הבנה שלנו יורכב מ 3 מבנים עיקריים :

1. מבנה של Employee, והוא מכיל השדות הבאים:

Int id;

Int company\_id;

Int grade;

Int salary;

1. מבנה של Company, והוא מכיל השדות הבאים:

Int id;

Double value;

Int enterprise\_value;

Hashtable employees\_with\_zero\_salary;

Ranktree employees\_with\_salary;

Int number\_of\_employees\_with\_no\_salary;

Int sum\_of\_grades\_with\_no\_salary;

Int grade\_bump\_for\_zero\_salary;

1. מבנה של Industry, והוא מכיל השדות הבאים:

Int number\_of\_companies;

UnionFind companies;

Hashtable employees;

Ranktree employees\_with\_salary;

Int number\_of\_employees\_with\_no\_salary;

Int sum\_of\_grades\_with\_no\_salary;

Int grade\_bump\_for\_zero\_salary;

Init(int k) –

נבדוק את הערך של k ואם הוא קטן/שווה לאפס אז נחזיר FAILURE. אחרת , ניצור UF עם k חברות ב O(k) כפי שנלמד בכיתה עם עץ לעובדים וטבלת ערבול אשר ניצור אותם ב O(1) ובסוף נחזיר את המבנה של Industry ב O(k) ונסיים כנדרש!

addEmployee((void \*DS,int EmployeeID, int CompanyID, int Grade) –

נבדוק שהנתונים שקיבלנו תקינים. אחרת, נחזיר שגיאה מתאימה מסוג INVALID\_INPUT.

נחפש את העובד בעל מזהה EmployeeID ב Hashtable של כלל העובדים במבנה ב O(1) ואם נמצא אותו אז נחזיר שגיאתFAILURE ונסיים ב O(1) בממוצע.

נחפש את החברה בעלת מזהה CompanyID ב O(log\*(k)) משוערך כפי שנלמד בכיתה,ונסמנה company\_to\_find, ואז נכניס את העובד החדש ל Hashtable של העובדים במבנה של החברה עצמה וגם כן למבנה הכללי של Industry O(1) בממוצע.ונבצע ההוספות הבאות :

Company\_to\_find->number\_of\_employees\_with\_no\_salary++

Company\_to\_find->sum\_of\_grades\_with\_no\_salary+=Grade

DS->number\_of\_employees\_with\_no\_salary++

DS->sum\_of\_grades\_with\_no\_salary+=Grade

וכך נסיים ההכנסה ב O(log\*(k)) משוערך בממוצע כנדרש!

removeEmployee(void \*DS, int EmployeeID) –

נבדוק שהנתונים שקיבלנו תקינים. אחרת, נחזיר שגיאה מתאימה מסוג INVALID\_INPUT.

נחפש את העובד בעל מזהה EmployeeID ב Hashtable של כלל העובדים במבנה ב O(1) ,ונסמנו emp,ואם לא נמצא אותו אז נחזיר שגיאת FAILURE ונסיים ב O(1) בממוצע.

אם כן מצאנו אותו אז נחפש את החברה אליה הוא שייך ב O(1) בקבוצות ששמרנו(הסבר: החיפוש של הקבוצה מתבצע ב O(1) כי ב UF לאחר כל פעולת איחוד בין קבוצות אנחנו מעדכנים את החברה אליה הם שייכים להיות השורש החדש של איחוד החברות ולכן בעת החיפוש על הקבוצה ב UF החיפוש יתרחש ב O(1) כי האב של השורש הוא NULL ו החיפוש מסתיים באיטירציה אחת!),ונסמנה company\_to\_find, נוציא אותו מטבלת הערבול של העובדים בחברה וגם במבנה הכללי של Industry ב O(1) בממוצע, ואז אם יש לו salary > 0 נוציא אותו מעץ הדרגות של העובדים בחברה ובמבנה הכללי ב O(log(n)).

ואם ה salary שלו היא 0, אז נבצע :

Company\_to\_find->number\_of\_employees\_with\_no\_salary--

Company\_to\_find->sum\_of\_grades\_with\_no\_salary-=emp->Grade

DS->number\_of\_employees\_with\_no\_salary--

DS->sum\_of\_grades\_with\_no\_salary-=emp->Grade

acquireCompany(void \*DS, int AcquirerID, int TargetID, double Factor) –

נבדוק שהנתונים שקיבלנו תקינים. אחרת, נחזיר שגיאה מתאימה מסוג INVALID\_INPUT.וגם נחפש את החברה של AcquirerID ושל TargetID ב UF של החברות ב O(log\*(k)) משוערך כפי שנלמד בכיתה,ואם נמצא שהן שייכות לאותה קבוצה אז נחזיק INVALID\_INPUT.

