שאלה 1

סעיף א'

1. טענה: $16^{\log n} = O(n^3)$

הפרכה:

$$16^{\log n} = (2^4)^{\log n} = (2^{\log n})^4 = n^4$$

לכל $c \in \mathbb{R}$ מתקיים לכל $\mathbb{N} \ni n > c$:

$$n^4 > c \cdot n^3$$

2. טענה: לכל $k \in \mathbb{N}$ קבוע ופונקציות $f_1, \dots, f_k$, $g_1, \dots, g_k$ כך שלכל $1 \le i \le k$ מתקיים $f_i = O(g_i)$, מתקיים

$$\sum_{i=1}^{k} f_i = O(\max_{1 \le i \le k}\{g_i\})$$

הוכחה: נסמן עבור $1 \le i \le k$ לפי הגדרת $O(\cdot)$, הקל N ו-N_i לכל $n > N_i$ $f_i(n) < c_i \cdot g_i(n)$. נסמן $C = \max_{1 \le i \le k}\{c_i\}$, $N = \max_{1 \le i \le k}\{N_i\}$. אז לכל $n > N$

$$\sum_{i=1}^{k} f_i(n) < \sum_{i=1}^{k} C \cdot g_i(n) < C \cdot \sum_{i=1}^{k} \max_{1 \le i \le k}\{g_i\}(n) =$$

$$\overbrace{C \cdot k}^{\in \mathbb{R}} \cdot \max_{1 \le i \le k}\{g_i\}(n) \Rightarrow \sum_{i=1}^{k} f_i(n) = O(\max_{1 \le i \le k}\{g_i\})$$

3. טענה: אם $f_1(n)=O(g_1(n))$ וגם $f_2(n)=O(g_2(n))$

אז $f_1(n)\cdot f_2(n)=O(g_1(n)\cdot g_2(n))$.

הוכחה: עבור $i\in\{1,2\}$ $f_i < C_i g_i$ החל מ N n_i. נסמן $C=\max_{i\in\{1,2\}}\{C_i\}$,

$N=\max_{i\in\{1,2\}}\{n_i\}$. אז לכל $n>N$:

$$f_1\cdot f_2(n) < \overbrace{C^2}^{\in\mathbb{R}}\cdot g_1\cdot g_2(n) \Rightarrow f_1\cdot f_2(n)=O(g_1\cdot g_2(n))$$

4. טענה: אם $f_1(n)=O(g_1(n))$ / $f_2(n)=O(g_2(n))$ אז

$f_1\circ f_2(n)=O(g_1\circ g_2(n))$

הפרכה:

$$\left.\begin{array}{l} f_1(n)=2^n,\ g_1(n)=2^n \rightarrow 2^n\leq 1\cdot 2^n \\ f_2(n)=2n,\ g_2(n)=n \rightarrow 2n\leq 3\cdot n \end{array}\right\} \Rightarrow \begin{array}{l} F_1=O(g_1) \\ f_2=O(g_2) \end{array}$$

אבל:

$$f_1(f_2(n))=2^{2n},\quad g_1(g_2(n))=2^n$$

לפי טענה מהתרגול $2^{2n}\neq O(2^n)$.

סעיף ב׳

$f, g : \mathbb{R}^+ \to \mathbb{R}^+$ פונקציות הפיכות.

<u>טענה</u>: $f(x) = O(g(x)) \Rightarrow g^{-1}(x) = O(f^{-1}(x))$

<u>הפרכה</u>: נגדיר $f(x) = \log_2 X$ $\quad g(x) = \log_{10} X$

נשים לב שאכן לכל $X \in \mathbb{R}^+$:

$$f(x) = \log_2 X \leq \frac{1}{\log_{10} 2} \cdot \log_{10} X = \underbrace{\frac{1}{\log_{10} 2}}_{C \in \mathbb{R}} g(x)$$

ולכן אכן מתקיים $f(x) = O(g(x))$.

נראה שלכל $C \in \mathbb{R}^+$ קיים $Y_0 \in \mathbb{R}^+$ כך שלכל $Y > Y_0$

$g^{-1}(Y) > C f^{-1}(Y)$.

