מבני נתונים – עבודה מספר 1

I, (**Niv Dan -**), assert that the work I submitted is entirely my own. I have not received any part from any other student in the class, nor did I give parts of it for use to others. I realize that if my work is found to contain code that is not originally my own, a formal case will be opened against me with the BGU disciplinary committee.

#1

נוכיח בעזרת גבולות את הסדר האסימפטומטי של הפונקציות הנתונות:

$$f(n) = O(g(n)) \ \ if \ \ \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$$

$$f(n) = \theta(g(n)) \ \ if \ \ \lim_{n\to\infty} \frac{f(n)}{g(n)} = a, \ 0 < a < \infty$$

נשים לב כי היחס O הוא יחס טרנזיטיבי ולכן נסתפק בלהוכיח את היחס אך ורק בין כל שתי פונקציות עוקבות ביחס:

$$\lim_{n\to\infty} \frac{f_4(n)}{f_1(n)} = \lim_{n\to\infty} \frac{\frac{1}{\sqrt{n}}}{2020} = \lim_{n\to\infty} \frac{1}{2020\cdot\sqrt{n}} = 0$$

$$\lim_{n\to\infty} \frac{f_1(n)}{f_{13}(n)} = \lim_{n\to\infty} \frac{2020}{2^{100}} = 0$$

$$\lim_{n\to\infty} \frac{f_{13}(n)}{f_9(n)} = \lim_{n\to\infty} \frac{2^{100}}{\log\left(n^{\frac{2}{5}}\right)} = \lim_{n\to\infty} \frac{2^{100}}{\frac{2}{5}\cdot\log(n)} = 0$$

$$\lim_{n\to\infty} \frac{f_9(n)}{f_{11}(n)} = \lim_{n\to\infty} \frac{\log\left(n^{\frac{2}{5}}\right)}{\log(n^5)} = \lim_{n\to\infty} \frac{\frac{2}{5}\cdot\log(n)}{5\cdot\log(n)} = \frac{2}{25}$$

$$\lim_{n\to\infty} \frac{f_{11}(n)}{f_{10}(n)} = \lim_{n\to\infty} \frac{\log(n^5)}{\log_3\left(3^n\cdot n^2\right)} = \lim_{n\to\infty} \frac{5\cdot\log(n)}{\log_3(3^n) + \log_3(n^2)} = \frac{5\cdot\log(n)}{n + 2\cdot\log_3(n)} = 0$$

$$\lim_{n\to\infty} \frac{f_{10}(n)}{f_{14}(n)} = \lim_{n\to\infty} \frac{\log_3\left(3^n\cdot n^2\right)}{\frac{4n}{3}} = \lim_{n\to\infty} \frac{3\cdot\log_3\left(3^n\cdot n^2\right)}{4n} = loop\ \frac{\infty}{\infty} = \lim_{n\to\infty} \frac{3 + \frac{2}{n\cdot\ln(3)}}{4} = \frac{3}{4}$$

$$\lim_{n\to\infty} \frac{f_{14}(n)}{f_2(n)} = \lim_{n\to\infty} \frac{\frac{4n}{3}}{3^{\log_{\sqrt{3}}(n)}} = \lim_{n\to\infty} \frac{\frac{4n}{3}}{n^{\log_{\sqrt{3}}(3)}} = \lim_{n\to\infty} \frac{\frac{4n}{3}}{n^2} = \frac{4n}{3n^2} = 0$$

$$\lim_{n\to\infty} \frac{f_2(n)}{f_{12}(n)} = \lim_{n\to\infty} \frac{3^{\log_{\sqrt{3}}(n)}}{n^3 + n^2 + \log(n) + n} = \lim_{n\to\infty} \frac{n^2}{n^3 + n^2 + \log(n) + n} = 0$$

$$\lim_{n\to\infty} \frac{f_{12}(n)}{f_3(n)} = \lim_{n\to\infty} \frac{n^3 + n^2 + \log(n) + n}{2\sqrt{n}} = loop\ \frac{\infty}{\infty} = 0$$

$$\lim_{n\to\infty} \frac{f_3(n)}{f_5(n)} = \lim_{n\to\infty} \frac{2\sqrt{n}}{5^n} = 0$$

$$\lim_{n\to\infty} \frac{f_5(n)}{f_7(n)} = \lim_{n\to\infty} \frac{5^n}{n^n} = \lim_{n\to\infty} \left(\frac{5}{n}\right)^n = \lim_{n\to\infty} e^{n\ln\left(\frac{5}{n}\right)} = e^{\lim_{n\to\infty} n\ln\left(\frac{5}{n}\right)} = 0$$

