4

א. נשמור את כל העצים ב-Rank Tree על בסיס AVL.
המיון במבנה יתבצע לפי עבי העצים. בכל צומת בנוסף לשדות הרגילים נגדיר שדות num לשמירת מספר הצמתים בתת העץ השמאלי של הצומת ו-sum שישמור את סכום הגבהים של כל העצים בתת העץ הנ"ל.

- איתחול המבנה הינו בנית עץ ריק. סיבוכיות $O(1)$.
- insert(height,width)
הכנסה רגילה לעץ AVL. בנוסף יש לעדכן את ערכי השדות num ו-sum: כל פעם שפונים שמאלה בחיפוש המקום להכנסה יש לבצע

$$num \leftarrow num + 1 \qquad sum \leftarrow sum + height$$

סיבוכיות $O(\log n)$.

- delete(width)
מחיקה רגילה מעץ AVL. נשמור במשתנה $height$ את גובהו של העץ הנמחק. לאחר מכן יש לבצע עדכון השדות num ו-sum בכל צומת על המסלול מהשורש ועד למקום המחיקה. העדכון הינו:

$$num \leftarrow num - 1 \qquad sum \leftarrow sum - height$$

סיבוכיות $O(\log n)$.

- average(width1,width2)
נגדיר משתני עזר NUM ו-SUM. נאתחלם לאפס.
נבצע חיפוש במבנה לפי הערך $width2$. תוך כדי החיפוש בכל פניה ימנה מצומת מסויים v יש לבצע:

$$NUM \leftarrow NUM + num(v) + 1 \qquad SUM \leftarrow SUM + sum(v) + height(v)$$

כאשר נגיע לצומת u בעל עבי $width2$, נבצע

$$NUM \leftarrow NUM + num(u) \qquad SUM \leftarrow SUM + sum(u)$$

כעת נבצע חיפוש לפי הערך $width1$. תוך כדי החיפוש בכל פניה ימנה תתבצע הפעולה:

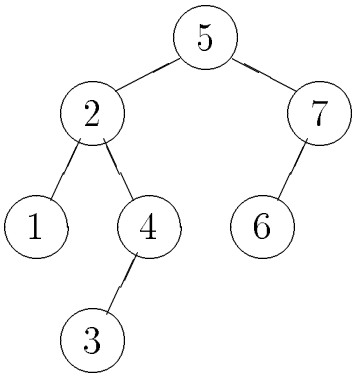
$$NUM \leftarrow NUM - num(v) - 1 \qquad SUM \leftarrow SUM - sum(v) - height(v)$$

בצומת בעל עבי $width1$ הפעולה היא:

$$NUM \leftarrow NUM - num(u) - 1 \qquad SUM \leftarrow SUM - sum(u) - height(u)$$

סיבוכיות $O(\log n)$.

ב. העץ המבוקש המינימלי יכיל שבעה צמתים:



ביצוע delete(1) בעץ הזה גורם לגלגול RL,
ואילו delete(6) גורם לגלגול LR.

ג. האלגוריתם לאיחוד שני עצי AVL הינו:

(1) מצא את הערך המקסימלי ב-T_1 ע"י ירידה ימינה בעץ החל מהשורש ועד הצומת שאין לו בן ימני. סמן את הצומת ב-v.

(2) מחק את v מ-T_1 ושמור אותו "בצד".

(3) סמן את גובהו של העץ T_1 ב-h_1, וגובהו של T_2 ב-h_2.

- אם מתקיים: $|h_1 - h_2| \leq 1$, אזי חבר את T_1 כתת עץ שמאלי של v ואת T_2 כתת עץ ימני של v. לאחר מכן סיים.
  בזה מתקבל עץ AVL חוקי.
- אם $|h_1 - h_2| > 1$, נניח בה"כ כי $h_1 > h_2$. נסמן את גובהו של העץ הקטן ב-h (כלומר, $h = h_2$).