יהי $C \in \mathbb{R}$. נבחן עבור איזה $Y_0 \in \mathbb{R}^+$ מתקיים לכל $Y > Y_0$:

$$\begin{cases} \mathrm{I} & g(X_p) = \log_{10} X_p = Y \\ \mathrm{II} & f(X_f) = \log_2 (X_f) = Y \\ \mathrm{III} & X_p > C \cdot X_f \end{cases} \equiv \quad g^{-1}(Y) > C \cdot f^{-1}(Y)$$

ב/2:

$$\mathrm{I}\quad \log_{10} X_p = y \rightarrow 10^y = X_p$$

$$\mathrm{II}\quad \log_2 X_f = y \rightarrow 2^y = X_f$$

$$\mathrm{III}\quad X_p > c\cdot X_f \xrightarrow{\mathrm{I}+\mathrm{II}} 10^y > c\cdot 2^y \quad /:2^y>0$$

$$\rightarrow \left(\frac{10}{2}\right)^y > c$$

$$y > \log_5 c$$

כלומר עבור $y_0 = \log_5 c + 1$ מתקיים.

ולכן לא קיים c המקיים את הנדרש ולכן הטענה לא מתקיימת.

סעיף ג׳

$a_1,\dots,a_n$ סדרה של מספרים אי-שליליים, $A = \max\{a_1,\dots,a_n\}$.

1. טענה: אם ישנם שני קבועים $0<b,\ c\le 1$ כך, שלפחות $b\cdot n$ מתוך איברי הסדרה הם בגודל של לפחות $c\cdot A$ אז מתקיים:

$$\sum_{i=1}^{n} a_i = \Theta(n\cdot A)$$

הוכחה:

נשים לב שלכל $1 \le i \le n$ $a_i \le A$ ולכן $\sum_{i=1}^{n} a_i \le \sum_{i=1}^{n} A = nA$

כלומר $\sum_{i=1}^{n} a_i = O(na)$. לפחות $b \cdot n$ איברי הסדרה גדולים של

אחרת אם $A=0$ ברור.

לפחות $c \cdot A$. נמצא $c' \in \mathbb{R}$ כך שלכל n: ⊛ $\sum_{i=1}^{n} a_i \ge c' \cdot nA$

$\sum_{i=1}^{n} a_i \ge (\cancel{n} \cdot b) c \cdot \cancel{A} \ge c' \cdot \cancel{nA}$ $/ : nA > 0$

אם מתקיים אז בהגדרת ⊛ מתקיים.

לכן $\sum_{i=1}^{n} a_i \ge \underbrace{(bc)}_{\in \mathbb{R}} nA$, סה״כ $\sum_{i=1}^{n} a_i = \Theta(nA)$ כנדרש.

2. **טענה:** $\log n! = \Theta(n \log n)$

הוכחה: ראינו בתרגול ש $\log n! = O(n \log n)$. נמצא אילו ערכי

$a < n$ מקיימים עבור $c = \frac{1}{2}$ $\log_2 a > c \cdot \log_2 n$.

מתקיים עבור $a > 2^{c \log_2 n} = 2^{\log_2 n^c} = n^c = \sqrt{n}$ ועבור $n > 4$

מתקיים עבור $a > \frac{1}{2} n$. ואז לפי טענה 1, עבור $b, c = \frac{1}{2}$, נקבל

$\log n! = \Omega(n \log n)$ כלומר $\log n! = \Theta(n \log n)$ והטענה מתקיימת.

3. **טענה:** בהינתן שלם חיובי k נגדיר את הפונקציה הבאה:

$P_k(n) = \sum_{i=1}^{n} i^k$ כש $n \in \mathbb{N}$. לכל k מתקיים

$P_k(n) = \Theta(n^{k+1})$.

הוכחה: נגדיר $a_i = i^k$ עבור $1 \le i \le n$. אז $A = \max_{1 \le i \le n}\{a_i\} = n^k$. נמצא אילו ערכי $a < n$ מקיימים עבור $c = \frac{1}{2}$ ש- $a^k > \frac{1}{2}n^k \Leftarrow a > \sqrt[k]{\frac{1}{2}} \cdot n$.

