מבני נתונים 1
234218

תרגיל רטוב 1

מספר

הוגש ע"י:

דורון שלומוביץ	931202782
שם	מספר זהות
צחי אדרי	205854599
שם	מספר זהות

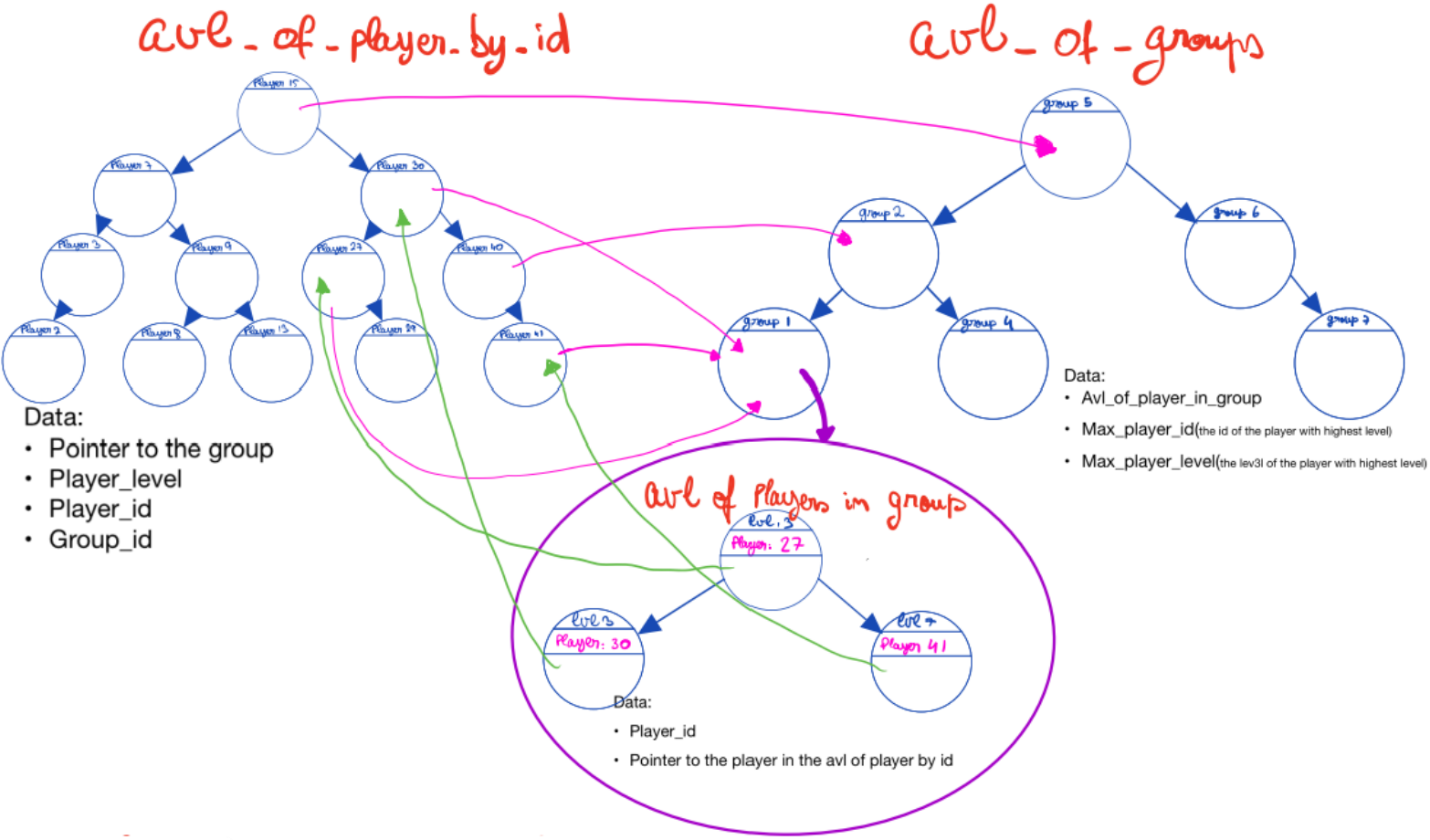
<u>ציון</u>:

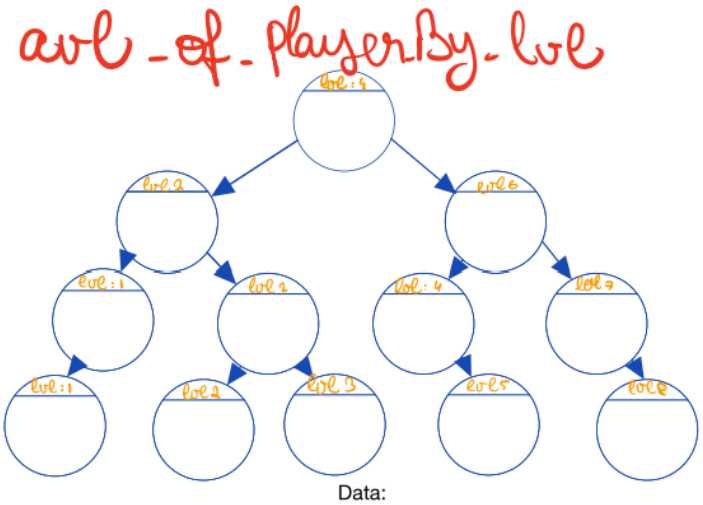
לפני בונוס הדפסה:

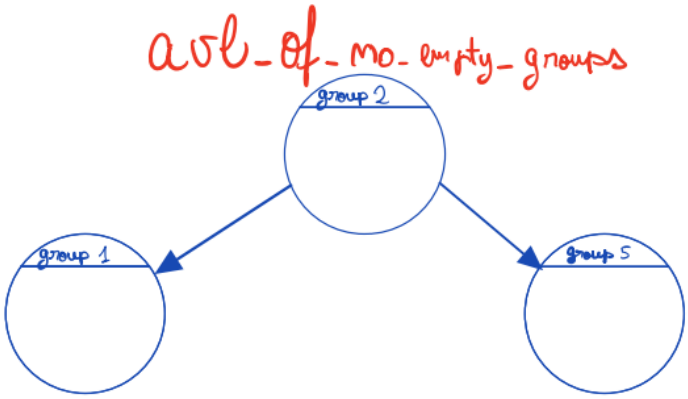
כולל בונוס הדפסה:

נא להחזיר לתא מס':

שרטוט להמחשת המבנה(בעמודים האחרונים שרטוטים נוספים עבור מספר פעולות):







Data in playerManager:

- Max_player_id(the id of the player with the highest level in all the data structure)
- Max_player_level(the level of the player with the highest level in all the data structure)

הסבר כללי על המערכת:

במהלך התרגיל נשתמש פעמים רבות בעץ AVL גנרי, שתוכנן בדיוק כפי שנלמד בהרצאות ובתרגולים כדי לאפשר פעולות של הכנסה ומחיקה ומציאת איבר בסיבוכיות לוגריתמית, כאשר כל צומת תכיל מפתח (key), מידע (data), מצביעים לבן שמאלי וימני. בנוסף כל עץ ישמור מצביע לצומת השורש שלו, ומספר שלם שיתאר את כמות הצמתים שהוא מכיל.
המערכת עצמה תורכב מ4 עצים חיצוניים, אשר באחד מהם רכיב ה'מידע' הוא עץ AVL בפני עצמו (עץ פנימי בכל צומת של העץ החיצוני).
העצים החיצוניים הינם:

- **עץ קבוצות**
- **עץ שחקנים לפי מספר מזהה**
- **עץ שחקנים לפי רמה**
- **עץ הקבוצות הלא ריקות**

<u>עץ קבוצות:</u> המפתחות בעץ זה יהיו מספר המזהה של הקבוצה (GroupID),ורכיב המידע יהיה מחלקה המכילה **עץ פנימי שיקרא עץ השחקנים בקבוצה** ואת פרטיו של השחקן בעל הרמה המקסימלית בקבוצה (מספר מזהה והרמה שלו).
בעץ הפנימי של השחקנים בקבוצה המפתחות יהיו בראש ובראשונה הרמה של כל שחקן, וכאשר מוכנסים לעץ שני שחקנים באותו הרמה, תינתן עדיפות לשחקן בעל מספר הזיהוי הקטן יותר (כלומר מזהה קטן יותר יהיה גבוה/ימינה יותר ממזהה נמוך יותר עבור שתי שחקנים בעלי אותה הרמה).
רכיב המידע בעץ זה יכיל את המספר המזהה של כל שחקן, ומצביע למבנה המכיל את פרטיו של השחקן, פרטי הקבוצה בה הוא מופיע ומצביע לרכיב המידע של הקבוצה בה הוא נמצא.

<u>עץ שחקנים לפי מספר מזהה:</u> המפתחות בעץ זה יהיו מספר המזהה של השחקן, ורכיב המידע יהיה מצביע למבנה המכיל את פרטיו של השחקן, פרטי הקבוצה בה הוא מופיע ומצביע לרכיב המידע של הקבוצה בה הוא נמצא.

<u>עץ שחקנים לפי רמה:</u>בדומה לעץ השחקנים בקבוצה, המפתחות בעץ זה יכילו את רמת השחקן והמספר המזהה שלו, וימוינו קודם לפי הרמה ולאחר מכן לפי המספר המזהה. רכיב המידע בעץ זה יהיה מספר שלם שהוא מספר המזהה של השחקן.

<u>עץ הקבוצות הלא ריקות:</u> כל צומת בעץ זה הינה קבוצה במערכת אשר יש בה לפחות שחקן אחד. המפתחות בעץ זה יהיו מספר הזיהוי של הקבוצה, והמידע יהיה פרטיו של השחקן המקסימלי בקבוצה (רמתו ומספר הזיהוי שלו).

מלבד העצים החיצוניים והפנימיים, ישמר במערכת עצמה את המידע לגבי השחקן המקסימלי במערכת כולה (רמה ומזהה).

הפעולות בהן המערכת תתמוך:

Init – פעולה זו תאתחל את המערכת. באתחול נקצה מקום לארבעת העצים החיצוניים שיצרנו (עצים ריקים), וכתלות בסוג העץ תתכן הקצאה נוספת גם למפתח/למידע של הצומת הראשון. יצירת מקום ריק הינה פעולה קבועה, ועשייה של פעולה שכזו מספר בודד של פעמים תשאיר אותנו בסיבוכיות של O(1), כנדרש.

