דף שער להגשה באיחור

מגיש 1:	דניאל יאשין	ת.ז.	308408673
מגיש 2:	שקד מגדל	ת.ז.	312140411
תרגיל:	רטוב 1		
תאריך הגשה מפורסם:	07.12.2017	שעה:	23:30
תאריך הגשה בפועל:	09.12.2017	שעה:	23:30
מספר ימי איחור*:	1		(*נמדד לפי נהלי הקורס)
מתוכם מוצדקים:	1		

ימי האיחור המוצדקים נובעים מ:

ימי מילואים:	כן
בחני אמצע:	לא

שימו לב:

1) צרפו את האישורים המתאימים (אישור מילואים ו\או צילום כרטיס נבחן). ללא אישורים אלו, לא תאושר הדחייה ויורדו נקודות מציון התרגיל בהתאם.
2) הדחייה בתאריך ההגשה של תרגיל אחד לא משפיעה על תאריך ההגשה של התרגיל הבא. נהלו את זמנכם בהתאם.
3) במקרה של למעלה מ-5 ימי דחייה, יש ליצור קשר עם המתרגל האחראי על התרגיל.

ציון: לפני בונוס הדפסה:

כולל בונוס הדפסה:

נא להחזיר לתא מס':

לכל המעוניין,

הנדון: אישור על תקופות שירות מילואים פעיל בתאריכים 01/10/2017 - 02/12/2017

1. הריני לאשר כי:

מספר אישי	שם משפחה	שם פרטי	תעודת זהות
7555298	יאשין	דניאל	308408673

שירת בשירות מילואים פעיל בתאריכים 01/10/2017 - 02/12/2017 בתקופות הרשומות מטה:

תאריך תחילה	תאריך סיום	סה"כ ימים	הערות	אופן הקריאה לשמ"פ
19/11/2017	23/11/2017	5.0	---	---

2. אישור זה נכון ליום הוצאתו.

3. הצהרת חייל המילואים: אני מצהיר בזאת כי שרתתי שירות מילואים פעיל בתאריכים הרשומים מעלה

מספר אישי	שם משפחה	שם פרטי	תאריך	חתימה
7555298	יאשין	דניאל	2 בדצמבר 2017	

חייל מילואים יקר,
במידה והנך **עובד שכיר**, עליך להגיש את האישור למעסיק בתום שירות המילואים.
במידה והנך **עובד עצמאי** (בלבד) התשלום עבור תקופת המילואים יבוצע אוטומטית ע"י המוסד לביטוח לאומי.אם חלפו שבועיים מיום השירות וטרם קבלת התשלום עליך להגיש תביעה אישית למוסד לביטוח לאומי.
במידה והנך **שכיר ועצמאי** עליך להגיש את האישור למעסיק בתום שירות המילואים ורק לאחר תשלום ההפרש תוכל להגיש תביעה אישית כעצמאי.
סטודנט או מי שאינו עובד – באפשרותך להגיש את האישור באתר האינטרנט של המוסד לביטוח לאומי בכתובת: **www.btl.gov.il** או לסניף הביטוח הלאומי הקרוב לביתך, בדואר או בפקס.
לקבלת מידע באפשרותך לפנות למוקד הטלפוני 02-6463010 או 6050*.

308408673

האישור הופק באתר המילואים עפ"י מספר אסמכתא 11518526

* טופס צבאי זה מוכר לתשלום דמי המילואים באמצעות המוסד לביטוח לאומי *

מחלקות מבנה הנתונים

מבנה הנתונים שלנו יורכב משלוש מחלקות:

א.מחלקת המאמן

המחלקה מייצגת מאמן, לכל מאמן יש מזהה יחודי, $TrainerID$. היא תהיה מורכבת מהבא:

1. $TrainerID$ - מזהה היחודי של המאמן שהיא מספר.
2. N - מספר הגלדיאטורים השייכים למאמן הנ"ל.
3. $SplayTree < LvlNode >$ - פוינטר לעץ המכיל את כל הגלטיאטורים של המאמין הנ"ל, המסודרים על דרגת הגלדיאטורים.
4. $LvlNode$ - $LvlNode$ בודד המחזיק את הגלדיאטור הכי חזק במערכת.

ב.מחלקת גלדיאטור

המחלקה מייצגת גלדיאטור, לכל גלדיאטור יש מזהה יחודי, $GladiatorID$. היא תהיה מורכבת מהבא:

1. $GladiatorID$ - המזהה היחודי של הגלדיאטור.
2. $Level$ - הדרגה של אותו הגלדיאטור.
3. $TrainerID$ - ה־IDשל המאמן.

