מטלה 6 - מבני נתונים

שאלה 1 -

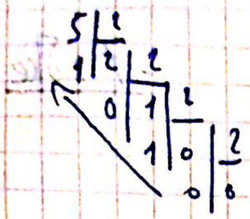
```
int height (Node n)
    if (n == null) return 0
    int right = height (n.getRight())
    int left = height (n.getLeft())
    return 1 + Math.max (right, left)
end- height.

int size (Node n)
    if (n == null) return 0
    int right = size (n.getRight())
    int left = size (n.getLeft())
    return 1 + right + left.
end- size

boolean complete (Node n)
    int nodes = size (n)
    int h = height (n)
    if (nodes == Math.pow (2, h) - 1) return true
    return false.
end- complete
```

שאלה 2-

א) יש ב-A 5 איברים.



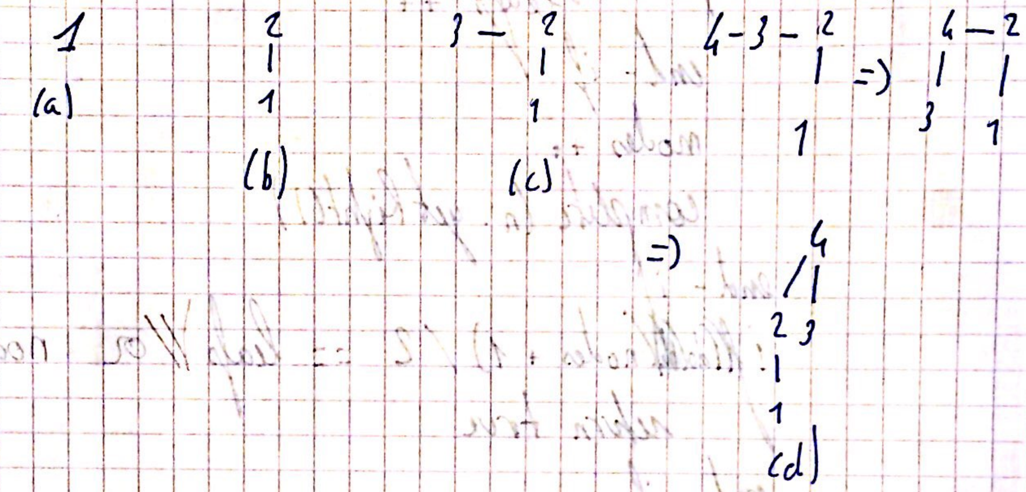
$5_{10} = 101_2$

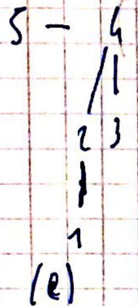
$5_{10} = 2^0 + 2^2$

ולכן, מקבלים ערמה כזאת:

A היא ערמה שמתקבלת אם נכניס את האיברים בסדר עולה, ולפי דרך מוסיפים איברים:

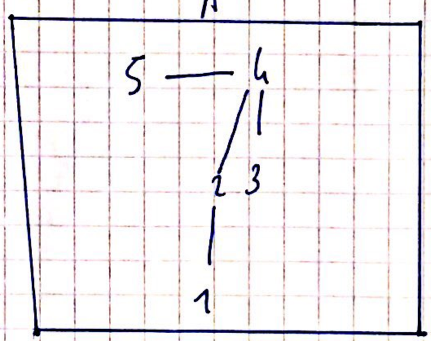
1,2,3,4,5 :

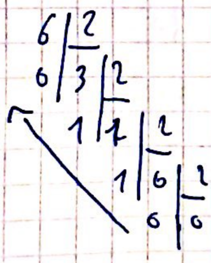




קיבלנו ערמה בינומית:

A





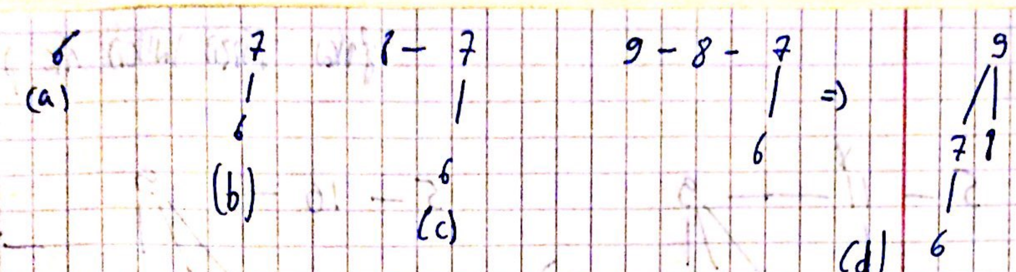
יש ב-B 6 איברים

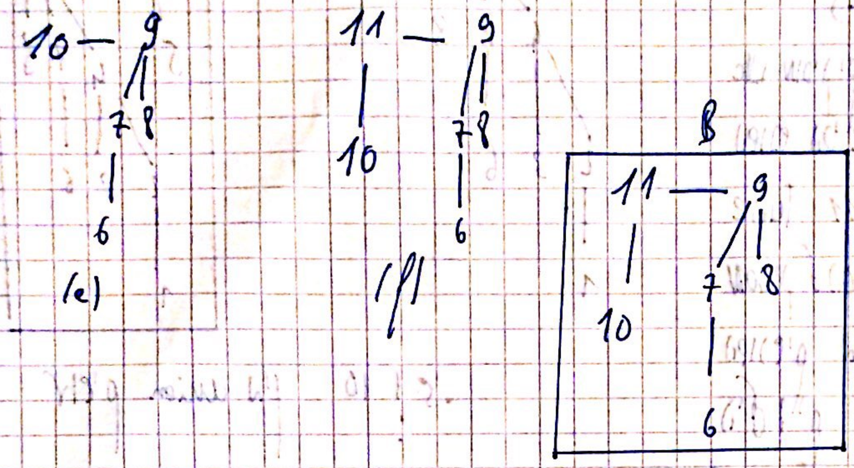
$6_{10} = 110$
$= 2^1 + 2^2$

ולכן מקבלים ערמה כזאת:

הערמה שמורה על maxHeapify כמו הערמה A:

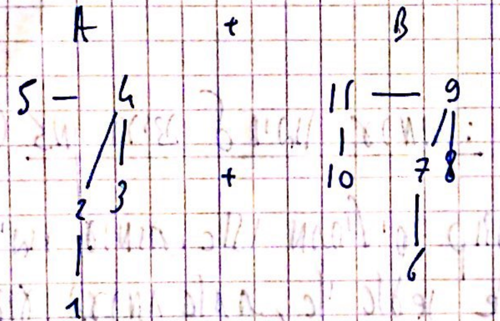
6, 7, 8, 9, 10, 11:

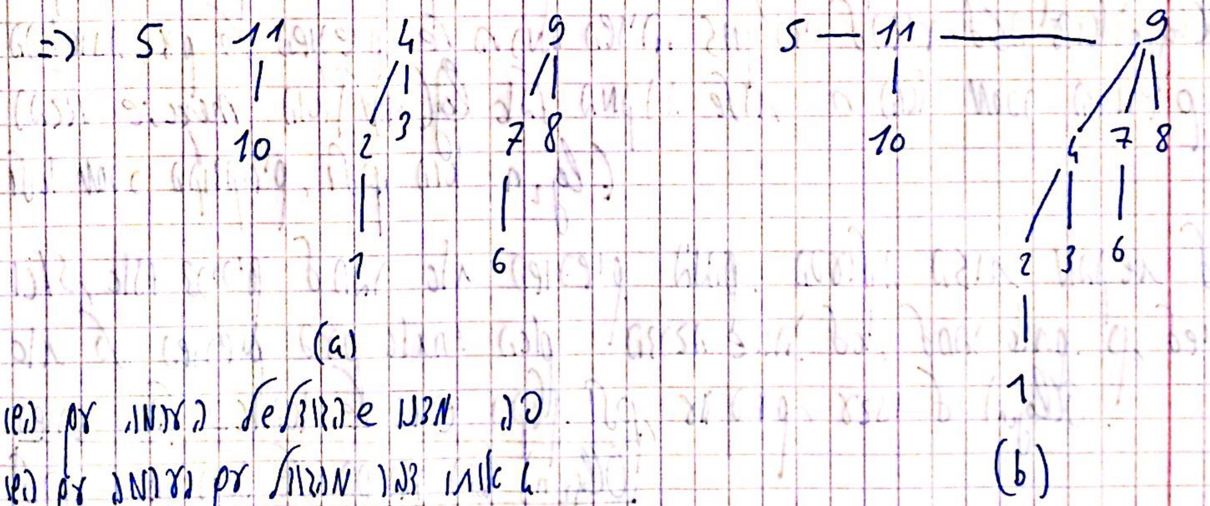




קיבלנו ערמה בינומית:

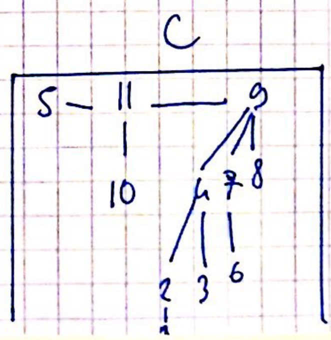
ב



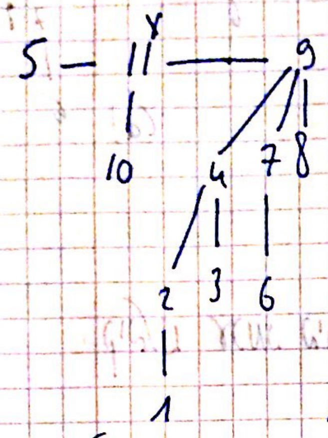


פה מכאן שהגודל הערמה, עם השורש
4 אותו דבר מהגודל עם ערמה עם השורש
5, ולכן, מכניסים השורש הקטן היותר
כילד של השורש הגדול היותר.

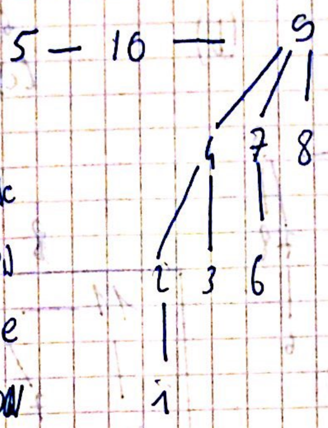
לכן, קיבלנו

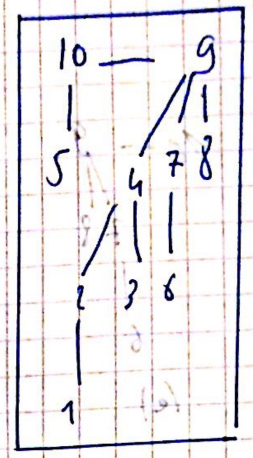


5) אנו רוצים למחוק את המינימום הגדול מאוחד



אנו מוסיפים
משורש הכי גדול
שהוא 11,
ממשיכים להסתכל על
השורשים בגלל שהערמה מסודרת על Max Heapify
הילדים של השורש ממלאים כערמות
עצמאיות.





עושים union בין 10 ל-5.