$$\lim_{n\to\infty} \frac{f_7(n)}{f_8(n)} = \lim_{n\to\infty} \frac{n^n}{3^{2^n}} = \lim_{n\to\infty} \frac{e^{n\ln(n)}}{e^{2^n\ln(3)}} = \lim_{n\to\infty} \frac{n\ln(n)}{2^n\ln(3)} = loop\ \frac{\infty}{\infty} = \lim_{n\to\infty} \frac{\ln(n)+1}{2^n\ln(2)\ln(3)} =$$

$$loop\ \frac{\infty}{\infty} = \lim_{n\to\infty} \frac{1}{n2^n\ln^2(2)\ln(3)} = 0$$

$$\lim_{n\to\infty} \frac{f_8(n)}{f_6(n)} = \lim_{n\to\infty} \frac{3^{2^n}}{2^{3^n}} = \lim_{n\to\infty} \frac{e^{2^n\ln(3)}}{e^{3^n\ln(2)}} = \frac{\ln(3)}{\ln(2)} \lim_{n\to\infty} \frac{2^n}{3^n} = 0$$

$$\frac{1}{\sqrt{n}} \le 2020 \le 2^{100} \le \log n^{\frac{2}{5}} \le \log n^5 \le \log_3(3^n \cdot n^2) \le \frac{4}{3}n \le 3^{\log_{\sqrt{3}} n}$$
$$\le n^3 + n^2 + \log n + n \ \le 2^{\sqrt{n}} \le 5^n \le n^n \le 3^{2^n} \le 2^{3^n}$$

נראה הוכחה פורמלית בין 2 פונקציות הדוקות :

- $\mathbf{2020 = \Theta(2^{100})}$: $\exists_{c_1=c_2=\frac{2020}{2^{100}}} (c_1 \cdot 2^{100} \le 2020 \le c_2 \cdot 2^{100})$, $\forall_{n\ge n_0=\text{כל } n}$

- $\mathbf{\frac{2}{5}\log n = \Theta(5\log n)}$: $\exists_{c_1=\frac{1}{15}, c_2=1} \left(c_1 \cdot 5\log n \le \frac{2}{5}\log n \le c_2 \cdot 5\log n\right)$, $\forall_{n\ge n_0=\text{כל } n}$

- $\mathbf{\log_3(3^n \cdot n^2) = \Theta\left(\frac{4}{3}n\right)}$: $\exists_{c_1=1, c_2=\frac{4}{3}} \left(c_1 \cdot \frac{4}{3}n \le \log_3(3^n \cdot n^2) \le c_2 \cdot \frac{4}{3}n\right)$

$$\left(c_1 \cdot \frac{4}{3}n \le n + 2\log_3 n \le c_2 \cdot \frac{4}{3}n\right)$$

$$\left(c_1 \le \frac{3}{4} + \frac{3\log_3 n}{2n} \le c_2\right), \forall_{n\ge n_0=2}$$

#2

א. הוכיחו כי: $f(n) = O(g(n))$ וגם $f(n) = \Omega(g(n)) \Leftrightarrow f(n) = \Theta(g(n))$

<u>הוכחה :</u>

יהיו $f(n), g(n)$ פונקציות.

$\Leftarrow$ נניח $f(n) = O(g(n))$ וגם $f(n) = \Omega(g(n))$.

צ"ל $f(n) = \Theta g(n)$,

כלומר צריך להראות $\exists_{c_1,c_2}: 0 \le c_2 \cdot g(n) \le f(n) \le c_1 \cdot g(n), \forall_{n>n_0}$.

מההנחה נובע כי:

$f(n) = O(g(n))$ אז $\exists_{d_1} : 0 \le f(n) \le d_1 \cdot g(n)$

$f(n) = \Omega(g(n))$ אז $\exists_{d_2} : 0 \le d_2 \cdot g(n) \le f(n)$

לכן נגדיר $c_1 = d_1, c_2 = d_2$. נציב ונקבל :

$$0 \le c_2 \cdot g(n) \le f(n) \le c_1 \cdot g(n)$$

לכן $f(n) = \Theta(g(n)$.