ג.מחלקת $IDNode$ לגלדיאטור על פי ID

המחלקה תהיה $Node$ בעץ הגלדיאטורים, היא תהיה מורכבת מפוינטר למחלקת גלדיאטור ותשווה אותו על פי $GladiatorID$ בעץ (מול מחלקות אחרות).

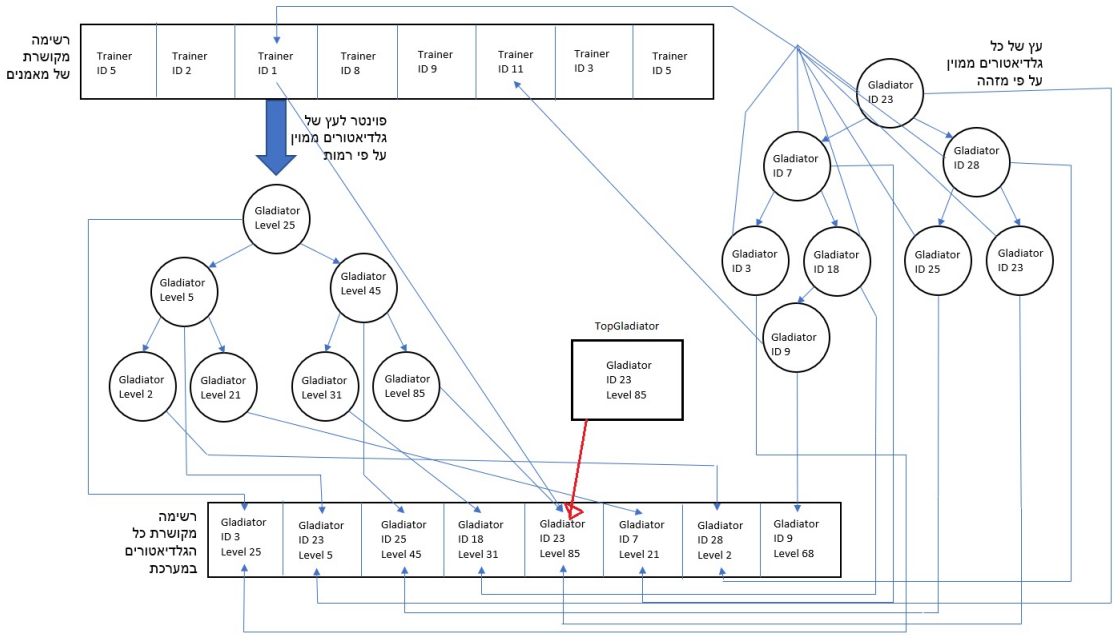
1. $*Gladiator$ - פוינטר למחלקת $Gladiator$ שה־ID מצביע אליה.
2. $*Trainer$ - פויטנר למאמן הרלוונטי של אותו גלדיאטור.

ד.מחלקת $LvlNode$ לגלדיאטור על פי $level$

המחלקה תהיה $Node$ בעץ הגלדיאטורים (תחת המאמן), והיא תהיה מורכבת מפוינטר למחלקת גלדיאטור ותשווה אותו על פי $Level$ בעץ (מול מחלקות אחרות) ועל פי $GladiatorID$.

1. $Gladiator*$ - פוינטר למחלקה של גלדיאטור.

תרשים ופירוט מבנה הנתונים



1. מערכת תהיה מורכבת מעץ כל של פוינטרים של כל הגלדיאטורים על פי ID הכי כל אחד מהם יצביעה למאמן שהוא שייך אליו ול$gladiator$ הרשימה המקושרת אל המאמן.

2. לכל מאמן ברשימה מקושרת היה עץ של $LvlNode$ (שממיון על פי הרמה של הגלדיאטורים) אשר היה פוינטרים ל־$Node$ של הגלדיאטור הרללונטי אליו

3. הרשימה המקושרת של הגלדיאטורים סה"כ תשמור את כל הנתונים על גלדיאטורים.

4. נשמור על $TopGladiator$ אצל כל מאמן כך שיצביע למאמן הכי טוב בעץ הפרטי שלו (לא מופיע בתרשים).

5. נשמור על $TopGladiator$כללי של העץ שיצביע לגלדיאטור החזק במערכת.