AddGroup – פעולה זו תבצע הוספה של קבוצה חדשה (ריקה) למערכת, עם המזהה GroupID. ראשית נבצע חיפוש שהקבוצה אינה נמצאת כבר במערכת (כלומר מופיעה בעץ הקבוצות). במידה ונמצאת, נסיים. אחרת, נוסיף צומת חדשה לעץ הקבוצות, עם המפתח GroupID ועם מידע כפי

שנכתב בעץ הקבוצות. לצורך זה ניצור עץ פנימי ריק (עץ השחקנים בקבוצה). ההוספה של הקבוצה לעץ הקבוצות תתבצע באמצעות פעולת ההכנסה שקיימת בעץ.
חיפוש הקבוצה בעץ הקבוצות יתבצע בסיבוכיות לוגריתמית כתלות במספר הקבוצות שבעץ, כנ"ל לגבי הכנסתו לעץ, ויצירת העץ הפנימי הינה פעולה קבועה. בסך הכל סיבוכיות הזמן לביצוע פעולת הוספת הקבוצה תעלה לנו סיבוכיות של O(logK), כאשר K הינו מספר הקבוצות שבמערכת.

AddPlayer – פעולה זו תוסיף שחקן חדש למערכת, עם מספר זיהוי PlayerID, מספר זיהוי לקבוצה GroupID ורמתו הנוכחית של השחקן (Level).
ראשית נבדוק שהקבוצה אכן קיימת במערכת (כלומר נמצאת בעץ הקבוצות) ושהשחקן טרם הוכנס למערכת (כלומר לא נמצא בעץ השחקנים לפי מספר מזהה), באמצעות שני חיפושים בעצים. במידה ותנאים אלה לא מתקיימים, נסיים. אחרת, נבצע את השלבים הבאים:

- נבדוק האם הGroupID קיים בעץ הקבוצות הלא ריקות. אם לא, נכניס לעץ זה צומת חדשה עם פרטי הקבוצה שלנו (אם הוא לא נמצא סימן שעד כה הגודל שלו היה 0, וכעת בסיום תהליך הכנסת השחקן יש לדאוג שימצא בעץ).
- נעדכן את השחקן המקסימלי במערכת כולה (כלומר נבדוק האם השחקן שאנחנו מעוניינים להכניס אמור להפוך לשחקן המקסימלי במקום הקודם).
- נכניס לעץ השחקנים לפי רמה צומת חדשה עם פרטי השחקן החדש.
- נמצא את הקבוצה שלנו בעץ הקבוצות ונשמור במשתנה זמני את הDATA שלה (כדי לקבל גישה לעץ השחקנים שנמצא בתוכה).
- נכניס לעץ השחקנים לפי מזהה את השחקן החדש.
- נעדכן את המקסימום של הקבוצה אליה הוספנו במידת הצורך.
- נוסיף את השחקן לעץ השחקנים בקבוצה (הפנימי שנמצא בתוך המידע של הקבוצה).

הסבר על הסיבוכיות: בדיקה אם הקבוצה בעץ הקבוצות, סדר גודל של log(k) כאשר k מספר הקבוצות, בדיקה אם השחקן טרם הוכנס למערכת, log(n) כאשר n מספר השחקנים, בדיקה אם הקבוצה בעץ הקבוצות הלא ריקות, שזה לכל היותר log(n) (n שחקנים) שכן הקבוצות הריקות לא נכנסות, לכן באופציה הגרועה ביותר כל שחקן מופיע בקבוצה שונה ולכן יש n קבוצות לא ריקות, הכנסת הקבוצה לעץ זה תהיה גם היא log(n).
עדכון השחקן המקסימלי הינה פעולה בודדת שכן יש לנו שדות במחלקת המערכת ששומרות את פרטי השחקן המקסימלי.
הכנסה לעץ השחקנים לפי רמה תהיה גם היא log(n), שכן ישנם n שחקנים.
מציאת הקבוצה בעץ הקבוצות log(n), הכנסה לעץ השחקנים לפי מזהה לכל היותר log(n) (רק במידה וכל השחקנים בקבוצה אחת). עדכון המקסימום בקבוצה פעולה בודדת, והוספה של השחקן לעץ השחקנים בקבוצה log(n) במקרה הגרוע ביותר.
סה"כ סיבוכיות O(log(k)+log(n)).
מצב המערכת לאחר הפעולה:המידע על המקסימום במערכת שמור בשדה של המערכת. עץ הקבוצות הלא ריקות מעודכן עם הקבוצה הנוכחית ועם פרטי השחקן המקסימלי בקבוצה. יש צומת בשביל השחקן בעץ השחקנים לפי רמה ובעץ השחקנים לפי מזהה וכן בעץ הפנימי של השחקנים בקבוצה בו הוא נמצא.

RemovePlayer – פעולה זו תמחוק את השחקן בעל המזהה PlayerID מהמערכת.
ראשית נבדוק האם המזהה חוקי והשחקן אכן נמצא במערכת. אם תנאים אלו לא מתקיימים, נסיים. נבצע מספר פעמים חיפוש בעץ השחקנים לפי מזהה על מנת לקבל פרטים הנחוצים למימוש כמו הרמה שלו, מספר המזהה של הקבוצה בה הוא מופיע ואת המצביע למידע בקבוצה שבה הוא נמצא(מספק גישה לעץ הפנימי של הקבוצה), מבלי לחפש בעץ הקבוצות. בכך מתאפשר לנו לבצע את המחיקה בסיבוכיות לוגריתמית במספר השחקנים, בלי תלות במספר הקבוצות.
כעת נבצע מחיקה של השחקן מעץ השחקנים לפי רמות. במידה ועץ השחקנים שלנו כרגע ריק, נעדכן את ערכי המקסימום במערכת למצב שמסמן מקסימום לא קיים. אחרת, נעדכן את המקסימלי במערכת באמצעות מציאת הצומת המקסימלית בעץ השחקנים לפי רמה.

