**מבני הנתונים**

לצורך מימוש הדרישות השתמשנו במספר מבני נתונים, שיתוארו להלן:

* **Ranked AVL Tree** – מבנה נתונים המממש עץ דרגות מסוג AVL – הממויין לפי דרגת הצמתים. כל צומת, פרט לדרגתו, מחזיק את ה-id שלו(יחיד – משמש לצורך השוואה והחזרה), את רמתו ואת הרמות המקסימליות בתתי העץ השמאלי והימני שלו. לאחר כל פעולה בעץ נשמרת השמורה הבאה:

בכל צומת מעודכנות הרמות המקסימליות של תתי-עציו.

תומך בפעולות:

* – הכנסת איבר לעץ.
* – הוצאת איבר מהעץ.
* – מחזירה את ה-id של האיבר מהדרגה הגבוהה ביותר.
* – החזרת משתנה בוליאני לגבי מצב הצומת בעץ.

השוואה בין צמתים תתבצע באופן הבא:

* אם המציינים(id) שווים – החזר שווה.
* אחרת, אם הדרגה של הצומת הראשון קטנה מהשניה – החזר קטן.
* אחרת, אם הדרגות שוות והמציין הראשון קטן מהשני – החזר קטן.
* אחרת, החזר גדול.

סיבוכיות זמן הפעולות – כפי שנלמד בכיתה:

Init –

Insert, Remove, Find, doesExist –

getStrongest – הסיבוכיות חסומה בגובהה העץ.

סיבוכיות מקום –

כאשר n הוא מספר הצמתים.

* **Union-Find** – מבנה נתונים התומך בקבוצות זרות של מספרים, כאשר הוא מאותחל ע"י מספר האיברים n. משתמש בעצים הפוכים עם איחוד לפי גודל וכיווץ מסלולים.

תומך בפעולות:

* – מוצא ומחזיר את נציג מחלקת השקילות שבקבוצה עם x.
* – מאחד את הקבוצות של מחלקות השקילות של x ו-y.
* – מחזיר משתנה בוליאני הבודק האם x ו-y נמצאים באותה מחלקת שקילות.

סיבוכיות זמן הפעולות – כפי שלנמד בכיתה:

Init – O(n)

Find, Union, areConnected – משוערך.

סיבוכיות מקום – O(n)

כאשר n הוא מספרים האיברים השונים ביצירה.

* **Hash-Table** – מבנה נתונים המממש טבלת ערבול דינאמית.

הערבול מתבצע באמצעות שיטת double-hashing ובאמצעות מערך דינמי. גודל המערך משתנה באופן דינמי, ע"י threshold.

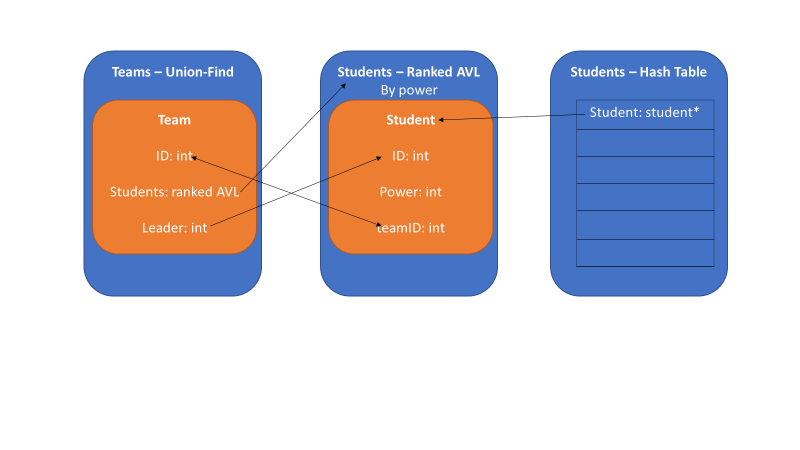
תומך בפעולות:

* Insert(key, value)– הכנסת איבר לטבלה.
* – הוצאת איבר מהטבלה.
* – החזרת את ה-value המשויך ל-key.

סיבוכיות זמן הפעולות – O(1) בממוצע על הקלט באופן משוערך, כפי שנלמד בכיתה.