$\Rightarrow$ נניח $f(n) = \Theta(g(n)$, צריך להוכיח כי $f(n) = O\big(g(n)\big)$ וגם

$f(n) = \Omega\big(g(n)\big)$ כלומר צריך להראות : $\exists_{c_1} : 0 \le f(n) \le c_1 \cdot g(n)$

וגם $\exists_{c_2} : 0 \le c_2 \cdot g(n) \le f(n)$. מההנחה נובע :

$$\exists_{d_1,d_2}: 0 \le d_2 \cdot g(n) \le f(n) \le d_1 \cdot g(n), \forall_{n>n_0}$$

נגדיר $c_1 = d_1, c_2 = d_2$, נציב ונקבל :

$$0 \le c_2 \cdot g(n) \le f(n) \le c_1 \cdot g(n)$$

נפצל את האי שיוון ונקבל :

$0 \le f(n) \le c_1 \cdot g(n)$ לכן לפי הגדרת O , $f(n) = O(g(n))$

$0 \le c_2 \cdot g(n) \le f(n)$ לכן לפי הגדרת Ω , $f(n) = \Omega(g(n))$

■

ב. הוכיחו כי : אם $f(n) = O(g(n))$ אז $g(n) = \Omega(f(n))$

__הוכחה:__

נניח $f(n) = O(g(n))$, צ״ל : $\forall_{n>n_0} : \exists_{c_1} : 0 \le c_1 \cdot f(n) \le g(n)$
מההנחה נובע כי :

$\forall_{n>n_0}$, $\exists_{d_1} : 0 \le f(n) \le d_1 \cdot g(n)$

לכן $/d_1$ $\quad 0 \le f(n) \le d_1 \cdot g(n)$

$0 \le \frac{1}{d_1} \cdot f(n) \le g(n)$,

נגדיר : $c_1 = \frac{1}{d_1}$

$0 \le c_1 \cdot f(n) \le g(n)$

לכן $g(n) = \Omega(f(n))$

■

ג. יהיו $p_1(n), p_2(n)$ פולינומים מחזקה n_1, n_2 בהתאמה.
נסמן: $p_1(p_2(n)) = \Theta(g(n))$. צ״ל : $g(n)$.
מהנתון : $0 \le c_1 \cdot g(n) \le p_1\big(p_2(n)\big) \le c_2 \cdot g(n)$
לפי משפט שנלמד בכיתה : כל פולינום מדרגה k : $p(n) = \Theta(n^k)$
לכן , $p_1(n) = \Theta(n^{n_1})$, $p_2(n) = \Theta(n^{n_2})$
אז $p_1(p_2(n)) = p_1(\Theta(n^{n_2})) = (n^{n_2})^{n_1} = \Theta(n^{n_1 \cdot n_2})$
$p_1(p_2(n)) = \Theta(g(n))$ אז $g(n) = n^{n_1 \cdot n_2}$.

ד. ננתח לפי שורות :

מקרה הגרוע :

Line	Cost	Times
1	c_1	1
2	c_2	*n*
3	c_3	*n-1*
4	c_4	1
5	c_5	1
6	c_6	1

$$c_1 + c_2 \cdot \mathrm{n} + c_3 \cdot (\mathrm{n} - 1) + c_4 + c_5 = (c_1 - c_3 + c_4 + c_5) + n(c_2 + c_3)$$

נסמן : $(c_2 + c_3) = a$, $(c_1 - c_3 + c_4 + c_5) = b$
קיבלנו : $an + b$ משוואה לינארית. המקרה הגרוע ביותר $\Theta(n)$.

#3

א. נשתמש בשיטת האיטרציה :

נניח כי $T(2) = k = \Theta(1)$,

$$T(n) = T\left(n^{\frac{1}{2}}\right) + 1 = \left(T\left((n^{\frac{1}{2}})^{\frac{1}{2}}\right) + 1\right) + 1 = \cdots =$$

$$T\left(n^{\frac{1}{2^i}}\right) + i$$

כאשר $1 \le i \le n$ נבדוק מתי $n^{\frac{1}{2^i}} = 2$: $\log n^{\frac{1}{2^i}} = \log 2$

$$\frac{1}{2^i}\log n = 1$$

$$2^i = \log n$$

$$i = \log(\log n)$$

כלומר לאחר $i = \log(\log n)$ צעדים , הגענו למקרה הבסיס ולכן

$$T(n) = T\left(n^{\frac{1}{2^i}}\right) + i = T(2) + \log(\log n) = \Theta(1) + \log(\log n) = \boldsymbol{\Theta(\log(\log n))}$$