6. בנוסף היה עץ של $LvlNode$ של כל הגלדיאטורים על פי הרמה שלהם (כמו העץ שיש לכל $trainer$). (לא מופיע בתרשים)

פירוט מימוש הפונקציות הקשורות למבנה הנתונים

1. $void * int()$ **- מאתחל מבנה נתונים ריק.** המערכת תאתחל:

- עץ של מחלקה 3 (פוינטרים לגלדיאטורים הממיונים על פי ID), $O(1)$.
- עץ של מחלקה 4 ($LvlNode$) שמכילה. $O(1)$.
- רשימה מקושרת ריקה של מאמנים, $O(1)$.
- רשימה מקושרת ריקה של כלל הגלדיאטורים, $O(1)$.
- $LvlNode$ בודד אשר היה $TopGladiator$ של כלל המערכת. $O(1)$.

כיוון שהסיבוכיות הינה $O(1)$ בכל הפעולות, אז הסיבוכיות של הפונקציה הנ"ל תהיה גם היא $O(1)$.

2. $StatusType\ AddTrainer(void * DS, int\, trainerID)$ **- הוספת מאמן חדש עם המזהה** $trainerID$**.**

- המערכת תבדוק ברשימה המקושרת של המאמנים האם המאמן הנ"ל קיים, פעולה המתקיימת בסיבוכיות של $O(k)$ כאשר k הוא מספר המאמנים (סיבוכיות רשימה מקושרת).
- אם לא קיים המאמן עם ה$trainerID$ הנ"ל, המערכת תוסיף את המאמן לרשימה המקושרת של המאמנים, סיבוכיות של $O(1)$ כי ההוספה מתבצעת בתחילת הרשימה.
- המערכת תאתחל עץ ריק של מחלקה 4 ($LvlNode$ של פוינטרים לגלדיאטורים), כאשר העץ ממיון על פי ה-$level$ של אותו גלדיאטור, $O(1)$ אתחול עץ ריק.

כיוון שהסיבוכיות הינה $O(k) + O(1) + O(1)$, הסיבוכיות הכללית תהיה $O(k)$ כנדרש.

3. $StatusType\ BuyGladiator(void* DS, int\, gladiatorID, int\, trainerID, int\, level)$ **- הוספת גלדיאטור חדש למערכת עם מזהה** $GladiatorID$ **השייך למאמן** $trainerID$ **ונמצא רמה** $level$**.**

- המערכת תבדוק את התנאים $DS = NULL$, $gladiatorID \leq 0$, $trainerID \leq 0$ או $level \leq 0$, ואם אחד מהם יתקיים המערכת תחזיר שגיאה ($INVALID\,INPUT$), הפעולה מתבצעת בסיבוכיות של $O(1)$.
- המערכת תעבור על הרשימה המקושרת של המאמנים ותבדוק האם המאמן המדובר ($trainerID$) קיים, אם הוא איננו קיים אז תחזיר שגיאה ($FAILURE$), בפעולה תתבצעה בסיבוכיות של $O(k)$ (כאשר k הוא מספר המאמנים במערכת) כיוון במקרה הכי גרוע (שמאמן איננו קיים) הוא יעבור על כל הרשימה המקושרת.
- המערכת תעבור על העץ של מחלקה 3 (פוינטרים של גלדיאטורים הממיונים על פי ID) של הגלדיאטורים ותבדוק האם הגלדיאטור המדובר ($IDNODE$) איננו קיים במערכת, אם הוא קיים אז המערכת תחזיר שגיעה ($FAILURE$), פעולה זאת תתבצעה בסיבוכיות משוערכת של $O(logn)$ כיוון שמדובר בעץ $splay$.

- אם שלושת השלבים לפני זה עברו בהצלחה בלי לזרוק שום שגיאה, המערכת תיצור גלדיאטור חדש ותכניס אותו לרשימה המקושרת. עם הפוינטר לגלדיאטור הנ"ל היא איבר מהמחלקה 3 ו־4. את האיבר מהמחלקה 3 היא תכניס לעץ $splay$ של כלל הגלדיאטורים (על פי ID), ואת האיבר מהמחלקה 4 היא תחפש את המאמן הרלוונטי לה ברשימה המקושרת של המאמנים ותכניס אותו למאמן הרלוונטי, בנוסף תוסיף את איבר מהמחלקה 4 לעץ כל הגלדאיטורים של $LvlNode$, תבדוק אם הגלדיאטור שנכנס יותר חזק מהגלדיאטור הכי חזק במערכת ותחליף אם כן $O(logn)$. הסיבוכיות לבניית המחלקות תהיה $O(1)$, לחיפוש המאמן תהיה $O(k)$ ולהכנסת כל איבר לעץ שלו $O(logn)$.

לכן הסיבוכיות תהיה $O(log(n)+k)$ משוערכת, כאשר n הוא מספר הכללי של הגלדיאטורים.