נמחוק את השחקן מהעץ הפנימי בקבוצה שבה הוא נמצא. נבדוק האם הקבוצה ריקה ובמידה וכן נמחוק את הצומת מעץ הקבוצות הלא ריקות. אחרת, נחפש את המקסימום החדש בעץ הפנימי ונעדכן את הצומת של הקבוצה בעץ הקבוצות הלא ריקות.
לסיום נמחק את השחקן מעץ הקבוצות לפי מזהה.
הסבר סיבוכיות:בדיקה האם השחקן במערכת (כלומר חיפוש בעץ השחקנים לפי מזהה) log(n).
החיפושים הנוספים בעץ השחקנים לפי מזהה גם הם log(n).
מחיקת השחקן מעץ השחקנים לפי רמות – log(n), עדכון שדה המקסימום במערכת פעולה בודדת, ו/או מציאת השחקן המקסימלי החדש log(n). מחיקת צומת מהעץ הפנימי log(n) (במקרה הגרוע ביותר). מחיקת הצומת מעץ הקבוצות הלא ריקות log(n) (ולא log(k) שכן מספר הקבוצות הריקות יכול להיות גדול ממספר השחקנים, אבל מספר הקבוצות הלא ריקות קטן ממספר השחקנים במערכת).חיפוש בעץ הפנימי של השחקנים בקבוצה וגממחיקה נוספת של השחקן מעץ הקבוצות לפי מזהה log(n).
סה"כ סיבוכיות O(log(n)).
מצב המערכת לאחר הפעולה:אם מקסימום של המערכת/של הקבוצה שבו השחקן הופיע השתנה, אז הוא עודכן בהתאם. במידה והקבוצה הפכה ריקה אז נמחקה מעץ הקבוצות הלא ריקות. השחקן נמחק מעץ השחקנים לפי מזהה ועץ השחקנים לפי רמה, ובעץ הפנימי של השחקנים שבקבוצה בה הוא מופיע.

ReplaceGroup – פעולה זו תסיר את הקבוצה בעלת המזהה GroupID, והקבוצה בעלת המזהה ReplacementID תקבל את השחקנים שהיו בGroupID.
ראשית בודקים האם המספרי הזיהוי של הקבוצות תקינים ומחפשים בעץ הקבוצות האם שניהם נמצאים. במידה ואחד התנאים לא מתקיים, נסיים.
במידה והקבוצה GroupID ריקה, נמחק אותה מעץ הקבוצות ונסיים.
אחרת, בדומה לאלגוריתם למיזוג עצים שנלמד בתרגול, נעביר את עץ השחקנים הפנימי בכל אחת מהקבוצות למערך. נמזג את המערכים למערך ממוין באמצעות אלגוריתם merge. ניצור עץ שחקנים פנימי חדש ריק, בגודל של מספר השחקנים בקבוצה ReplacementID + GroupID.
נבצע סיור על עץ השחקנים הפנימי שבקבוצה GroupID (הישנה) כדי לעדכן את המידע שכל שחקן מחזיק על הקבוצה בה הוא נמצא לקבוצה ReplacementID (החדשה), ונמחוק בכל פעם את הצומת שהמידע בה שונה מהעץ (מידע זה משפיע על השחקן עצמו, שאינו נמחק מהמערכת).
כעת נמחוק את הקבוצה הישנה מעץ הקבוצות הלא ריקות.
נמחוק את הקבוצה הישנה מעץ הקבוצות.
מעדכנים את המקסימום של הקבוצה ReplacementID במידה והשתנה (לצורך פעולה זו, אין צורך לסייר בעץ אלה נשנה את המקסימום להיות האיבר האחרון במערך הממוזג שכן הוא ממוין).
בודקים אם הקבוצה ReplacementID נמצאת בעץ הקבוצות הלא ריקות. במידה ולא, נוסיף אותה עם פרטי המקסימום של השחקן המקסימלי. אחרת, נבדוק האם השתנה המקסימום ונעדכן בהתאם.
הסבר על הסיבוכיות:שני חיפושים בעץ הקבוצות – $\log(n_{group})+\log(n_{replacement})$. אלגוריתם המיזוג עצים שנלמד בתרגול, כולל בתוכו שתי מסלולי inorder בכל עץ פנימי של קבוצה (להעביר את השחקנים למערך) , $n_{group+}\ n_{replacement}$, לאחר מכן merge ($n_{group+}\ n_{replacement}$). יצירת עץ ריק בגודל הכולל לוקחת גם היא ($n_{group+}\ n_{replacement}$).
סיור בעץ השחקנים הפנימי $n_{replacement}$. מחיקת הקבוצה הישנה מעץ הקבוצות הלא ריקות ומעץ הקבוצות log(k), חיפוש בעץ הקבוצות הלא ריקות log(k).
בסה"כ סיבוכיות $O(\log(k)+\ n_{group+}\ n_{replacement})$.
מצב המערכת לאחר הפעולה: הקבוצה GroupID אינה בעץ הקבוצות, כל השחקנים שהיו בה עברו לReplacementID, המקסימום התעדכן בReplacementID וGroupID נמחקה מעץ הקבוצות הלא ריקות במידה והייתה שם. המערכים שהתעסקנו בהם במהלך הפעולה נמחקו ואינם נמצאים עוד. המקסימום במערכת כולה לא התשנה שכן לא נוספו/נמחקו שחקנים נוספים.