נוכיח באינדוקציה: $T(4) = 1$

$$T(n) = T\left(\sqrt{n}\right) + 1$$

ניחוש : $T(n) = \log(\log n)$

<u>בסיס</u> : $n = 2 \leftarrow T(4) = \log(log4) = \log(2) = 1$

<u>הנחה</u> : נניח שלכל $k < n$ מתקיים $T(k) = \log(logk)$

<u>צעד</u>: נוכיח עבור n , כלומר $T(n) = \log(logn)$:

מהנחת האינדוקציה $T(n) = T\left(\sqrt{n}\right) + 1 = \log\left(\log\left(\sqrt{n}\right)\right) + 1$

$$\log\left(\log\left(\sqrt{n}\right)\right) + 1 = \log\left(\frac{1}{2}\right) + \log(logn) + 1 = log1 - log2 + \log(logn) + 1$$

$$log1 - log2 + \log(logn) + 1 = 0 - 1 + \log(logn) + 1 = \mathbf{\log(logn)}$$

ב. נשתמש בשיטת המאסטר : $T(n) = 5T\left(\frac{n}{2}\right) + n^3\log n$

$$a = 5, b = 2, f(n) = n^3\log n$$

נבדוק לפי מקרה 3 :נבדוק תנאי ראשון :

$f(n) = \Omega\left(n^{\log_2 5+\varepsilon}\right)$ כאשר $\varepsilon = 3 - \log_2 5$:

$$0 \le c \cdot n^{\log_2 5 + (3-\log_2 5)} \le n^3\log n$$

ניקח *c=1* ונקבל $0 \le 1 \cdot n^3 \le n^3\log n$

נבדוק תנאי שני : $0 \le 5(\frac{n^3\log n}{2}) \le d \cdot n^3\log n$

ניקח *d=3* ונקבל : $0 \le 5(\frac{n^3\log n}{2}) \le 3 \cdot n^3\log n$

לכן $\boldsymbol{T(n) = \Theta(n^3\log n)}$

ג. $T(n) = T(cn) + T\big((1-c)n\big) + 1$

תחילה נראה באינדוקציה כי $T(n) = O(n)$ כלומר : $T(n) \le dn - b$. b,d קבועים.

<u>מקרה בסיס</u> : $T(1) = O(1)$

<u>הנחת האינדוקציה</u> : עבור כל $0<k<n$ מתקיים $T(k) = O(k)$ כלומר

$T(k) \le d \cdot k - b$

<u>צעד האינדוקציה</u> : נוכיח כי קיים b כך ש $T(n) \le d \cdot n - b$.

מהנחת האינדוקציה נובע כי :

$$T(n) = T(cn) + T\big((1-c)n\big) + 1 \le dcn - b + dn(1-c) - b + 1$$
$$= dn - 2b + 1 \le dn, \forall_{b \ge 0.5}$$

לכן : $\boldsymbol{T(n) = O(n)}$

כעת נראה באינדוקציה $T(n) = \Omega(n)$ כלומר $T(n) \le mn$. m קבוע.

<u>מקרה בסיס</u> : $T(1) = \Omega(1)$ עבור $m=1$

<u>הנחת האינדוקציה</u> : עבור כל $0<k<n$ מתקיים $T(k) = \Omega(k)$ כלומר

$T(k) \le m \cdot k$

<u>צעד האינדוקציה</u>:

מהנחת האינדוקציה נובע כי:

$$T(n) = T(cn) + T\big((1-c)n\big) + 1 \ge mcn + mn(1-c) + 1 = mn + 1 \ge mn$$

עבור $m=1$ מתקיימת הגדרת החסם התחתון ולכן $T(\mathrm{n}) = \Omega(\mathrm{n})$

לסיכום : $T(n) = O(n)$ וגם $T(\mathrm{n}) = \Omega(\mathrm{n})$ אז $\boldsymbol{T(\mathbf{n}) = \Theta(\mathbf{n})}$.

ד. $T(n) = 2T\left(\frac{n}{2}\right) + n$

נשתמש בשיטת המאסטר מקרה 2 : $a = 2, b = 2, f(n) = n$

$f(n) = \Theta\left(n^{\log_2 2}\right) = \Theta(n)$

לכן $T(n) = \Theta\left(n^{\log_2 2} \cdot logn\right) = \boldsymbol{\Theta(n \cdot log(n))}$

ה. $T(n) = T\left(\frac{n}{3}\right) + T\left(\frac{2}{3}n\right) + n$

ננתח את עץ הרקורסיה למציאת הניחוש :

מצד אחד יש את הרקורסיה של $T\left(\frac{n}{3}\right)$ אשר במקרה הבסיס שלה מתקיים $\frac{n}{3^i} = 1$

כלומר $i = \log_3 n$.