4. $StatusType\, FreeGladiator(void *\ DS,\ int\, gladiatorID)$ **- לחופש נולד, שחרור הגלדיאטור בעל המזהה** $gladiatorID$

- המערכת תבדוק קודם כל אם הגלדיאטור קיים, אם לא אז תחזיר $FAILURE$, פעולה תקח $O(logn)$.
- המערכת תמצא את הגלדיאטור המדובר בעץ של הכל הפוינטרים לגלדיאטורים ותוציא ממנו את המידע של איזה מאמן הוא שייך אליו ומה דרגתו, אחרכך תשתמש בפוינטר הזה על מנת להסיר את הגלדיאטור מהרשימה המקושרת שהוא שייך אליה, פעולה זאת תקח סיבוכיות $O(logn)$ משוערכת.
- המערכת תחפש את המאמן ותסיר את הגלדיאטור המדובר מהעץ הקשור למאמן, חיפוש המאמן תקח $O(k)$ והוצאת הפוינטר לגלדיאטור תיקח $O(log(n))$ משוערכת עקב היותו עץ חיפוש בינרי ותעדכן את הגלדיאטור הכי חזק אצל אותו המאמן.
- כעת המערכת תחפש את אותו הגלדיאטור בעץ הגדול של $LvlNode$ שקיים במערכת ותוריד אותו משם $O(logn)$ ואז תעדכן את הגלדיאטור הכי חזק שכעת במערכת אם הגלדיאטור שהוצאנו היה הכי חזק.

הפעולות האלו יקחו $O(logn + k)$ משוערך, כאשר k הוא מספר המאמנים וn הוא מספר הגלדיאטורים.

5. $StatusType\, LevelUp(void\ *\ DS,\ int\, gladiatorID,\ int\, levelincrease)$ **- העלת רמת הלחימה של גלדיאטור בעל המזהה**

- המערכת תבדוק את התנאים $DS == NULL$, $gladiatorID \leq 0$, $levelincrease \leq 0$, אם אחד מהם מתקיים אז תחזיר $INVALID\, INPUT$.
- המערכת תבדוק אם הגלדיאטור קיים, אם לא אז תוציא שגיעת $FAILURE$, יקח $O(logn)$משוערך.
- אם הגלדיאטור קיים, נשתמש ב$IDNode$ שמצאנו על מנת להוציא את המאמן, ונוציא את אותו הגלדיאטור מהעץ של המאמן, $O(logn)$.
- כעת נוציא את אותו הגלדיאטור מהעץ הגדול של ה־$LvlNode$ של המערכת $O(logn)$
- נעדכן את רמת הגלדיאטור דרך $IDNode$ ואז נכניס בחזרה אל העצים שהוצאנו את ה־$LvlNode$, ונעדכן את הגלדיאטור הכי חזק גם אצל המאמן וגם במערכת כולה, $O(logn)$.

הפעולות יקחו $O(Logn)$.

6. $StatusType\,GetTopGladiator(void*\ DS,\ int\,trainerID,\ int*\ gladiatorID)$
- להחזיר את הגלדיאטור בעל הרמה הגבוהה ביותר שבין אלו השייכים ל־$trainerID$

- אם קיבלנו מאמן $trainerID = -1$ אז נחזיר את ה־ID של הגלדיאטור הכי חזק במערכת השמור לנו ב־$LvlNode$ של $TopGladiator$. $O(1)$
- אם קיבלנו $trainerID < -1$ או $trainerID = 0$, נחזיר שגיעה.
- אם קיבלנו $trainerID > 0$ אז נחפש את המאמן, פעולה שתקח $O(k)$ ככמות המאמנים. אם הוא לא קיים אז נחזיר $FAILURE$.
- נחזיר את ה־$TopGladiator$ השמור אצל המאמן. $O(1)$

כל הפעולות יקחו $O(k)$.

7. $StatusType\,GetAllGladiatorsByLevel(void*DS,\ int\,trainerID,\ int**gladiators,\ int*\ numOfGladiators)$
- יש להחזיר את כל הגלדיאטורים של המאמן בעל המזהה $trainerID$ המוינים על סמך הרמה שלהם.