ולכן עבור $c = \frac{1}{2}$, $b = (1 - \sqrt[k]{\frac{1}{2}})$ לפי סעיף 1

$$\sum_{i=1}^{n} \underbrace{i^k}_{a_i} = \Theta(\underbrace{n^{k+1}}_{n \cdot n^k})$$

כנדרש.

סעיף ד׳

ד. הסיבוכיות היא $\underline{O(n^2)}$. נסביר:

הבדיקה ⊛ "if i in L" היא מסיבוכיות $O(n)$, (במקרה הגרוע i אף פעם לא ב-L ונבדקו n ערכים).

באיטרציה הראשונה של לולאות ה-while, ⊛ מופעלת $\frac{1}{2}n$ פעמים לערך, באיטרציה הבאה $\frac{1}{4}n$ וכך הלאה. כלומר מבוצעות לערך, $\left(\sum_{k=2}^{\log_2 n} \frac{1}{k} n\right) \cdot n$ פעולות (פעולה כאן היא בדיקת ערך בתוך הרשימה).

$$\left(\sum_{k=1}^{\log_2 n} \frac{1}{k} \cdot n\right) \cdot n = n^2 \cdot \underbrace{\sum_{k=1}^{\log_2 n} \frac{1}{k}}_{\text{סכום סדרה הנדסית } = 1 - \frac{1}{n} < 1} \le n^2$$

כנדרש.

[הבהרה לגבי L.append(i): המקרה הגרוע ביותר הוא קורה
שהתנאי לא מתקיים אף פעם ולא מגיעים לשורה הזו. אבל, כיוון
שאם התנאי מתקיים <u>נחסכו לכל הפחות</u> $|L|(1-\frac{1}{2^k})$ איטרציות
(כיוון ש $i \in L[:\frac{len(L)}{2^k}]$ והפקודה "in" לא ממשיכה לרוץ על L
לאחר ש-i נמצא), והתווספו <u>לכל היותר</u> $\frac{1}{2^{k+1}}|L| = \frac{1}{2^{k+1}}|L|(|L|+1-|L|)$.
כמובן ש $\frac{1}{2^{k+1}}|L| \le (1-\frac{1}{2^k})|L|$.]

2. נחשב את כמות הפעולות שהפונקציה מבצעת. נטען שהסיבוכיות
היא <u>$O(n\log^2 n)$</u>. בלולאת ה for הראשונה מבצעים
$(n-500)$ איטרציות, בשנייה $\log_2 i$ לכל איטרציה של הראשונה
$(i\in[n]-[500])$, ובלולאת ה while, $\log_2 n$ איטרציות לכל איטרציה
של ה for השנייה. סה"כ:
↳ k כל פעם גדל פי 2.

$$\sum_{i=500}^{n} \log_2 i \cdot \log_2 n < (n-500)\cdot \log_2^2 n \le n\log^2 n$$

כנדרש.

שאלה 2

סעיף א'

(א) כל מספר בקטע זה הוא חיובי ולכן הערך שלו
בתא ה sign הוא 0. כל מספר גדול או שווה
ל 8^k וקטן ממש מ 8^{k+1} ולכן הערך שלו בתא ה exp
הוא $k+255$. אז ה sign וה exp זהים לכל המספרים.
הערך המקסימלי שניתן להשיג מ fraction הוא זה המיוצג ע"י
7 בכל 12 התאים, ואיתו נקבל מספר שאכן קטן ממש מ 8^{k+1}
כנדרש $(exp=k+255, sign=0)$. המינימלי הוא 0 בכל אחד מ12 התאים
ואיתו נקבל מספר שאכן גדול או שווה ל 8^k (אכן שווה). כל הערכים
שניתן לקבל ע"י fraction ביניהם תקינים. אז יש 12 תאים
ובכל תא ניתן לשים את הספרות $0,1,\dots,7$, כלומר 8 ספרות.
סה"כ $\underline{8^{12}}$ ייצוגים תקינים, כלומר 8^{12} מספרים שונים ב $[8^k, 8^{k+1})$.