נחפש את שתי החברות ב O(log\*(k)) משוערך ב UF,ונסמנם target ו acquirer.

נשתמש בפעולת Union ב UF כדי לאחד הקבוצות ו נשמור את השורש של האיחוד במשתנה new\_root.

ולאחר מכן נמזג את שתי טבלאות הערבול של העובדים בחברות לתוך טבלה אחת שנשים אותה בתוך החברה של acquirer.ו נמזג גם את שני העצים של העובדים לתוך עץ אחד כפי שנלמד בכיתה וכל זה מתבצע ב בממוצע(כמובן על הדרך משנים את החברה של העובדים להיות new\_root כך שבעת החיפוש על החברה שלהם ב UF נמצא אותה ב O(1)!).ובנוסף לכך נאחד את המשתנים של סכום דרגות העובדים שיש להם Salary = 0 ומספר העובדים בעלי Salary = 0 ע"י סכימתם ב O(1).וכך נסיים הפעולה ב כנדרש.

(\*) הסבר על איחוד הקבוצות ב UF: בכל חוליה בעץ ההפוך נשמור מידע נוסף בשם offset שהוא "ההפרש שצריך לחסרו מערך שורש הקבוצה כדי לקבל ערך החברה האמיתי ". בעת איחוד קבוצות אם acquirer הוא השורש של איחוד הקבוצות אז נבצע : target->offset +=(val\_inc+acquirer->value – target->value)

Acquirer->enterprise\_value +=val\_inc.

אחרת, ,acquirer->offset +=(-val\_inc - acquirer->value + target->value)

target->enterprise\_value =val\_inc + acquirer->enterprise\_value

בעצם משהו שדומה למה שראינו בשאלת הארגזים בתרגול UF עם עדכון ההפרשים שהתרחש שם.

employeeSalaryIncrease(void \*DS, int EmployeeID, int SalaryIncrease) –

נבדוק שהנתונים שקיבלנו תקינים. אחרת, נחזיר שגיאה מתאימה מסוג INVALID\_INPUT.

נחפש את העובד בעל מזהה EmployeeID ב Hashtable של כלל העובדים במבנה ב O(1) ,ונסמנו emp,ואם לא נמצא אותו אז נחזיר שגיאת FAILURE ונסיים ב O(1) בממוצע.

אם כן מצאנו אותו אז נחפש את החברה אליה הוא שייך ב O(1) כפי שהסברנו למעלה,ונסמנה company\_to\_find, ואז אם יש לו salary > 0 נוציא אותו מעץ הדרגות של העובדים בחברה ובמבנה הכללי ב O(log(n)) ונכניס אותו שוב אחרי שנגדיל את ה salary שלו בערך של salaryincrease לשני העצים שהוצאנו אותו מהם ב O(log(n)).

ואם יש לו salary==0 אז נבצע :

Company\_to\_find->number\_of\_employees\_with\_no\_salary--

Company\_to\_find->sum\_of\_grades\_with\_no\_salary-=emp->Grade

DS->number\_of\_employees\_with\_no\_salary--

DS->sum\_of\_grades\_with\_no\_salary-=emp->Grade

ונכניס אותו לעצי הדרגות של החברה ושל המבנה הכללי ב O(log(n)) (אחרי שהעלינו את ה Salary כמובן).

וכך נסיים בפעולה ב O(log(n)) בממוצע כנדרש!

promoteEmployee(void \*DS, int EmployeeID, int BumpGrade) –

נבדוק שהנתונים שקיבלנו תקינים. אחרת, נחזיר שגיאה מתאימה מסוג INVALID\_INPUT.

נחפש את העובד בעל מזהה EmployeeID ב Hashtable של כלל העובדים במבנה ב O(1) ,ונסמנו emp,ואם לא נמצא אותו אז נחזיר שגיאת FAILURE ונסיים ב O(1) בממוצע.

אם כן מצאנו אותו אז נחפש את החברה אליה הוא שייך ב O(1) כפי שהסברנו למעלה ,ונסמנה company\_to\_find, ואז ואם יש לו salary==0 אז נבצע :

Company\_to\_find->sum\_of\_grades\_with\_no\_salary+=diff

DS->sum\_of\_grades\_with\_no\_salary+=diff

כאשר diff הוא ההפרש בין ה grade החדש לישן.

ואם יש לו salary > 0 נוציא אותו מעץ הדרגות של העובדים בחברה ובמבנה הכללי ב O(log(n)) ונכניס אותו שוב אחרי שנגדיל את ה grade שלו לשני העצים שהוצאנו אותו מהם ב O(log(n)).

