  כלומר זמן הצפייה המתאים לו נמוך יותר.
* ***root*** – מצביע לשורש של עץ AVL,   
  עץ שהצמתים שלו הם שיעורים מכל הקורסים, בעלי זמן הצפייה מתאים ל-id של האיבר.   
  העץ ממוין לפי המזהה של האיבר, כאשר המזהה מורכב מה-id של הקורס אליו הוא שייך ו-id של השיעור.
* ***first*** – מצביע לאיבר הראשון ברשימה המקושרת של השיעורים בעלי זמן הצפייה המתאים לאיבר זה.
* ***last*** - מצביע לאיבר האחרון ברשימה המקושרת של השיעורים בעלי זמן הצפייה המתאים לאיבר זה.

**מצביע בשם last\_time:**מצביע לאיבר האחרון ברשימה המקושרת times, כלומר לזמן הצפייה הגבוה ביותר,  
על מנת לשפר את יעילות הזמן של הפעולה GetMostViewedClasses.  
  
**קאונטר בשם class\_count:**סופר את כמות השיעורים בכל הקורסים ביחד.

|  |
| --- |
|  |
|  |
|  |
|  |
|  |
|  |

**CourseManager**

**מכיל את.../מצביע ל...  
  
 זהות  
  
  
  
עצי AVLממוין לפי ה-id של הקורס,  
  
עצים בשמות courses ו-courses\_in\_time0 בנויים זהה**

איבר בעץ courses:

**מערך שיעורים בשם classes**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
| count - 1 | **.** | **.** | **.** | **2**  איבר במערך classes: | **1** | **0** |

|  |
| --- |
|  |

**מערך של שיעורים מטיפוס Node\* בשם classes**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
| count - 1 | **.** | **.** | **.** | **2** | **1** | **0**  איבר מטיפוס Node\* השייך גם למערך classes, גם לעץ ה-AVL של איבר ברשימה times, וגם לרשימה של איבר הראשון ברשימה :times |

|  |
| --- |
|  |

=

=

**רשימה מקושרת דו-כיוונית בשם timesלפי זמן צפייה עולה**, **כל איבר ברשימה מכיל עץ AVL שהצמתים שלו הם שיעורים ורשימה מקושרת בעלת אותם שיעורים,  
למעט האיבר של זמן 0 המכיל רק רשימה מקושרת שכל איבר בה הוא שיעור בעל זמן צפייה 0**

class

class

class

איבר ברשימה times:

**הוכחת עמידה בסיבוכיות הזמן עבור הפעולות:**

**void \* Init()**

**סיבוכיות זמן:** מכיוון שמבנה הנתונים שלנו ריק באתחול,

* אתחול עצים ומצביעים להצביע על nullptrעולה
* אתחול האיבר הראשון ברשימה מקושרת של זמנים שמאתחלים אותו להיות זמן 0 עולה
* אתחול קבוע עולה

ולכן סיבוכיות הזמן כוללת תהיה .

**סיבוכיות מקום:** מכיוון שמבנה הנתונים ריק סיבוכיות המקום תהיה .

**StatusType AddCourse (void \* DS, int courseID, int numOfClasses)**

**הסבר המימוש:** אנו יוצרים קורס חדש, מכניסים אותו לעץ courses\_in\_time0, כאשר פעולת ההכנסה לעץ AVL עולה . כמו כן פעולת ההכנסה מוצאת ומחזירה את הצומת בעל המזהה הגדול הבא והצומת בעל מזהה קטן באחד, כאשר מציאה זו מחפשת בגובה העץ ולכן עולה .  
לאחר ההכנסה אנו מאתחלים את הקורס לפי הנתונים שקיבלנו, יוצרים מערך שיעורים, מאתחלים כל אחד מהם כך שהמזהה שלו יתאים למקום שלו במערך, מוסיפים אותו לרשימה המקושרת של זמן אפס כאשר הקודם והבא שלו הם השיעורים בעלי מזהה קטן וגדול באחד בהתאמה, ולכן אתחול זה עולה .  
אנו שומרים את השיעור הראשון והאחרון שהוספנו לזמן 0, מחברים את השיעור הראשון לשיעור האחרון של הקורס שאחד לפניו (שמצאנו בפעולת ההכנסה) וכן מחבר את השיעור האחרון לשיעור הראשון של הקורס הבא אחריו מבחינת מזהה, פעולה שעולה ,   
וכך יצרנו רשימה מקושרת ממוינת בסדר עולה של כל השיעורים בזמן 0.  
  