מצד שני יש את הרקורסיה של $T\left(\frac{2}{3}n\right)$ אשר במקרה הבסיס שלה מתקיים $(\frac{2}{3})^k \cdot n = 1$

כלומר $k = \log_{\frac{3}{2}} n$.

בכל שלב בעץ הרקורסיה כמות העבודה היא cn . לכן הניחוש של המקרה "הגרוע" של הכי הרבה קריאות יהיה :

$$T(n) = O(n \log_{\frac{3}{2}} n)$$

והמקרה "הטוב" שיהיה הוא $T(n) = \Omega(n \log_3 n)$

נתבונן בבסיסים של הלוגים (1.5 ,3) ולכן הניחוש הוא ש $T(n) = \Theta(nlogn)$.

נוכיח זאת באינדוקציה עבור חסם עליון וחסם תחתון בנפרד .

חסם עליון : ניחוש $T(n) = O(nlogn)$ כלומר $\forall_{n\geq n_0=2}$, $T(n) \leq c\cdot nlogn$:

<u>בסיס</u> : $T(2) = 1 \leq 2\cdot 2log2 = 4$ (לאחר בדיקת הצעד קבענו כי *c=2*).

<u>הנחת האינדוקציה</u> : נניח שעבור כל $k<n$ מתקיים $T(k) \leq c.klog(k)$

<u>צעד האינדוקציה</u> :

$$T(n) = T\left(\frac{n}{3}\right) + T\left(\frac{2}{3}n\right) + n \leq c\cdot\left(\frac{n}{3}\right)\log\left(\frac{n}{3}\right) + c\cdot\left(\frac{2n}{3}\right)\log\left(\frac{2n}{3}\right) + n =$$

$$\frac{cn}{3}\log(n) - \frac{cn}{3}\log(3) + \frac{2cn}{3}\log(2) + \frac{2cn}{3}\log(n) - \frac{2cn}{3}\log(3) + n =$$

$$cn\cdot\log(n) - cn\cdot\log(3) + \frac{2cn}{3} + n$$

כלומר אנו רוצים ש :

$$cn\cdot\log(n) + (-cn\cdot\log(3) + \frac{2cn}{3} + n) \leq c\cdot nlog(n)$$

לכן צריך להתקיים : $(-cn\cdot\log(3) + \frac{2cn}{3} + n) \leq 0$

$$n\left(\frac{2c}{3} + 1 - clog(3)\right) \leq 0 \quad \Rightarrow c\left(\frac{2}{3} - \log(3)\right) \leq -1 \Rightarrow c \geq \frac{3}{3\log(3) - 2}$$

$$c \geq 1.089$$

לכן ניקח $c = 2$.

מסקנה : $\boldsymbol{T(n) = O(nlog(n))}$.

חסם תחתון : ניחוש $T(n) = \Omega(nlog(n))$ כלומר $\forall_{n\geq n_0=2}$, $T(n) \geq d\cdot nlogn$:

<u>בסיס</u> : $T(1) = 1 \geq 1\cdot 1log1 = 0$ (לאחר בדיקת הצעד קבענו כי *d=1*).

<u>הנחת האינדוקציה</u> : נניח שעבור כל $k<n$ מתקיים $T(k) \geq d.klog(k)$.

<u>צעד האינדוקציה</u> : נוכיח נכונות הטענה עבור n כלומר $T(n) \geq d.nlog(n)$:

$$T(n) = T\left(\frac{n}{3}\right) + T\left(\frac{2}{3}n\right) + n \geq d\cdot\left(\frac{n}{3}\right)\log\left(\frac{n}{3}\right) + d\cdot\left(\frac{2n}{3}\right)\log\left(\frac{2n}{3}\right) + n =$$

$$\frac{dn}{3}\log(n) - \frac{dn}{3}\log(3) + \frac{2dn}{3}\log(2) + \frac{2dn}{3}\log(n) - \frac{2dn}{3}\log(3) + n =$$

$$cn\cdot\log(n) - cn\cdot\log(3) + \frac{2cn}{3} + n$$

כלומר אנו רוצים ש :