IncreaseLevel – פעולה זו תגדיל את רמתו של השחקן PlayerID בLevelIncrease.
ראשית, נבדוק האם השחקן קיים במערכת, אחרת נסיים.
ניגשים למיקום השחקן בעץ השחקנים לפי מזהה ומעדכנים את רמתו החדשה.
נבצע מספר פעמים חיפוש בעץ השחקנים לפי מזהה על מנת להביא פרטים הנחוצים למימוש כמו הרמה הנוכחית שלו, מזהה הקבוצה אליה הוא שייך ואת המצביע למבנה הקבוצה בה הוא מופיע.
מוחקים את השחקן מהעץ השחקנים הפנימי שבקבוצה ומעץ השחקנים לפי רמה באמצעות הפרטים הישנים של השחקן, וכעת מכניסים בחזרה לכל אחד מהעצים הללו באמצעות הפרטים החדשים שלו, (כלומר בעדכון הרמה הנוכחית), פעולת הכנסה זו תעדכן את המקסימלי החדש בקבוצה במידת הצורך. נעדכן את הפרטים של המקסימום בקבוצה זו בעץ הקבוצות הלא ריקות.
נעדכן את המקסימום במערכת כולה במידת הצורך.
הסבר על הסיבוכיות: חיפוש השחקן בעץ הקבוצות לפי מזהה, log(n). מחפשים שוב את השחקן באותו העץ על מנת לעדכן את רמתו. מבצעים עוד מספר חיפושים כאלה על מנת להביא את הפרטים הנחוצים, log(n). ניגשים לעץ הפנימי של הקבוצה בה הוא נמצא (קיבלנו מצביע ישירות לשם כתוצאה מהחיפוש הקודם) ומסירים אותו מהעץ (לכל היותר log(n) במידה וכל השחקנים בקבוצה אחת), ומוסיפים אותו בחזרה log(n). מסירים ומכניסים אותו גם מעץ השחקנים לפי רמה log(n). מעדכנים את המקסימלי בצומת הקבוצה בעץ השחקנים הלא ריקים (לשם כך, עשינו חיפוש בעץ הקבוצות הלא ריקות, שמספרם לכל היותר n כמספר השחקנים, שכן קבוצה ריקה לא נכנסת לעץ זה {כלומר כל שחקן שנכנס למערכת מגדיל לכל היותר את מספר הקבוצות הלא ריקות באחד, ולכן מספר הקבוצות הלא ריקות קטן ממספר השחקנים}), log(n). ומעדכנים את המקסימלי במערכת, פעולה קבועה.
בסה"כ סיבוכיות O(log(n)).
מצב המערכת לאחר פעולה זו: רמתו של השחקן השתנתה בהתאם, השחקן הוזז ממקומו בעץ השחקנים לפי רמה בהתאם למיקום המיועד לו ברגע זה, וגם בעץ הפנימי בקבוצה בו הוא נמצא (עץ זה ממוין לפי רמה גם כן).במידת הצורך השתנה המקסימלי של המערכת ו/או המקסימלי של הקבוצה (ובמידה והמקסימלי של הקבוצה גם השתנה, ישתנה גם המקסימלי בצומת הקבוצה בעץ הקבוצות הלא ריקות).

GetHighestLevel – פעולה זו תחזיר את מזהה השחקן המקסימלי בקבוצה בעלת המזהה GroupID.
אם GroupID קטן מאפס, נחזיר את המקסימלי במערכת, אשר מאופן תחזוק המערכת נשמר ומתעדכן עם כל פעולה. במידה וGroupID תקין, נבצע חיפוש בעץ הקבוצות על מנת לבדוק אם אכן נמצא קיים. אם קיים, נחזיר את המקסימלי בקבוצה (באופן דומה, מאופן תחזוק המערכת המקסימלי בקבוצה נשמר ומתעדכן גם הוא).
סה"כ סיבוכיות O(log(k)) כאשר k מספר הקבוצות, וזאת משום שמבצעים מספר חיפושים בעץ הקבוצות בלבד. גם במקרה בו נצטרך להחזיר את המקסימלי במערכת כולה לא נצטרך לבצע חיפוש מיוחד כלשהו שכן המידע על כך כבר שמור.

GetAllPlayerByLevel – פעולה זו תחזיר את כל השחקנים ששייכים לקבוצה בעלת המזהה GroupID, ממויינים לפי השלב שלהם.
נפריד לשתי מקרים.

- אם GroupID קטן מאפס: נרצה להחזיר את כל השחקנים בכל המערכת ממוינים לפי השלב שלהם בסדר. לכן ניגש לעץ השחקנים לפי רמות ונבצע בעץ סיור inorder הפוך (כלומר סיור בו קודם ניגשים לימין, לאחר מכן לשורש ובסוף לשמאל). מכיוון שinorder רגיל מחזיר לנו את האיברים בצורה ממוינת עולה, אם נהפוך נקבל את אותה התוצאה רק הפוך, כלומר בסדר יורד. בכל הגעה לצומת נכניס את פרטי השחקן למערך אותו נחזיר.
  בסה"כ סיבוכיות של O(n), כאשר n הוא מספר השחקנים הכולל במערכת.
- אחרת, נחפש בעץ הקבוצות את הקבוצה בעלת המזהה המתאים,ונעשה בדיוק את אותו התהליך רק שהפעם בעץ הפנימי של השחקנים בקבוצה (עץ זה ממוין גם הוא לפי הרמה).