**סיבוכיות זמן:**

* הכנסת איבר לעץ AVL -
* מציאת קודם והבא בעץ AVL, שני חיפושים בגובה העץ ולכן -
* אתחול מערך השיעורים -
* חיבור השיעורים ברשימה המקושרת, שני חיבורים עבור כל אחד ולכן -

ולכן סיבוכיות הזמן הכוללת של הפעולה תהיה .

**סיבוכיות מקום:**

* הקצאת מערך בגודל מספר השיעורים -
* הכנסה ומציאה רקורסיבית בעץ AVL, כאשר עומק הרקורסיה יהיה במקרה הגרוע ביותר בגובה העץ, ולכן סיבוכיות המקום תהיה במקרה הגרוע -

ולכן סיבוכיות המקום הכוללת של הפעולה תהיה .

**StatusType RemoveCourse( void \* DS, int courseID)**

**הסבר המימוש:** מסירים את הקורס מהעץ אליו הוא שייך, בין אם הוא בעץ עם זמן 0 או לא, יהיו בעץ מקסימום n  
קורסים ולכן סיבוכיות ההסרה תהיה .   
כל זמן מכיל רשימה מקושרת ממוינת ולכן שומר מצביע לשיעור הראשון והאחרון ברשימה, ולכן בהסרה אנו בודקים האם כל שיעור הוא הראשון או האחרון בזמן שלו ומשנים את הרשימה בהתאם, ואז מסירים אותו מהרשימה.  
כמו כן עבור כל שיעור אנו מסירים אותו מעץ השיעורים של זמן הצפייה המתאים לו בו הוא שמור בעלות .  
לאחר מכן מוחקים את כל הנתונים של הקורס בסיבוכיות של .  
  
**סיבוכיות זמן:**

* הוצאת איבר מעץ AVL -
* עדכון הרשימה המקושרת והעץ של הזמן המתאים לכל שיעור -

ולכן מכיוון שנתון סיבוכיות הזמן הכוללת של הפעולה תהיה .

**סיבוכיות מקום:**

* הוצאת קורס מעץ AVL של הקורסים, כאשר עומק הרקורסיה יהיה במקרה הגרוע ביותר בגובה העץ,   
  ולכן סיבוכיות המקום תהיה במקרה הגרוע -
* הוצאה שיעור מעץ AVLשל שיעורים, כאשר עומק הרקורסיה יהיה במקרה הגרוע ביותר בגובה העץ,   
  ולכן סיבוכיות המקום תהיה במקרה הגרוע -

ולכן מכיוון שנתון סיבוכיות המקום הכוללת של הפעולה תהיה .

**StatusType WatchClass( void \* DS, int courseID, int classID, int time)**

**הסבר המימוש:** בודקים תקינות וכן האם הקורס קיים במערכת, כלומר מבצעים חיפוש בעץ הקורסים .  
לאחר הבדיקה, אם הזמן הקודם של השיעור הוא 0, בודקים האם זה היה השיעור האחרון של הקורס בזמן 0, אם כן מסירים את הקורס מעץ courses\_in\_time0 ומוסיפים אותו לעץ courses בסיבוכיות של .  
מוציאים את השיעור מהעץ של הזמן הנוכחי שלו במידה והזמן הנוכחי שונה מ-0, בסיבוכיות של .  
לאחר מכן מתקנים את הרשימה המקושרת של הזמן של השיעור במידה והשיעור היה ראשון או אחרון, ואז מסירים אותו מהרשימה בסיבוכיות של .  
אנו יודעים מה הזמן הנוכחי של השיעור שכן זה שמור לנו, לאחר מוצאים את הזמן החדש ברשימה המקושרת של הזמנים, כאשר מציאה או יצירה שלו תהיה ב-.  
לאחר מציאת איבר הזמן, מכניסים את השיעור לעץ של הזמן, תוך כדי מציאת השיעור הקודם לו והבא לו בעץ, בסיבוכיות כוללת של ,   
ולאחר מכן מספר פעולות תיקון וסידור הרשימה המקושרת בסיבוכיות של .  
  
**סיבוכיות זמן:**

* חיפוש קורס בעץ AVL ואז הוצאה והכנסה לעץ הקורסים השני במידה וצריך -
* הוצאת השיעור מעץ AVL של הזמן הקודם שלו -
* השיעור שומר לנו מצביע לזמן הנוכחי,   
  ולכן מציאת או יצירת איבר הגדול ממנו ב-t ברשימה מקושרת ממוינת -
* הכנסת השיעור לעץ AVL של הזמן החדש ומציאת השיעור הקודם לו והבא לו בעץ -
* פעולות ספורות של תיקון רשימות מקושרות -

ולכן סיבוכיות הזמן הכוללת של הפעולה תהיה .