(4) כעת, רד בעץ T_1 במסלול הימני ביותר (כמו שעשית ב-(1)) עד הצומת (נסמנו ב-u) אשר תת העץ הימני שלו בעל בגובה h.

(5) נתק את תת העץ הימני של u. חבר אותו כתת עץ שמאלי של v ואת T_2 כתת עץ ימני של v.
בזה מתקבל עץ AVL חוקי T_{sub} בעל גובה $h+1$.

(6) חבר את v כבן ימני של u.
בזה העץ T_{sub} מתחבר אל הצומת u צתת העץ הימני שלו.

(7) כעת עדכן (הקטן ב-1) את BF של u, כי לפני הפעולה היה לו תת עץ ימני בגובה h, ולאחר האיחוד תת העץ הימני שלו הינו בעל גובה $h+1$.

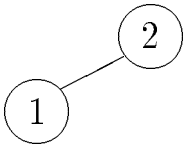
(8) בצע את התיקונים הרגילים בעץ AVL החל מ-u כלפי מעלה. התיקונים הינם בדיוק כמו בפעולת הכנסת צומת בודד לעץ AVL! זה נכון, מפני שתוספת גובה הנגרמת ע"י הפעולה הינה 1 בלבד, בדיוק כמו בהכנסה רגילה.

סיבוכיות האלגוריתם מורכבת מ-:

- מציאת המקסימום v ב-T_1 ומחיקתו: $O(\log n)$.
- מציאת גובה של כל עץ: $O(\log n)$.
- מציאת צומת u: $O(\log n)$.
- חיבורים בין הצמתים: $O(1)$.

סה"כ הסיבוכיות היא $O(\log n)$.

(3) **נכון.** דוגמה:



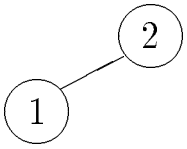
סיור postorder על העץ הבא מחזיר סדרה ממויינת של אברים, והעץ הינו עץ חיפוש.

(4) **לא נכון.** דוגמה נגדית:

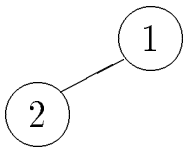
זאת הצורה של העצים A, B ו-C על המפתח $\{1\}$:

(5) **לא נכון.** דוגמה נגדית של הסעיף הקודם מתאימה גם פה.

(6) **נכון.** דוגמה:

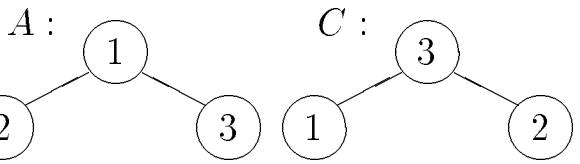


זאת הצורה של העצים B ו-C זהים על המפתחות $\{1,2\}$:



וזהו עץ A המתאים:

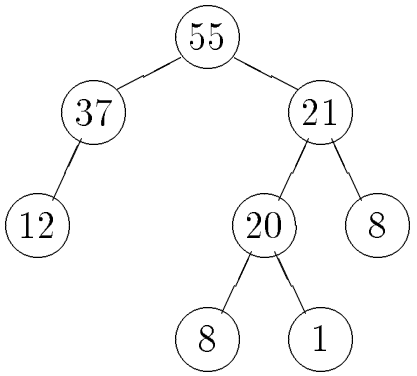
(7) **לא נכון.** דוגמה נגדית:

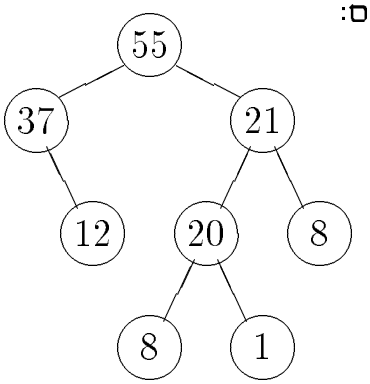


אלה עצים A ו-C על המפתחות $\{1,2,3\}$:

סיור postorder ב-A נותן סדרה $2,3,1$, לעומת זאת סיור preorder ב-C מחזיר סדרה $3,1,2$.