בסה"כ סיבוכיות של $O(n_{groupID}+\log(k))$, לוגריתמי לשם חיפוש הקבוצה ואז לינארי כתלות במספר השחקנים שבקבוצה.
מצב המערכת לא השתנה לאחר פעולה זו.

GetGroupsHighestLevel – פעולה זו תחזיר עבור כל אחד מnumOfGroups הקבוצות הלא ריקות בעלות המזהה GroupID הכי קטן את השחקן המקסימלי במערך ממוין לפי GroupID בסדר עולה.
ראשית נוודא כי numOfGroups קטן ממספר הקבוצות שבמערכת, שכן הפעולה לא אפשרית במידה והוא גדול יותר.
כעת נעשה סיור inorder בעץ הקבוצות הלא ריקות, ובכל מעבר במסלול נוסיף למערך את מזהה השחקן המקסימלי שנמצא שם. סיור זה יתן לנו את השחקנים בסדר הרצוי כיוון ומערך הקבוצות הלא ריקות מסודר לפי מספרי הזיהוי של הקבוצות, כלומר הקבוצה בעלת המזהה הקטן ביותר תופיע בצומת השמאלית הקיצונית ביותר בעץ. לכן סיור inorder כזה יתחיל בהכי שמאלית (כלומר בצומת הנכונה) ובהתאם לinorder ימשיך בהתאם. לסיור זה נוסיף כפרמטר את numOfGroups, על מנת שנוכל לדאוג שהסיור יעצר לאחר ביקור בnumOfGroups צמתים, ולא בכל העץ.
סה"כ סיבוכיות $O(numOfGroups + \log(k))$, הסבר – מסלול הinorder ראשית ירד לתחתית עץ הקבוצות הלא ריקות, כלומר יעשה מספר פעולות כגובה העץ שהוא $\log(k)$. מכשירדנו עד לתחתית השמאלית נטייל בnumOfGroups צמתים ואז נעצור.

Quit – פעולה זה תשחרר את המבנה לחלוטין ותמחק את כל המידע שבו.
לצורך פעולה זו נמחק את כל הצמתים בעץ השחקנים לפי רמה, בעץ השחקנים לפי מזהה ובעץ הקבוצות הלא ריקות. לאחר מכן נמחוק בכל צומת בעץ הקבוצות את העץ הפנימי של השחקנים שבקבוצה זו, ולסיום נמחק את כל הצמתים בעץ הקבוצות.
סה"כ סיבוכיות מקסימלית למחיקת עץ השחקנים לפי רמה – n, לעץ השחקנים לפי מזהה – n, לעץ הקבוצות הלא ריקות -k. מחיקת כל השחקנים מעצי השחקנים בקבוצות תהיה גם היא סיבוכיות של n, שכן בכל הקבוצות סך השחקנים שווה לסך השחקנים הכולל שבמערכת. לסיום הסרת עץ הקבוצות -k. לכן $O(n+k)$.

סיבוכיות מקום:

סיבוכיות המקום הכוללת תהיה $O(n+k)$, וזאת כי:

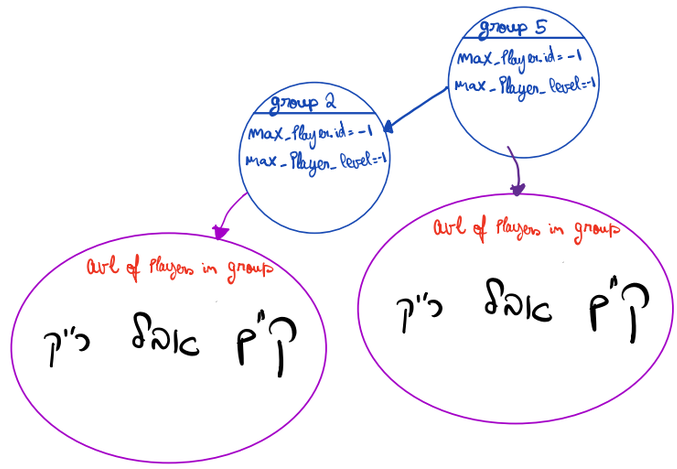
- עץ השחקנים לפי רמה $O(n)$
- עץ השחקנים לפי מזהה $O(n)$
- עץ הקבוצות הלא ריקות $O(k)$
- עץ הקבוצות $O(n+k)$ (שכן הוא מכיל בתוכו n קבוצות וk שחקנים המפוזרים בין הקבוצות).

DS = Init()

addgroup(DS, 2)

addgroup(DS, 1)