וכך נסיים בפעולה ב O(log(n)) בממוצע כנדרש!

sumOfBumpGradeBetweenTopWorkersByGroup (int CompanyID, int m, void \* sumBumpGrade) –

נבדוק שהנתונים שקיבלנו תקינים. אחרת, נחזיר שגיאה מתאימה מסוג INVALID\_INPUT.

אם companyID > 0 אז נמצא את החברה בעלת מזהה זה ב O(log\*(k)) משוערך כפי שנלמד בכיתה ונסמנה ב company\_to\_find ונפעול על העץ של העובדים בחברה באופן המתואר מטה, אחרת נפעול באותו אופן רק עם העץ של כלל העובדים במבנה.

נחפש את העובד בעץ בעל דרגה m ב O(log(n)) ואם הוא לא קיים אז זה סימן שאין מספיק עובדים במבנה שעונים על התנאים ולפיכך נחזיר FAILURE.

אם העובד הזה קיים אז סימן שיש מספיק ועובדים שעונים על התנאים ולכן נחפש העובד הראשון בתת העץ הימני שיש לו פחות מ m עובדים בתת העץ שלו ונחזיר את האב שלו, שנסמנו node, והחיפוש הזה לוקח O(log(n)) כפי שנלמד בכיתה. ואז החל מ node נבדוק כמה איברים יש בתת העץ שלו ונסמנם sons וגם כן נסמן את סכום הדרגות בתת העץ שלו ב sum ונתחיל לרדת בתת העץ השמאלי עד שנגיע לצומת שיש לו מספיק בנים כך שאם נחסיר אותם מ sons נקבל מספר קטן/שווה ל m. נחסיר את הערכים המתאימים של הצומת שהגענו אליו מ sons ו sum.ואז נוסיף את הערכים של תת הצומת הימני שלו ל sons ו sum עד שנגיע ל sons == m (בכל פעם נוסיף את הערגים של תת העץ הימני ונחסיר את של תת העץ השמאלי באופן דומה למה שעשינו למעלה) ונסיים הפעולה כך ב O(log\*(k) + log(n)) משוערך

averageBumpGradeBetweenSalaryByGroup () –

נבדוק שהנתונים שקיבלנו תקינים. אחרת, נחזיר שגיאה מתאימה מסוג INVALID\_INPUT.

נבדוק שהנתונים שקיבלנו תקינים. אחרת, נחזיר שגיאה מתאימה מסוג INVALID\_INPUT.

אם companyID > 0 אז נמצא את החברה בעלת מזהה זה ב O(log\*(k)) משוערך כפי שנלמד בכיתה ונסמנה ב company\_to\_find ונפעול על העובדים של החברה באופן המתואר מטה, אחרת נפעול באותו אופן רק עם העובדים של כלל במבנה.

נמצא את האב קדמון של ...

companyValue(void \*DS, int CompanyID, void \* standing) –

נבדוק שהנתונים שקיבלנו תקינים. אחרת, נחזיר שגיאה מתאימה מסוג INVALID\_INPUT.

נשתמש בפעולת find ב UF כדי למצוא את הקבוצה ששייכת אליה הקבוצה החברה בעלת מזהה CompanyID ב O(log\*(k)) ונסמן את ערכה ב root\_value.

ואז כדי למצוא את הערך של CompanyID נאתחל משתנה offset לאפס והחל מהחוליה של החברה בעץ ההפוך נתחיל לסכום offset += node->offset עד שנגיע לשורש, פעולה זאת לוקחת O(log\*(k)) בדומה ל find ואז אחרי שמצאנו את ה offset הנדרש נבצע כיווץ מסלולים באופן שישמור על ה offset בחיפושים עתידיים ו כדי לשמור על הסיבוכיות להיות O(log\*(k)) כפי ש find עושה.

לבסוף נחזיק real\_value = root\_value – offset ונחזירו כערך החברה המחופשת!

השתמשנו ב find ופעולת חיפוש אחרת שמאוד דומה לה במימוש כפי שראינו בכיתה עם שאלה הארגזים שמתבצעות ב O(log\*(k)) וכך נסיים הפעולה ב O(log\*(k)) כנדרש!

Quit(void \*\*DS) –

bumpGradeToEmployees(void \*DS, int lowerSalary, int higherSalary, int BumpGrade) –

נבדוק שהנתונים שקיבלנו תקינים. אחרת, נחזיר שגיאה מתאימה מסוג INVALID\_INPUT.