סעיף ב׳

(ב) בקטע (א) לפי סעיף א, $8^0=1$, $8^1=8$

יש 8^{12} מספרים הניתנים לייצוג ב $[1,8)$.

$8^1\cdot 2=16$ אז עם 8^1 לכל fraction רק ש

$1\le 1+7\cdot fraction<2$ יש ייצוג, כלומר לכל $0\le fraction<\frac{1}{7}$.

המספר המקסימלי שניתן לייצג ע״י fraction שקטן מ $\frac{1}{7}$

הוא $\sum_{i=1}^{12}\frac{1}{8^i}<\frac{1}{7}$, כיוון ש $\frac{1}{8^{12}}$ זהו הערך הקטן ביותר

שניתן להוסיף לכל ייצוג ב fraction ומתקיים ש:

$\sum_{i=1}^{12}\frac{1}{8^i}+\frac{1}{8^{12}}>\frac{1}{7}$. אם ניקח את אותו המספר המקסימלי

שקטן מ $\frac{1}{7}$ ונחלק ב 8^{-12} (מידת הדיוק המקסימלית) נקבל את

כמות הערכים של fraction שמקיימים את הנדרש ב $(8,16)$

אז סה״כ ישנם $\frac{\sum_{i=1}^{12}\frac{1}{8^i}}{8^{-12}}+8^{12}+\underbrace{1}_{\text{עבור } fraction=0}$ מספרים שונים

ב $[1,16)$.

סעיף ג׳

(c) כל מספר ב $[32,64)$ מקיים ש: סימן $=0$, $exp=256$.

רמת הדיוק של fraction היא עד כדי $\frac{1}{8^{12}}$, כלומר

לכל מספר $x\in(32,64)$ שניתן לייצוג, רק ש

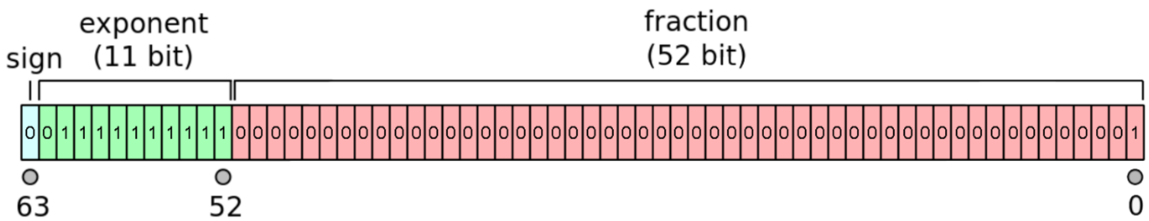
$k=b_{15}$ עבור $0\le k\le 7$ ערך הביט הימני בייצוג שלו,

נוכל לקבל את הקודם שלו ע״י $b_{15}=k-1$ (או שנשלב 1- שמאלה

עד ל $b_i\neq 0$ הראשון). סה״כ:

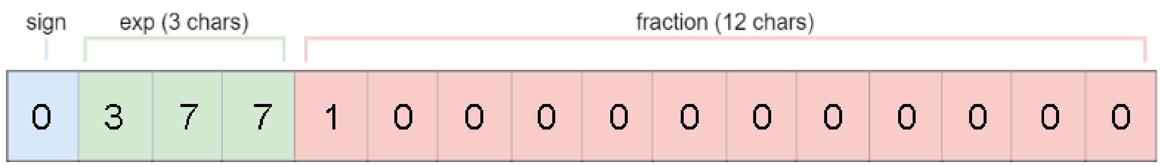
מספר בקטע − הקודם שלו $=8^1\cdot 7\cdot\frac{1}{8^{12}}=\frac{56}{8^{12}}$