avl_of_groups



avl_of_player_by_id

כ"ק אבל ק"פ

avl_of_playerBy_lvl

כ"ק אבל ק"פ

avl_of_no_empty_groups

כ"ק אבל ק"פ

max_Player_id = -1

max_Player_level = -1

---

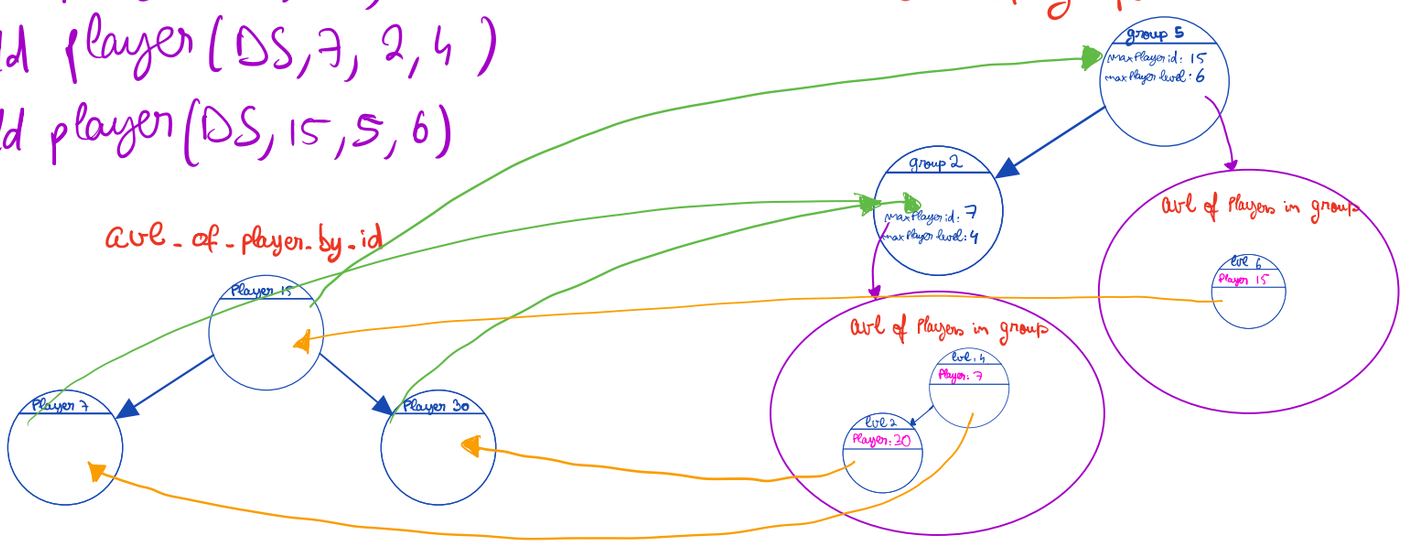
add player(DS, 30, 2, 2)

add player(DS, 7, 2, 4)

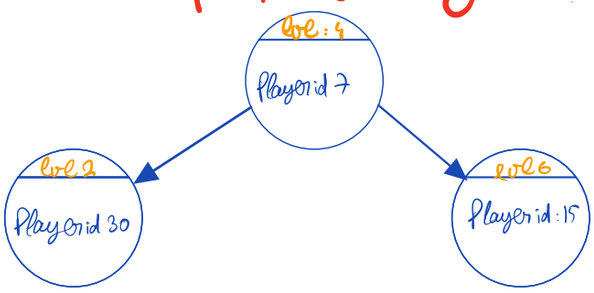
add player(DS, 15, 5, 6)