$$dn\cdot\log(n) + (-dn\cdot\log(3) + \frac{2dn}{3} + n) \geq d\cdot nlog(n)$$

לכן צריך להתקיים : $(-dn\cdot\log(3) + \frac{2dn}{3} + n) \geq 0$

$$n\left(\frac{2d}{3} + 1 - dlog(3)\right) \geq 0 \quad \Rightarrow d \leq \frac{3}{3\log(3) - 2}$$

$$d \leq 1.089$$

לכן ניקח $d = 1$

מסקנה : $T(n) = \Omega(nlog(n))$

מאחר ו $T(n) = O(nlog(n))$ וגם $T(n) = \Omega(nlog(n))$ אז $\boldsymbol{T(n) = \Theta(nlog(n))}$

ו. $T(n) = 2T(n-1) + 1$

נשתמש בשיטת האיטרציה:

נניח שהמקרה בסיס $T(1) = k = \Theta(1)$

$$T(n) = 2T(n-1) + 1 = 2(2T(n-2) + 1) + 1 = \cdots =$$
$$2^i T(n-i) + 2^i - 1$$

נעצור כאשר $n - i = 1$ כלומר כעבור $i = n - 1$ צעדים.

לכן נקבל:

$$T(n) = 2^{n-1}T(1) + 2^{n-1} - 1 = 2^{n-1} \cdot \Theta(1) + 2^{n-1} - 1 = 2 \cdot 2^{n-1} - 1 = \Theta(2^n)$$

נוכיח באינדוקציה שלמה: $T(n) = \Theta(2^n)$

נניח שעבור כל k<n קיימים d_1, d_2: כך ש $d_1 \cdot 2^k \le T(k) \le d_2 \cdot 2^k$

נוכיח נכונות עבור n, כלומר $d_1 \cdot 2^n \le T(n) \le d_2 \cdot 2^n$:

$$d_1 \cdot 2^n \le T(n) \le d_2 \cdot 2^n = d_1 \cdot 2^n \le 2T(n-1) + 1 \le d_2 \cdot 2^n$$

מהנחת האינדוקציה $d_1 \cdot 2^{n-1} \le 2 \cdot 2^{n-1} + 1 \le d_2 \cdot 2^{n-1}$

$$d_1 \le 2 + \frac{1}{2^{n-1}} \le d_2$$

$$\exists: \boldsymbol{d_1 = 2}, \boldsymbol{d_2 = 3}$$

לכן סה"כ $\boldsymbol{T(n) = \Theta(2^n)}$.

ז. $T(n) = T\left(\frac{9}{10}n\right) + 1$

נשתמש בשיטת המאסטר מקרה 2:

$$a = 1, b = \frac{10}{9}, f(n) = 1$$

$$f(n) = \Theta\left(n^{\log_{\frac{10}{9}} 1}\right) = \Theta(n^0) = \Theta(1)$$

לכן $$T(n) = \Theta\left(n^{\log_{\frac{10}{9}} 1} \cdot \log(n)\right) = \boldsymbol{\Theta(\log(n))}$$

#4

א.

```
function mystery (A[1..n])
    for i ← 1 to n
        index = i
        for j ← i+1 to n
            if (A[j] < A[index])
                index ← j;
            temp ← A[index]
            A[index] ← A[i]
            A[i] ← temp
```

עבור כל איטרציה של i כך ש-i נע בין 1 ל-n

מתבצעת איטרציה של j כך ש j נע בין $i+1$ ל-$n-1$.

נשים לב ששורות 2,5,6,7,8 מתארות פעולה קבועה ולכן לוקחות $\Theta(1)$

$$T(n) = \sum_{i=1}^{n} \sum_{j=i+1}^{n-1} 1 = \sum_{i=1}^{n} n - i = n^2 - \frac{n(n-1)}{2} = \frac{n^2 - n}{2}$$

$$T(n) = \Theta(n^2)$$

ב.

```
function exp(base , n)
    if (n = 0)
        return 1
    else if (n = 1)
        return base
    else
        return base · exp(base,n-1)
```

$$T(1) = \Theta(1)$$
$$T(n) = T(n-1) + c$$

נשתמש בשיטת האיטרציה :

$$T(n) = T(n-1) + c = T(n-2) + c + c = \cdots =$$
$$T(n) = T(n-i) + ic$$

נעצור כאשר $n - i = 1$ כלומר כעבור $i = n - 1$ צעדים :