**סיבוכיות מקום:**

* מציאה, הכנסה והוצאה של קורס בעץ AVL של הקורסים, כאשר עומק הרקורסיה יהיה במקרה הגרוע ביותר בגובה העץ, ולכן סיבוכיות המקום תהיה במקרה הגרוע -
* מציאה, הכנסה והוצאה של שיעור בעץ AVL של שיעורים, כאשר עומק הרקורסיה יהיה במקרה הגרוע ביותר בגובה העץ, ולכן סיבוכיות המקום תהיה במקרה הגרוע -

ולכן סיבוכיות המקום הכוללת של הפעולה תהיה .

**StatusType TimeViewed(void \* DS, int courseID, int classID, int \* timeViewed)**

**הסבר המימוש:** נבדוק האם הקורס קיים באחד מעצי הקורסים,   
אם הוא קיים ניכנס למערך השיעורים של הקורס ונמצא את זמן הצפייה שלו בתא המתאים ל-id של השיעור.  
  
**סיבוכיות זמן:**

* חיפוש קורס בעץ AVL -
* גישה למידע במערך כאשר אנו יודעים מה האינדקס -

ולכן סיבוכיות הזמן הכוללת של הפעולה תהיה .

**סיבוכיות מקום:**

* חיפוש בעץ AVL, כאשר עומק הרקורסיה יהיה במקרה הגרוע ביותר בגובה העץ,   
  ולכן סיבוכיות המקום תהיה במקרה הגרוע -

ולכן סיבוכיות המקום הכוללת של הפעולה תהיה .

**StatusType GetMostViewedClasses(void\* DS, int numOfClasses, int\* courses, int\* classes)**

**הסבר המימוש:** ניגש לאיבר בעל הזמן הגדול ביותר ברשימת הזמנים, אשר שמור לנו ולכן הגעה אליו ב-  
נעבור על כל השיעורים השמורים ברשימה המקושרת של זמן זה ונשמור אותם במערך. במידה והגענו לסוף הרשימה ונשארו לנו עוד שיעורים לשמור, נעבור לזמן הקודם לו אשר קיימים בו שיעורים, ונעבור גם על כל השיעורים בו, וכך הלאה עד שסיימנו.  
  
**סיבוכיות זמן:**

* גישה לזמן הגדול ביותר, שמור לנו ולכן נעשה ב-
* מעבר על רשימות מקושרות של שיעורים עד שהגענו למספר המבוקש -
* מעבר על הרשימה המקושרת של הזמנים, יכול להיות מקסימום כמספר השיעורים שכן לא נעבור על זמנים שאין בהם שיעורים, ולכן סיבוכיות המעבר על הרשימה של הזמנים -

ולכן סיבוכיות הזמן הכוללת של הפעולה תהיה .

**סיבוכיות מקום:**   
אין הקצאות חדשות ואין רקורסיה ולכן סיבוכיות המקום הכוללת של הפעולה תהיה .

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

**סיבוכיות זמן:**

* מחיקת עצי הקורסים, הריסה ושחרור של עץ AVL מלא נעשית בסיבוכיות של גודל העץ ולכן -
* שחרור מערך השיעורים השמורים בכל קורס נעשה ב-
* שחרור הרשימה המקושרת של הזמנים, יכולים להיות מספר זמני צפייה שונים כמספר השיעורים הקיימים במערכת ולכן -

ולכן סיבוכיות הזמן הכוללת של הפעולה תהיה   
  
**סיבוכיות מקום:**

* הריסת עץ AVL, כאשר עומק הרקורסיה יהיה במקרה הגרוע ביותר בגובה העץ,   
  ולכן סיבוכיות המקום תהיה במקרה הגרוע -

ולכן סיבוכיות המקום הכוללת של הפעולה תהיה .

**הוכחת עמידה בסיבוכיות המקום:**

**בכל זמן נתון אנו שומרים במערכת מקסימום n קורסים, מקסימום m שיעורים ולכן מקסימום m זמנים שונים, כלומר מידע בגודל . כמו כן סיבוכיות המקום של כל הפעולות היא מקסימום .  
לכן סיבוכיות המקום עבור המבנה וכן הפעולות היא - .**