avl_of_groups

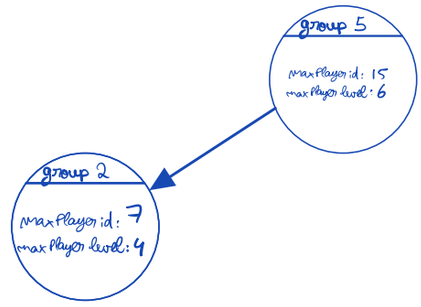
avl_of_player_by_id



avl_of_playerBy_lvl



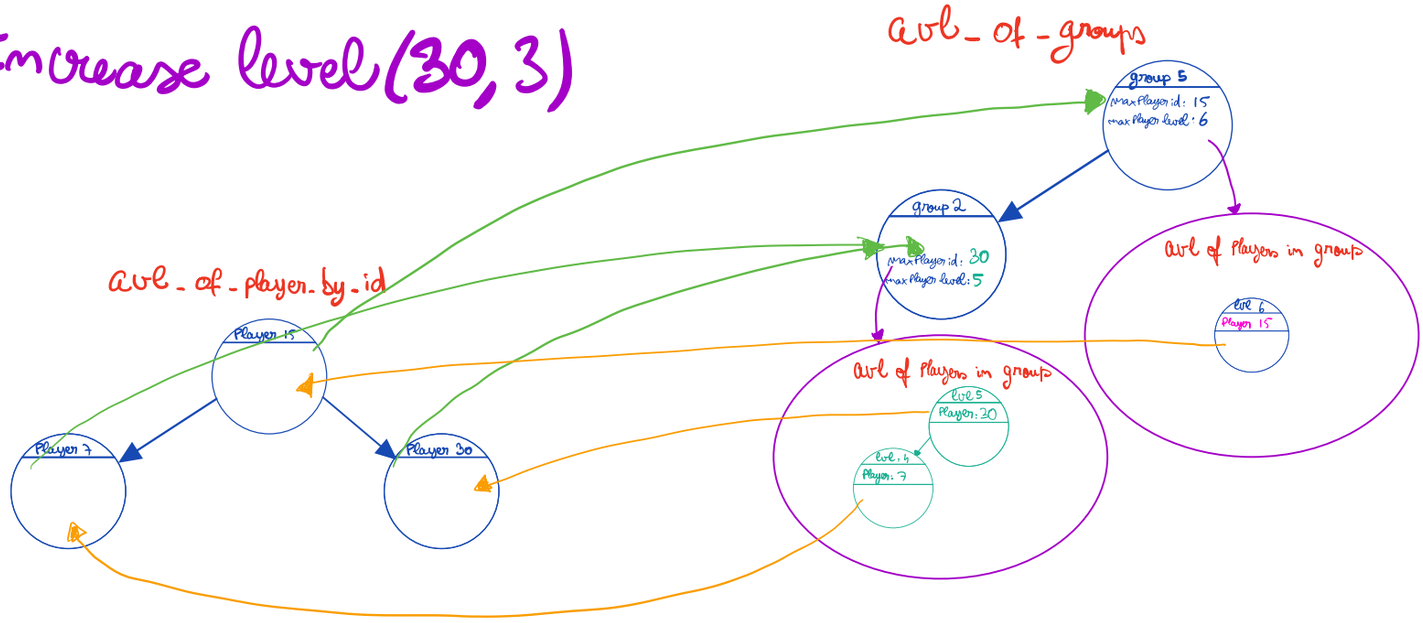
avl_of_no_empty_groups



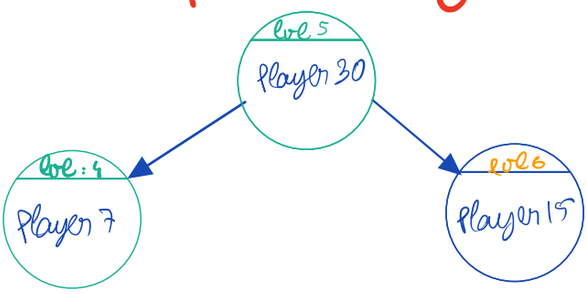
max_Player_id = 15

max_Player_level = 6

Increase level(30, 3)



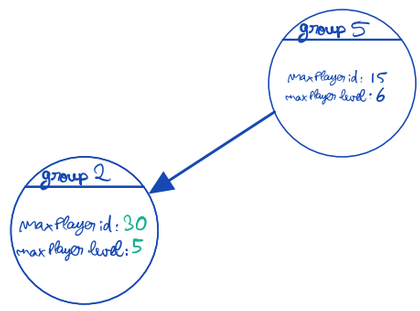
avl_of_PlayerBy_lvl



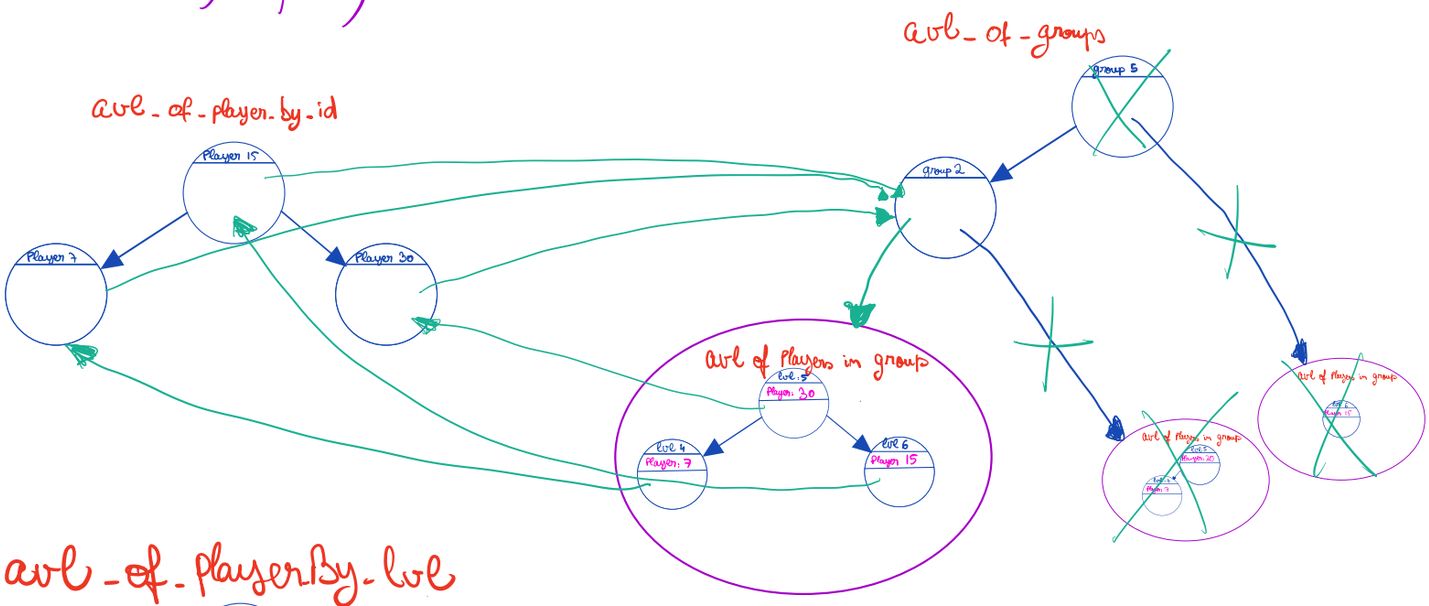
max_Player_id = 15

max_Player_level = 6

avl_of_no_empty_groups

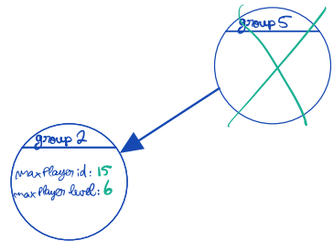


Replace(DS, 5, 2)



avl_of_PlayerBy_lvl

avl_of_no_empty_groups



max_Player_id = 15

max_Player_level = 6