7) **זמן הריצה לחיבור ערמות:**

בחיבור ערמות, אנו מתחילים קודם כל למיין את השורשים לפי הדרגה שלהם.
כידוע, בערמה אחת, אי אפשר ששני שורשים יהיו באותו דרגה.
ולכן, כדי למיין, אנחנו כן צריכים לעשות merge בין כל השורשים של הערמה
הראשונה, עם כל השורשים של הערמה השנייה. זמן כדי להגיע לזה יהיה $O(\log_2 n)$
(בגלל שה-merge $O(n)$ אבל במקרה שלנו, n הוא מספר השורשים,
ולא מספר הקודקודים, ולכן, הוא $\log_2 n$)

ואז, אנו צריכים לחבר את השורשים ממוינים. הפעולה הזאת מבקשת לבדוק
את כל השורשים פעם אחת: האם הדרגה שווה? אם כן, לחבר אותם. כן, לשים
השורש הגדול כילד של השורש הקטן. ולכן, גם פה, עדיין $O(\log_2 n)$!

$\boxed{\text{ולכן, הזמן ריצה של החיבור הוא } O(\log_2 n)}$

זמן ריצה של המחיקת איבר:

(כמו נתון)
במחיקת איבר, אנחנו צריכים לבחור איבר האיבר, ולשנות אותו כ-∞.
זה $O(1)$.

לאחר מכן, אנחנו צריכים להעלות את האיבר ∞ עד השורש (שימוש של max heapify), זמן ריצה הוא $O(\log_2 n)$. ($\log_2 n$ הוא מספר של הרמות של הערימה)

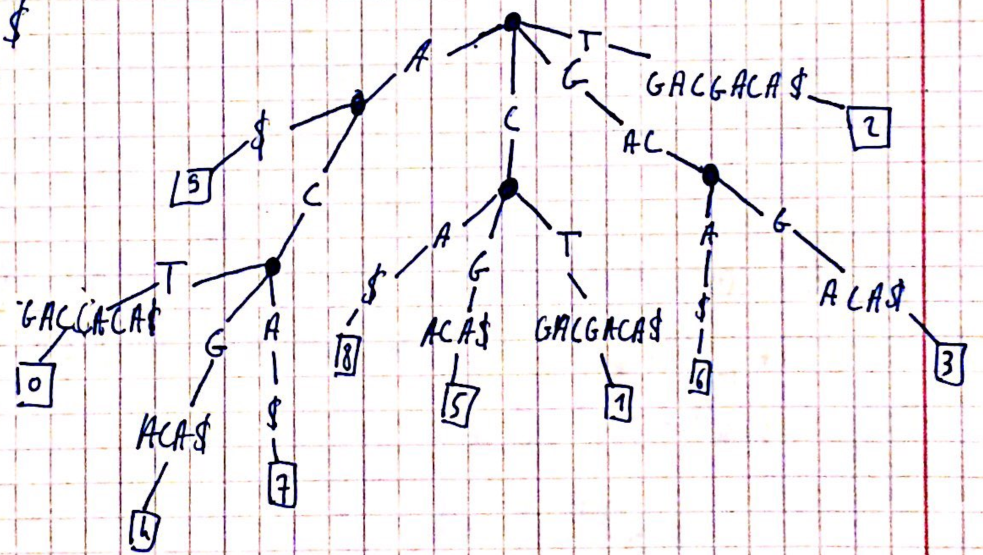
כאשר הוא השורש, מוחקים אותו, וכל הילדים שלו מתלכדים כערמות עצמאיות. ואז, הזמן הוא $O(\log_2 n)$ (בגלל שיש לשורש לכל היותר $\log_2 n$ ילדים).

ולכן, אנו צריכים לחבר כל הערמות חדשות והזמן הוא $\log_2(n)$ כמו הוכחנו.

