יבש

מבנה הנתונים הראשי בו עשינו שימוש לחלק השני של התרגיל נקרא DataCenterManager, והוא עושה שימוש במבנה נתונים משני נוסף הנקרא DataCener.

הסבר על $DataCenter$:
מבנה נתונים זה מנהל את המאפיינים הבאים

1. שתי רשימות מקושרת כפולות גנריות של אובייקטים מטיפוס $Server$.
   טיפוס $Server$ מוגדר להיות כשרת פיזי של מכונת $Linux$ או $Windows$.
   רשימה אחת של מכונות $Linux$.
   רישמה אחת של מכונות $Windows$.
2. מערך המכיל את כל השרתים באותו $DataCenter$.

ניתן לראות כי הסיבוכיות מקום של מספר השרתים ב־$DataCenter$ בזמן נתון (כאשר מניחים כי אמורים להיות n שרתים בחווה) היא:
$f(n) = n + n = 2n = O(n)$.

הפונקציונליות אשר $DataCenter$ מספק היא:

- בנייה והריסה.
- התקנת שרתים שונים $Linux$ או $Windows$.
- בקשה של שרת לשימוש.
- בקשה לפינוי שרת.
- קבלת שרת על ידי מזהה.

הסבר על $DataCenterManager$:

מבנה נתונים זה מנהל את המאפיינים הבאים

1. שתי עצי AVL של אובייקטים מטיפוס $DataCenter$:
   עץ המסודר לפי מספר שרתי ה־$Linux$ בכל $DataCenter$.
   ועץ המסודר לפי מספר שרתי ה־$Windows$ בכל $DataCenter$.

(א) הערה: אם באחד העצים קיים שוויון במספר מערכות ההפעלה המותקנות על שני $DataCenter$ שונים העץ ימיין אותם על פי ה־ ID שלהם.

ניתן לראות כי אם נסמן את מספר חוות השרתים בנקודת זמן ב־n ואת החווה עם מספר השרתים הרב ביותר ב־k. אז הסיבוכיות מקום שלנו תיהיה:
$f(n,k) = 2 \cdot k \cdot n = O(k \cdot n)$.

הפונקציונליות אשר $DataCenterManager$ מספק היא:

- בנייה והריסה של המבנה.
- הוספת $DataCenter$.
- הסרת $DataCenter$.
- בקשה של שרת לבצע בו שימוש.
- לפנות שרת משימוש.
- החזרת כל המזהים של חוות השרתים בצורה ממוינת.

אופן מימוש הפעולות הנדרשות:

1. $Init$:
מקצה זיכרון למבנה מטיפוס $DataCenterManager$ ומחזירה מצביע אליו.
סיבוכיות זמן הינה $O(1)$.

2. $AddDataCenter$:
הפעולה יוצרת מבנה חדש מטיפוס $DataCenter$ לפי ID ומספר שרתים אשר הוא מנהל. לאחר מכן מכניסה אותה ל־$DataCenterManager$ המבוקש ושם החווה ממוינת בשני העצים של חוות השרתים. בהנחה שקיימים סה"כ n־חוות שרתים ו־m שרתים בחווה החדשה. נקבל כי מיון בעצים עובד בסיבוכיות זמן של $O(log(n))$ והכנסת m־השרתים לחווה לוקח $O(m)$.
סיבוכיות זמן הינה $O(log(n)+m)$.

3. $DeleteDataCenter$:
הפעולה מחפשת בעץ את השרת ברצונה למחוק ופעולה זו לוקח $O(log(n))$ כאשר יש לנו סה"כ n־חוות שרתים. בנוסף מחיקת כל השרתים בחווה לוקחת $O(m)$ כאשר קיימים m־שרתים בחווה.
סיבוכיות זמן הינה $O(log(n)+m)$.

4. $RequestServer$:
הפעולה מבקשת שרת שתוכל כדי לאייש אותו.
חיפוש חוות השרתים המבוקשת לוקחת $O(log(n))$ כאשר יש n־חוות שרתים.
מציאת השרת המבוקש מתבצע ב־$O(1)$ כי השרתים שמורים בחווה במערך והם ממויינים לפי ה־ID שלהם.
סיבוכיות זמן הינה $O(log(n))$.

5. $FreeServer$:
הפעולה מבקשת לפנות שרת ספציפי.
חיפוש חוות השרתים המבוקשת לוקחת $O(log(n))$ כאשר יש n־חוות שרתים.
מציאת השרת המבוקש מתבצע ב־$O(1)$ כי השרתים שמורים בחווה במערך והם ממויינים לפי ה־ID שלהם.
סיבוכיות זמן הינה $O(log(n))$.

6. $GetDataCentersByOS$:
הפעולה מבקשת לקבל את כל המזהים של חוות השרתים ממוינים בסדר יורד לפי מספר השרתים מסוג OS שיש בחווה.
ובמקרה של שוויון יש למיין מיון משני על פי המזהים בסדר עולה.
$DataCenterManager$ ממיין את חוות השרתים ברגע הכנסתם למערכת לפי דרישות פעולה זו כאשר כל עץ מתאים לOS מבוקשת.
לכן החזרת כל חוות השרתים ממויינים תתבצע ב־$O(n)$ כאשר יש n־חוות שרתים.
סיבוכיות זמן הינה $O(n)$.

7. <u>$Quit$:</u>

הפעולה משחררת את $DataCenterManager$.

שחרור כל השרתים מחווה ספציפית מקיים $O(k)$ כאשר k־שרתים.

שחרור כל חוות השרתים מקיים $O(n)$ כאשר n־חוות שרתים.

סה"כ מספר השרתים הוא $m = n \cdot k$

<u>סיבוכיות זמן</u> הינה $O(n+m)$.

<u>הערה:</u> מתבצע בכל פעולה בדיקת קלט בסיבוכיות זמן של $O(1)$.