$$T(n) = T(1) + c(n-1) = \Theta(1) + cn - c = \boldsymbol{\Theta(n)}$$

נוכיח באינדוקציה חסם עליון $T(n) = O(n)$

נניח שעבור כל *k<n* מתקיים $T(k) \le ck$

נוכיח שעבור *n* $T(n) \le cn$

$$T(n) = T(n-1) + c \le c(n-1) + c = cn \le cn$$

עבור *c=1* האי שיוון מתקיים ולכן $T(n) = O(n)$

נוכיח באינדוקציה חסם תחתון כי $T(n) = \Omega(n)$

נניח שעבור כל *k<n* מתקיים $T(k) \ge dk$

נוכיח שעבור *n* $T(n) \ge dn$

$$T(n) = T(n-1) + d \ge d(n-1) + d = dn \ge dn$$

עבור *d=1* האי שיוון מתקיים ולכן $T(n) = \Omega(n)$

לכן לפי משפט : $T(n) = O(n)$ וגם $T(n) = \Omega(n)$ אז $\boldsymbol{T(n) = \Theta(n)}$.

ג.

```
function exp2(base , n)
    if (n = 0)
        return 1
    else if (n = 1)
        return base
    else if (mod(n, 3) = 0)
        tmp ← exp2(base, n/3)
        return tmp · tmp · tmp
    else
        return base · exp2(base,n-1)
```

במקרה הטוב *n* חזקה של 3 ובכל קריאה רקורסיבית נקרא *exp2(base,n/3)* ולכן נשתמש בשיטת האיטרציה :

$$T(1) = \Theta(1)$$
$$T(n) = T\left(\frac{n}{3}\right) + c = \cdots = T\left(\frac{n}{3^i}\right) + ic$$

נעצור כאשר $\frac{n}{3^i} = 1$ כלומר כאשר $i = \log_3 n$ ולכן :

$$T(n) = T(1) + c\log_3 n = \Theta(1) + c\log_3 n = \boldsymbol{\Theta(\log_3 n)}$$

נשים לב שכאשר *n* אינו מתחלק ב-3 ללא שארית, אנו נכנס לשורה 9 מספר פעמים השווה לשארית החלוקה של המספר ב-3,

כלומר בין 1-2 פעמים כלומר $O(1)$ כמו גם שורות 2,4,7.

<u>לפי שיטת המאסטר מקרה שני</u>: $a = 1, b = 3, f(n) = c = \Theta(1)$

$$f(n) = \Theta\left(n^{\log_3 1}\right) = \Theta(n^0) = \Theta(1)$$

לכן $T(n) = \Theta\left(n^{\log_3 1} \cdot \log(n)\right) = \Theta(\log(n)) = \boldsymbol{\Theta(\log_3 n)}$

ד.

```
function expC(base , n)
    if (n = 0)
        return 1
    else if (n = 1)
        return base
    else if (mod(n, c) = 0)
        tmp ← expC(base, n/c)
        ans ← 1
        for i ← 1 to c:
            ans ← ans * tmp
        return ans
    else
        return base · expC(base,n-1)
```

נשים לב שכאשר *c* קבוע אז מדובר בזמן ריצה כמו בסעיף הקודם

כלומר $\boldsymbol{T(n) = \Theta(\log_c n)}$ הרי ששורות 7,8,9,10 כאשר *c* קבוע הן $O(1)$.

כאשר *c=n* התנאי בשורה *5* תמיד יתקיים ולכן נקרא לפונקציה רק עוד **פעם אחת** ונגיע לתנאי עצירה, ואילו בשורה 8 נרוץ *n* פעמים,

ואילו כל שאר השורות מתארות מספר פעולות קבועות ולכן לוקחות $O(1)$.

לכן סך הכל $\boldsymbol{T(n) = \Theta(n)}$

#5

א.

נשתמש במערך עזר B בגודל n ונאתחל אותו באפסים ובנוסף נשתמש במחסנית S שבהתחלה ריקה.

במעבר על מערך הקלט A, עבור כל $A[i]$ ($1 \leq i \leq n$) נבדוק האם $A[i] \neq -1$ - אם כן נסמן $B[A[i]] = 1$, אחרת נכניס את i לתוך המחסנית S, כלומר $S.push(i)$-$O(1)$ (לתוך המחסנית נכניס את כל האינדקסים של המערך A בהם מצוי הערך 1-).