- אם קיבלנו $trainerID < -1$ או $trainerID = 0$, נחזיר שגיעה.
- אם קיבלנו מאמן $trainerID = -1$ אז ניקח את העץ של כלל הגלדיאטורים של $LvlNode$, כיוון שביקשו את כל הגלדיאטור
- אם קיבלנו $trainerID > 0$ אז נחפש את המאמן, פעולה שתקח $O(k)$ ככמות המאמנים. אם לא קיים המאמן, נחזיר $FAILURE$, אחרת ניקח את העץ $LvlNode$ של המאמן.
- נעבור על כל העץ עם $Inorder$ ונספור את כמות הגלדיאטורים, נחזיר אותו.
- נעבור על כל העץ עם $Inorder$ מימין לשמאל (הגדול לקטן) ונשים אותו במערך, פעולה שתקח $O(n)$ ונוציא ככה את הID של הגלדיאטורים לתוך המערך הנתון (נקצה אותו).

הפעולה תקח $O(k+n)$ כאשר k מספר המאמנים ו־n מספר הגלדיאטורים הכללי במערכת (מקרה שבו משתמשים בעץ הגדול) אחרת $n_{trainer}$ כעת מדובר על מאמן מסויים.

8. $StatusType\,UpgradeGladiator(void*DS,\ int\,gladiatorID,\ int\,upgradedID)$

- נבדוק אם $GladiatorID$ קיים ו־$upgradedID$ לא קיים במערכת, אם אחד התנאים לא מתקיים אז נחזיר $FAILURE$.
- נחפש את אותו גלדיאטור בעץ של כל הגלדיאטורים, $O(log(n))$.
- נוציא את האיבר מהעץ $O(log(n))$.
- נעדכן את הגלדיאטור שנמצא בפוינטר (האיבר המקורי שנמצא ברשימה המקושרת). $O(1)$.
- נכניס את האיבר בחזרה, הוא ימצא את מקומו בחזרה והיה מסודר בעץ. $O(log(n))$.

9. $StatusType\,UpdateLevels(void*\,DS,\ int\,stimulantCode,\ int\,stimulantFactor)$

- ניקח את העץ של כל הגלדיאטורים במערכת על פי $LvlNode$ ונפרק אותו למערך עם $InOrder$. $O(n)$
- נפרק אותו לשני מערכים כך שמערך אחד הוא הגלדיאטורים שלא השפעמו והמערך השני הם גלדיאטורים שהושפעו, נחכה עם שני המערכים האלו. $O(n)$
- עכשו נעבור על כל המאמנים $O(k)$ ולכל המאמן נעשה את הפעולות הבאות.
- *) ניקח את העץ ה$LvlNode$ שלו ונפרק אותו אותו למערך עם $InOrder$ $O(n_{trainer})$.
- נפרק אותו לשני מערכים כך שמערך אחד הוא הגלדיאטורים המושפעים והשני לא. $O(n_{trainer})$.
- נכפיל את המושפעים ב־פקטור $O(n_{trainer})$.
- נבצע $Merge$, כיוון שיש לי שני מערכים ממיונים של גלדיאטורים.
- כעת נהפוך את המערך הממיונן הסופי לעץ בינרי על ידי האלגוריתם: נקח את איבר האמצעי והוא היה שורש של העץ, ואז נרכיב רקורסיבית את הצד הימיני על ידי הצד הימיני של המערך, והצד השמאלי של העץ על ידי הצד השמאלי של המערך ונחזור על הפעולה הזאת.
- נבדוק את ה־$Topgladiator$ אצל המאמן.
- נחזור על כל הפעולות מה־$*$ על כל המאמנים.
- הפעולה של כלל המאמנים תיקח לנו $O(k+n)$ כיוון שכמות הגלדיאטורים שנעבור אצל כל הגלדיאטורים הוא n והתבצעו c פעולות על כל אחד. וכיוון שעברנו על כל מאמן בדרך.
- נחזור לעץ הראשי של $LvlNode$ ונעשה לו $Merge$ ונבנה את העץ החדש כמו שתיארנו ב־$trainer$.

כלל הפעולות יקחו $O(k+n)$.

10. $void\,Quit(void\ **DS)$

- מבצעים $delete$ על DS ופלאי ה־$C++$ יעשו את העבודה בשבלינו על ידי:
- מעבר על הרשימה המקושרת של הגלדיאטורים ומחיקת כל נתוני הגלדיאטורים. $O(n)$
- מעבר על הרשימה של המאמנים ומחיקת כל המאמנים. $O(k+n)$, כאשר k הם המאמנים ו־n הם כלל הגלדיאטורים.
- מעבר על העץ של המצביעים לגלדיאטורים ומחיקתם. $O(n)$.
- מעבר על עץ המציביעים של $LvlNode$ ומחיקתם: $O(n)$.
- מחיקת $TopGladiator$.

כל הפעולות יקחו $O(n+k)$.