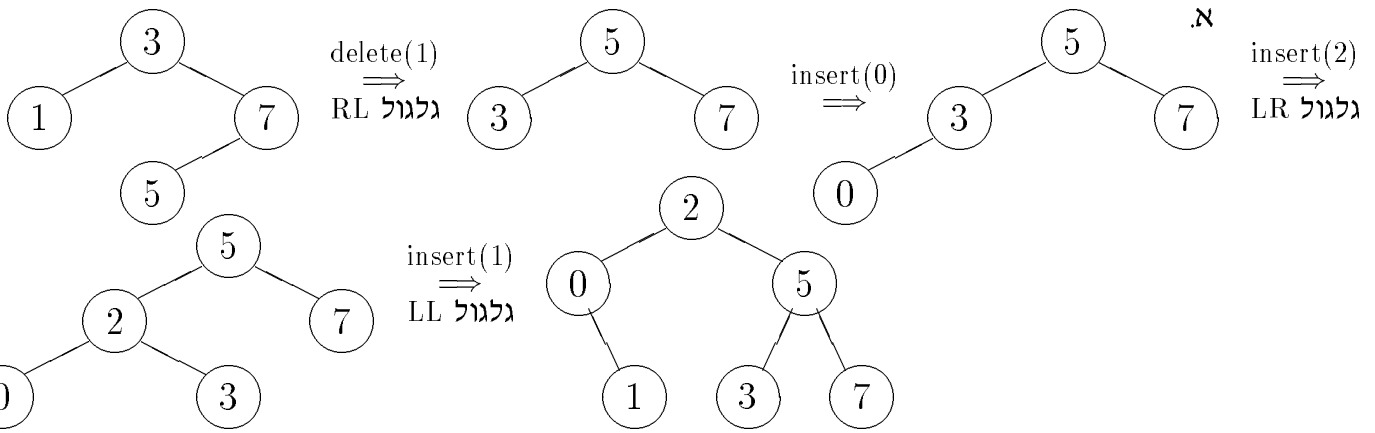
ב. אלה העצים המתקבלים:





3

א.



(1) בדוק האם $head$ נמצא על המעגל.
- אם כן, אזי $head$ הינו הצומת הראשון על המעגל. החזר אותו וסיים.
- אחרת, המשך.

(2) הגדר $first \leftarrow head$ ו-$last \leftarrow circ$.

(3) מצא את המרחק (מספר הצמתים) בין $first$ ל-$last$ ע"י מעבר מ-$first$ ל-$last$ וספירת הצמתים בדרך. סמן את המרחק ב-d.

(4) בדוק האם $d = 0$, כלומר האם $first$ ו-$last$ הם שכנים.
- אם כן, אזי $last$ הינו הצומת הראשון על המעגל. החזר אותו וסיים.
- אחרת, המשך.

(5) עבור לצומת הנמצא באמצע הדרך בין $first$ ל-$last$ ע"י מעבר על $\left\lceil \frac{d}{2} \right\rceil$ צמתים החל מ-$first$. סמן את הצומת אליו הגעת ב-$current$.

(6) בדוק האם $current$ נמצא על המעגל.
- אם כן, בצע $last \leftarrow current$.
- אחרת, בצע $first \leftarrow current$.

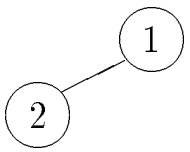
(7) חזור ל-(3).

הלולאה באלגוריתם הנ"ל מתבצעת $O(\log n)$ פעמים, כי בתחילת ריצת האלגוריתם אורך הקטע בו נערך החיפוש הינו $O(n)$, ובכל איטרציה הקטע מתקצר פי 2.
בכל איטרציה מבצעים בדיקת הימצאות צומת על המעגל ($O(n)$) ומציאת אמצע הקטע (גם $O(n)$).
מכאן סה"כ סיבוכיות האלגוריתם הינה $O(n \log n)$.