עד כה סך הכל מעבר על מערך הקלט ופעולה קבועה על מערך העזר ייקחו $O(n)$.

לאחר מכן, נעבור על מערך העזר B באמצעות לולאה שרצה מ-$j=1$ עד n ונבדוק האם קיים איבר במערך ששווה ל-0, אם כן נבצע $A[S.pop] = j$- מדובר בפעולה קבועה ולכן $O(1)$,

לכן עד כה מעבר על מערך העזר B זה $O(n)$.

סך הכל $T(n) = 2O(n) = \mathbf{O(n)}$.

QueenD(A)

B ← Array of size N

S ← empty stack

for i=1 to n

if A[i] ≠-1 then

B[A[i]]=1

else S.push[i]

for j=1 to n

if B[j]=0

A[S.pop]=j

return A

ב.

בהינתן שני מערכים ממוינים A ו B נבצע בצורה רקורסיבית חיפוש אחר האיבר ה-k.

נשתמש בפונקציית עזר

נרצה לעצור (מקרה הבסיס) של הרקורסיה יהיה כאשר $k=1$.

על מנת שהאלגוריתם יעבוד אנו מניחים כי גודל המערך A תמיד יהיה גדול מ B - במקרה כזה אנו נקרא לפונקציה עם החלפת תפקידים של מערך A ו-B ונמשיך בתהליך.

עד להגעת המקרה בסיס ובכל איטרציה נגדיר i,j שני אינדקסים למערכים A,B בהתאמה, כאשר הם יקבעו מי המינימום מבין גודל המערך לבין $k/2$ (הקטנת הבעיה).

בכל איטרציה העץ יתחלק ל-2 מקרים :

מקרה אחד : $A[i] < B[j]$ האיבר במקום ה-k שאנו מחפשים יימצא החל מאינדקס $i+1$ ב-A ועד סוף המערך בהכרח מאחר והמערך ממוין , וב-B מתחילת המערך עד האיבר במקום ה- $j+1$.

מקרה שני : $A[i] > B[j]$ האיבר במקום ה-k שאנו מחפשים יימצא מתחילת המערך A ועד אינדקס $i+1$, וב-B מאינדקס $j+1$ ועד סוף המערך, גם כן בגלל שהמערכים ממוינים.

נשים לב שבכל קריאה רקורסיבית (הקטנת הבעיה) אנו שולחים ברקורסיה את k-i או k-j כך שכמו שקבענו כי הם $k/2$, וגם את החלקים הרלוונטיים לפי המקרים של כל מערך.

לכן נעצור כאשר $\frac{k}{2^i} = 1$ כלומר לאחר i קריאות $i = \log_2 k$.

נשים לב שפרט לקריאות הרקורסיביות הללו כל שאר השורות הן פעולות קבועות ולכן $O(1)$.

לכן $T(n) = O(\log_2 k)$

$$T(1) = O(1)$$
$$T(n) = T\left(\frac{n}{2}\right) + c$$

נראה בשיטת האיטרציה :

$$T(n) = T\left(\frac{n}{2}\right) + c = T\left(\frac{n}{2^2}\right) + c + c = \cdots = T\left(\frac{n}{2^i}\right) + ic$$

לאחר i צעדים : $T(n) = T(1) + c\log_2 k = \mathbf{O}(\mathbf{log_2}\,\boldsymbol{k})$

Find(A,B,k)
  return ***FindRecurtion***(A,sizeA,B,sizeB,k)

Assume arrays start from index 1

```
        FindRecurtion(A, sizeA, B, sizeB, k)
(1).    if sizeA > sizeB then //(we assume array A is always smaller than B)
(2).      return Find(B, sizeB , A, sizeA, k)
(3).    if sizeA = 0 and size B > 0 then // base case
(4).      return B[k]
(5).    if k =1 then // base case
(6).      return min(A[1],B[1])
(7).    i ← min(sizeA, k/2)
(8).    j ← min(sizeB, k/2)
(9).   if A[i] < B[j] then //recursion calls
(10).     A' ← Making/looking on the right part of array A(from index i+1 to A.size)
(11).     B' ← Making/looking on the left part of array B(from index 1 to j+1 included)
(12).     return Find( A', sizeA', B', sizeB', k-i)
(13). else
(14).     A' ← Making/looking on the left part of array A(from index 1 to i+1 included)
(15).     B' ← Making/looking on the right part of array B(from index j+1 to B.size)
(16).     return Find( A', sizeA', B', sizeB', k-j)
```