**פרטי המימוש של מבני הנתונים**

* **עצי ה-AVL – סטודנטים** 
  + כל צוות מחזיק עץ עם הסטודנטים ששייכים אליו.
  + העצים ממוינים לפי רמות כח הסטודנטים והדרגה היא סכום הדרגות של תתי העצים שלו כולל הוא עצמו – סכום רמות הכח בתתי העץ (כולל עצמו).
* **Union-find**
  + כל צוות ייוצג ע"י קבוצה זרה.
* **Hash-Table**
  + בטבלת הערבול נשמור את המוטנטים עצמם לפי המזהה שלהם.



**תיאור הפעולות**

Init: נאתחל את שלושת מבני הנתונים הקיימים-

* ניצור nעצי דרגות מסוג AVL ריקים O(n) (אחד לכל צוות)
* נאתחל טבלת ערבול ונאתחל מערך בגודל התחלתי קבוע O(1) (כפי שנלמד בהרצאה)
* ניצור n קבוצות סינגלטונים בUnion-Find על פי ה-n שקיבלנו כקלט. O(n)

סה"כ סיבוכיות הזמן הינה O(n) במקרה הגרוע כנדרש.

AddStudent: נבצע הוספה של הסטודנט למערכת-

* נוסיף את הסטודנט לטבלת הערבול על פי הStudentID שלו . O(1) משוערך בממוצע על הקלט.
* נוסיף את הסטודנט למערך העצים לפי find(team) הספציפי שלו. O(logk+log\*n)
* נבצע בדיקה ועדכון לmaxID של הצוות שאליו הוספנו את הסטודנט. O(1)

סה"כ סיבוכיות הזמן הינה O(logk+log\*n) משוערך בממוצע על הקלט כנדרש.

RemoveStudent: נבצע הסרה של הסטודנט מן המערכת-

* נסיר את הסטודנט מטבלת הערבול על פי הStudentID שלו . O(1) משוערך בממוצע על הקלט.
* נסיר את הסטודנט ממערך העצים לפי find(team) הספציפי שלו. O(logk+log\*n)
* נבצע בדיקה ועדכון לmaxID של הצוות שממנו הסרנו את הסטודנט. O(logk) במקרה הגרוע.

סה"כ סיבוכיות הזמן הינה O(logk+log\*n) משוערך בממוצע על הקלט כנדרש.

JoinTeams: נבצע איחוד לקבוצות באופן הבא-

* נבצע איחוד על הקבוצות T1 T2 בעזרת פעולת Union. O(log\*n) משוערך.
* נאחד את עצי המערכים הרלוונטים ונשמור אותם במקום קבוע מראש. O(k1+k2)

סה"כ סיבוכיות הזמן הינה O(log\*n+ k1+k2) משוערך כנדרש.

TeamFight: נבצע קרב בין 2 הקבוצות-

* תחילה נמצא את שני הצוותים בעזרת find על שני הקבוצות. O(log\*n) משוערך.
* ניגש לעצי הavl המתאימים ובעזרת אלגוריתם select על שני העצים נוציא את סכום n החזקים ביותר(אלגוריתם select יביא אותנו אל הנק' שממנה המידע הנוסף שיישמר בצומת הוא בדיוק סכום כל הימניים ממנו, כלומר גדולים ממנו ולכן זהו אכן סכום n החזקים ביותר בעץ הנוכחי). O(logk).
* נעדכן נצחונות של הצוות הרלוונטי במידה וקיימים.

סה"כ סיבוכיות הזמן הינה O(log\*n+ logk) משוערך כנדרש.

GetNumOfWins: נמצא את הצוות teamID במערכת ונחזיר את הערך ששמור בתא wins של הצוות.

סה"כ סיבוכיות הזמן הינה O(log\*n) משוערך כנדרש.

GetStudentTeamLeader: נרצה להחזיר את המזהה של ראש הקבוצה-

* תחילה נחפש את הסטודנט על פי הid שלו בטבלת הערבול, ונשמור את הteamID שלו. O(1) משוערך בממוצע על הקלט.
* נחפש את הצוות בעזרת המידע ששמרנו במערכת. O(log\*n) משוערך.
* ניגש אל תא המערך המתאים לצוות שממצאנו ונוציא ממנו את maxID. O(1).

סה"כ סיבוכיות הזמן הינה O(log\*n) משוערך בממוצע על הקלט כנדרש.

Quit: נמחק ונסיר את שלושת מבני הנתונים הקיימים-

* נסיר את כל העצים במערך הצוותים. O(n+k) במקרה הגרוע.
* נסיר את טבלת הערבול O(1).
* נסיר את UF. O(n)

סה"כ סיבוכיות הזמן הינה O(n+k) במקרה הגרוע כנדרש.