$\boxed{\text{לכן, זמן ריצה של מחיקת איבר } O(\log_2 n)}$

שאלה 3 -

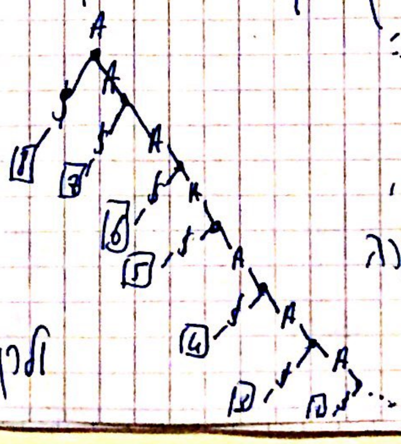
0 - ACTGACGACA$
1 - CTGACGACA$
2 - TGACGACA$
3 - GACGACA$
4 - ACGACA$
5 - CGACA$
6 - GACA$
7 - ACA$
8 - CA$
9 - A$



הזמן ריצה למיפוי את מחרוזת מילה לתווים הוא $O(5\cdot 26)$ בגלל שבמקרה הכי גרוע, המיפוי הוא מחזיק רק אותי במילון שבו ה-5 תווים ושמשמשים אותם בקלט, כגון עץ סיומות:

בת אנגליות

0 - AAAAAAA$
1 - AAAAAA$
2 - AAAAA$
5 - AAAA$
6 - AAA$
7 - AA$
8 - A$

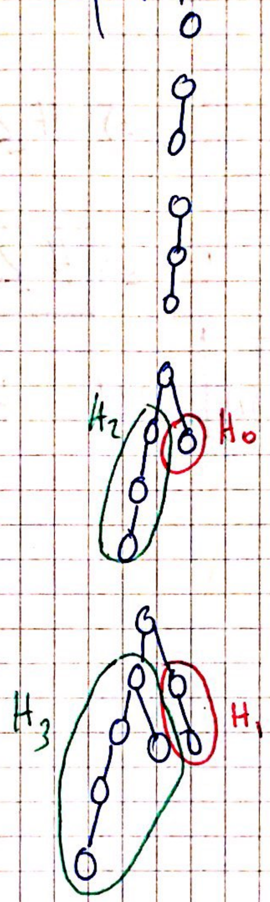


כדי להגיע ל "AAAAA",
אני צריך 5 פעולות במקרה
הגרוע, שהוא מקרה הכי גרוע.

ולכן, $\boxed{O(5\cdot 26) = O(1)}$ (במספר סופי של תווים).

שאלה ה- הוכחה:

מהו מספר מינימאלי של קודקודים לעץ AVL שגובהו H.



כאשר $H=0$, $N=1$

$H=1$, $N=2$

$H=2$, $N=3$

$H=3$, $N=5$

$H=4$, $N=8$

N_H הוא מספר מינימאלי של קודקודים בעץ AVL, $N_H = N_{H-3} + N_{H-1} + 1$

ברור ש $N_{H-3} < N_{H-1} + 1$ כי הגדרה היא סדרה מונוטונית עולה.

לכן, $N_H \geq N_{H-3} + N_{H-3}$

$N_H \geq 2N_{H-3}$.

נפתור משוואת $G_H = x^n$, ונניח $G_H = 2G_{H-3}$, $G_0 = 1$

קיבלנו $x^n = 2x^{n-3}$

$\Rightarrow x^3 = 2$

$\Rightarrow x = \sqrt[3]{2}$

לכן, $G_H = a\sqrt[3]{2}^H$

$G_0 = 1 = a$, לכן, $\boxed{G_H = \sqrt[3]{2}^H}$

מסוון: $G_H = \sqrt[3]{2}^H \geq n$ כי אם ש $\sqrt[3]{3}^H$ והוא

ולכן,

$$n \geq \sqrt[3]{2}^{H}$$

$$\Rightarrow \log_{\sqrt[3]{2}} n \geq H$$

$$\Rightarrow H \leq \frac{\log_2 n}{\log_2 \sqrt[3]{2}}$$

$$\Rightarrow H \leq \frac{\log_2 n}{\frac{1}{3}}$$

$$\Rightarrow \boxed{H \leq 3 \log_2 n}$$

קיבלנו ש: $H = O(\log_2 n)$, ולכן, $\boxed{\text{העץ [illegible] הוא כן מאוזן.}}$