סעיף ד׳

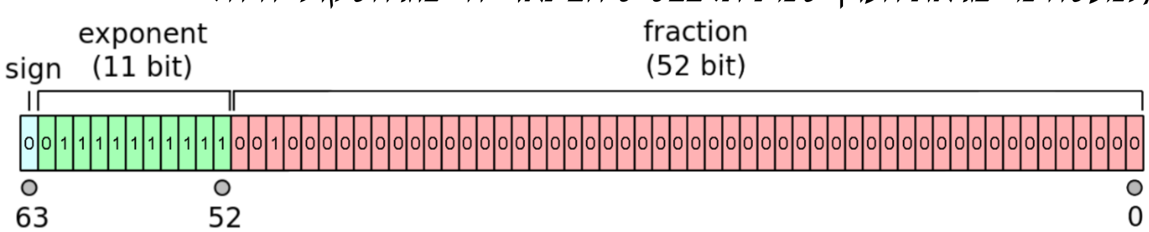


(המספר הקטן ביותר שאינו אפס בשיטת הייצוג האוקטאלי הוא אחת חלקי שמונה בחזקת 12. המספר הנתון קטן ממנו ושווה לאחת חלקי שתיים בחזקת חמישים ושתיים)

סעיף ה׳



(למעשה מייצג את הערך שמינית. בבסיס הבינארי הייצוג השקול יהיה:



)

שאלה 3

סעיף ב׳

מספר משחקים	100	1,000	10,000
מספר סיבובים ממוצע	2.89	2.757	2.714

שאלה 4

סעיף א׳

סיבוכיות זמן הריצה היא O(log(n)). בלולאת הwhile חוצים את הרשימה (עד כדי איבר אחד שמאלה או ימינה, זניח עד כדי הכפל בקבוע של הסיבוכיות) ובלולאת הfor מבצעים עד 4 איטרציות, O(1). (חציית הרשימה בדומה לחיפוש בינארי גוררת O(log(n))).

סעיף ב׳

הפתרון הוא למעשה איטרציה אחת של מיון בועות על הרשימה. כיוון שלא יתכן שאיבר ברח יותר ממקום אחד שמאלה או ימינה, איטרציה אחת מספיקה כדי למיין את הרשימה. ריצה על רשימה באורך n גוררת סיבוכיות של O(n).

סעיף ג׳

איברי הרשימה הם בין 0 לk טבעי כלשהו ולכן גם החציון נמצא בטווח זה. על כל מועמד להיות החציון אנחנו יכולים לדעת אם הוא גדול מדי או קטן מידי. ולכן ניתן לחפש אריה במדבר על הערכים בין 0 לk עד שיתקבל ערך שמקיים את תכונת החציון.

על אף שביצענו חיפוש אריה במדבר, בכל איטרציה (במקרה הגרוע, ולפחות אחת במקרה הטוב) בצענו עבודה בסיבוכיות של O(n), ולכן הסיבוכיות של הפונקציה היא O(n*log(k)).

<u>שאלה 5</u>

<u>סעיף ד׳</u>

אנחנו מתחילים בייצור רשימת מונים באורך 5**k. זה דורש מאתנו 5**k פעולות O(1), סה״כ סיבוכיות O(5**k).

לאחר מכן, אנחנו מפעילים את string_to_int על כל איבר ברשימה שקיבלנו. יש n איברים ברשימה וstring_to_int מסיבוכיות O(number length). כיוון שאורך כל מספר ברשימה הוא k נקבל שסה״כ סיבוכיות הלולאה היא O(n*k).

לבסוף אנחנו בודקים כל איבר ברשימת המונים, O(5**k). סה״כ סכום כל המונים הוא n (כל איבר מהרשימה שקיבלנו הגדיל באחת את אחד המונים). עבור כל מונה אנו מבצעים כערך המונה פעולות של O(k) int_to_string על מספר שמחזיר מחרוזת באורך k). סה״כ n*k פעולות כאלה (במונים מאופסים לא עושים דבר). אם כן סיבוכיות הלולאה השלישית היא O(5**k +n*k).

סה״כ: **O(n*k + 5**k)** = O(5**k) + O(n*k) + O(5**k + n*k).

<u>סעיף ו׳</u>

אנחנו רצים על כל המספרים מ0 עד 5**k. עבור כל מספר אנחנו רצים על כל הרשימה, שבה יש n מחרוזות. על כל מחרוזת אנו מפעילים את הפונקציה string_to_int, שהיא מסיבוכיות אורך המחרוזת, k. סה״כ קיבלנו **O((5**k)*n*k).**