2

א.

(1) לא נכון. דוגמה נגדית:



סיור preorder על העץ הבא מחזיר סדרה ממויינת של אברין, אך העץ אינו עץ חיפוש.

(2) נכון.

הוכחה באינדוקציה על גובה העץ :

בסיס: עבור עץ בגובה 0 הטענה טריוויאלית, כי עץ בעל צומת בודד מקיים את התכונה של עץ חיפוש בינארי באופן ריק.

הנחה: נניח נכונות הטענה עבור עץ בגובה k.

צעד: נוכיח עבור עץ בגובה $k+1$.

סיור inorder בעץ בינארי הינו סיור inorder בתת העץ השמאלי של השורש, טיפול בשורש, ואז סיור inorder בתת העץ הימני.
מכיוון שהסיור מחזיר סדרה ממויינת של צמתים, אזי סיור inorder בכל אחד מתת העצים מחזיר סדרה ממויינת, והערך הנמצא בשורש גדול מכל אלה שבתת העץ השמאלי וקטן מאברי תת העץ הימני. לכן שורש העץ מקיים את התכונה של עץ חיפוש בינארי. וכל אחד מתת העצים מקיים את התכונה ע"ס הנחת האינדוקציה, כי גובה של כל אחד מהם הינו k לכל היותר.

פתרון תרגיל בית יבש מס׳ 2

1

א. נקרא לשדה הנוסף $used$.
האלגוריתם למציאת הצומת הראשון על המעגל תוך שימוש בשדה $used$:

(1) עבור על הרשימה ואתחל את השדות $used$ בכל הצמתים.
כלומר, כל עוד לא נתקלת <u>פעם שניה</u> בצומת $circ$,
עבור לצומת הבא ובצע בו $used \leftarrow FALSE$.

(2) כעת עבור על הרשימה שנית החל מהצומת הראשון.
בכל צומת בדוק האם $used = TRUE$.

- אם כן, זהו הצומת הראשון על המעגל. החזר מצביע אליו וסיים.
- אם לא, בצע $used \leftarrow TRUE$ והמשך.

סיבוכיות האלגוריתם:

- סיבוכיות האתחול הינה: $2n = \mathrm{O}(n)$.
- סיבוכיות המעבר השני: $n = \mathrm{O}(n)$.

סה"כ $n = \mathrm{O}(n)$.

ב. נשתמש במשתנה עזר $count$ המאותחל ל-1.
האלגוריתם למציאת אורך המעגל:

(1) עבור על הרשימה החל מראשה ועד לצומת $circ$.

(2) כעת המשך מעבר על הרשימה ובכל צומת בדוק האם הינו $circ$

- אם כן, החזר את ערכו של $count$ וסיים.
- אם לא, בצע $count \leftarrow count + 1$ והמשך.

במהלך ביצוע האלגוריתם עוברים על כל צומת פעמיים לכל היותר, לכן הסיבוכיות היא $2n = \mathrm{O}(n)$.

ג. כדי לבדוק האם צומת כלשהו v נמצא על המעגל יש לעבור על הרשימה החל מהצומת ועד שיקרה אחד המאורעות הבאים:

- אם בקרנו שנית ב-v, אזי הצומת הינו על המעגל.
- אם עברנו פעמיים בצומת שידוע עליו שהינו על המעגל (למשל צומת $circ$), פרושו הדבר שעשינו מעגל שלם ולא נתקלנו ב-v (אחרת היה קורה המאורע הראשון).
כלומר, v אינו על המעגל.

סיבוכיות הבדיקה היא $\mathrm{O}(n)$.

הצומת הראשון על המעגל פרושו האיבר ברשימה שכל האיברים הקודמים לו אינם על המעגל וכל אלה שאחריו (כולל עצמו) הם כן במעגל.
לכן הצומת המבוקש בטוח נמצא בין $head$ ל-$circ$.
למציאתו נשתמש ברעיון של חיפוש בינארי: