תרגיל בית 4

מגיש : תומר מילדוורט | 316081355

שאלה 1

א.

הוכחה

מכיוון שלפי הגדרת p ישנם n איברים לסדר ב-n מקומות (אינדקסים) שונים ללא חזרות, ניתן לומר כי :

$$p(n) = \frac{n!}{(n-n)!} = \frac{n!}{0!} = \frac{n!}{1} = n!$$

לכן :

$$p(n) = n!$$

ב.

הוכחה

נראה כי :

$$w(n) = 2^{n-1}$$

נוכיח באינדוקציה על n :

בסיס האינדוקציה

נניח כי $n = 1$ ולכן :

$$w(1) = 2^{1-1} = 1$$

כנדרש.

צעד האינדוקציה

נניח את נכונות הטענה עבור $n-1$ ונוכיח עבור n :

$$w(n-1) = 2^{n-2}$$

כדי לקבל את זמן הריצה הארוך ביותר עלינו להפעיל את W על האיבר המקסימלי או המינימלי, לכן, בכדי לקבל את $W(n)$ עלינו להוסיף אופציה נוספת, ולפי עקרון הכפל עלינו לכפול את $W(n-1)$ ב-2 :

$$w(n) = 2 \cdot w(n-1) = 2^{n-1} \cdot 2 = 2^{n-1}$$

כנדרש.

■

ג.

$$\lim_{n\to\infty}\frac{w(n)}{p(n)}=\frac{2^{n-1}}{n!}=\frac{2^n}{2n!}=\frac{2^n}{n!}\cdot\frac{1}{n}$$

ידוע כי:

$$\lim_{n\to\infty}\frac{1}{n}=0\ ,\lim_{n\to\infty}\frac{2^n}{n!}=\infty$$

לכן נפעיל את מבחן המנה:

$$\lim_{n\to\infty}\frac{\frac{2^{n-1+1}}{(n+1)!}}{\frac{2^{n-1}}{n!}}=\lim_{n\to\infty}\frac{2}{n+1}=0<1$$

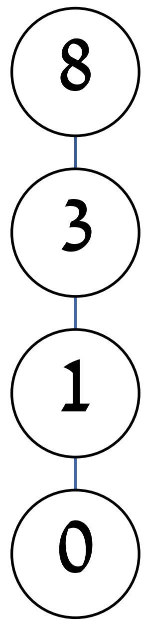
ונקבל כי הסדרה שואפת ל-0.

לכן, ניתן להסיק שככל ש-n גדל (וכך גם $p(n)$ גדל) הסיכוי לקבל את זמן הריצה הגרוע ביותר קטן.

<u>שאלה 2</u>

א1.

עץ הרקורסיה:



א2.

סיבוכיות הזמן של הפונקציה היא $O(n\log n)$.

אורך הרשימה כקלט התלוי ב-n מצטמצם פי 2 בכל איטרציה, ולכן חסום ע״י $O(\log n)$. בנוסף, בכל קריאה רקורסיבית מתבצע חיתוך של הרשימה פי 2 בערך שלם תחתון. פעולה זו חסומה ע״י $O(n)$. לכן, סך כל סיבוכיות הזמן הינה: $O(n)\cdot O(\log n) = O(n\log n)$.

ב.

התקלה בקוד היא שבמידה וערך ה-key נמצא ב-lst הפונקציה תחזיר ערך $None$ בעוד היא אמורה להחזיר את ערך ה-key.

דוגמת הרצה מייצגת:

נניח כי

$$lst = [1,2,3,4], key = 4.$$

לכן, בהכנסת הנתונים לפונקציה נקבל:

$$n = 4$$

כעת, מכיוון שאנו לא עומדים באף אחד מהתנאים, נכנס לבלוק ה-$else$ ונקרא רקורסיבית לפונקציה עם הנתונים:

$$lst = lst[n//2:n] = [3], key = 4$$

כעת, בקריאה הנוכחית:

$$n = 1$$

לכן, נכנס בשנית לתנאי האחרון ונקרא רקורסיבית שוב לפונקציה עם הנתונים:

$$lst = lst[n//2:n] = [], key = 4$$

כעת, נראה כי:

$$n = 0$$

ומכיוון שכעת מתקיים $n \leq 0$ נכנס לתנאי הראשון והפונקציה תחזיר את הערך $None$ במעלה עץ הרקורסיה עד לקריאה האחרונה, למרות שבמצב תקין הייתה אמורה להחזיר 4.

<u>שאלה 3</u>

א.

נוכיח באינדוקציה על n כי במטריצה $had(n)$ כל שורה מלבד השורה העליונה מכילה מספר שווה של אפסים ואחדות:

נסמן:

$$k = |amount\ of\ 1's|, l = |amount\ of\ 0's|$$

לכן, צ"ל:

$$\forall n \in \mathbb{N}. \forall had(n).\ k = l$$

<u>בסיס האינדוקציה</u>

נראה עבור $n = 1$.

נתון כי $had(1) = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$. קל לראות כי $k = 1, l = 1 \Longrightarrow k = l$

צעד האינדוקציה

נניח את נכונות הטענה עבור n ונוכיח עבור $n+1$.

לפי הגדרת מטריצת $had(n)$ עבור $had(n)_{i,j} \in M_n(\mathbb{F})$ מתקיים כי

$$|had(n)_i| = |had(n-1)_i| + |had(n-1)_i|$$

לכן ניתן לומר כי:

$$|had(n+1)_i| = |had(n)_i| + |had(n)_i|$$

לפי הנחת האינדוקציה, עבור $had(n)$ מתקיים $k = l$, לכן, בשורות בהן לא מתבצע חילוף מתקיים עבור $had(n+1)$:

$$k = 2k, l = 2l \implies 2k = 2l \implies k = l$$

במידה ומתבצע חילוף (0 הופך ל-1 ולהיפך), נאמר כי:

$$k = l, l = k \implies l = k$$

כנדרש.

■

ג.

סיבוכיות זמן הריצה של had_local היא $O(n\log n)$. נתחיל בניתוח פונקציות העזר:

$bin\_lst(x, bin\_len)$:

- תחילה, הפעולות המובנות של פייתון הן בסיבוכיות של $O(1)$.
- לאחר מכן, לולאת ה-for מבצעת $O(n)$ איטרציות בסיבוכיות של $O(1)$.
- פעולת ה-$padding$ גם היא מתבצעת בסיבוכיות של $O(n-2)$.

הפונקציה נקראת פעמיים (אחת לכל אינדקס) וסיבוכיות הזמן היא $O(n)$.

$check\_for\_bit(n, bin\_i, bin\_j)$

- תחילה, הפעולות המובנות של פייתון הן בסיבוכיות של $O(1)$.
- השוואת המספר ל-0 דורשת $\lfloor\log_2 n\rfloor + 1$ ביטים ולכן סיבוכיות הזמן היא $O(\log n)$.
- פעולת החיסור על n שומרת את n בזיכרון בכל פעם, לכן $\lfloor\log_2 n\rfloor + 1$ מהווה חסם עליון לכמות הביטים של n, לכן סיבוכיות הזמן היא $O(\log n)$.

הפונקציה נקראת n פעמים, לכן סיבוכיות הזמן בסך הכל היא $O(n)\cdot O(\log n) = O(n\cdot\log n)$.

$\underline{had\_local(n,i,j)}$

כאמור, הפונקציה מבצעת את $bin\_lst(x, bin\_len)$ פעמיים, ומחזירה את הקריאה הרקורסיבית $check\_for\_bit(n, bin\_i, bin\_j)$. לכן, סיבוכיות הזמן הכוללת של הפונקציה היא

$$O(n) + O(n \log n) = O(n\ \log n)$$

<u>שאלה 4</u>

ב.

נניח והמטריצה מסדר $n \times n$.

נראה כי הסיבוכיות היא $O(n^4)$. מכיוון שבמקרה הגרוע נקבל מטריצה מסדר $n \times n$ שכולה אפסים, נראה כי לכל כיוון ("מימין-לשמאל" או "מלמעלה-למטה") ישנן n^2 אפשרויות כ"א, נכפיל את שתי האפשרויות ונקבל:

$$O(2^n) \cdot O(2^n) = O(2^n \cdot 2^n) = O(2^{2n}) = O(4^n)$$

נניח והמטריצה מסדר 10×10.

במקרה זה נראה כי הסיבוכיות היא $O(1)$. לפי הגדרת סיבוכיות הזמן, קיים c כלשהו עבורו סיבוכיות הזמן שווה ל-$O(1)$. נתבונן ב-$c = 4^{10}$ ולכן מתקיים:

$$O(1) = O(c) = O(4^{10})$$

<u>שאלה 5</u>

א.

סיבוכיות הזמן של הקוד שכתבתי היא אקספוננציאלית, משמע קיים $c > 1$ כך שסיבוכיות הזמן של הפונקציה היא $O(2^n)$.

נניח כי אורך הרשימה הוא n. נתבונן במקרה הגרוע (WCT) בה הפונקציה נכנסת ללולאה האחרונה n פעמים, שכן בלולאה זו נוצרות 2 קריאות רקורסיביות שונות: not_in_bag ו- in_bag. אורך הרשימה בכל אחת מקריאות אלו הוא $n - 1$. כל קריאה כזו תכנס גם היא ללולאה האחרונה בקוד, ולכן כמות הקריאות הרקורסיביות תוכפל פי 2 בכל שלב בעץ הרקורסיה. הכפלה זו תקרה $n - 1$ פעמים (לפי ההנחה) כעומק עץ הרקורסיה: $n - 1$. בנוסף, הלולאה מבצעת 2 פעולות נוספות: האחת היא פעולה אריתמטית של חיבור (שלפי הנתון אורכת זמן קבוע) והשנייה היא פונקציית max של פייתון לשני איברים שאורכת $O(2)$. סיבוכיות זמן זו זניחה ולכן נתעלם ממנה כשנחשב את סיבוכיות הזמן של הפונקציה כולה. לכן, סיבוכיות הזמן של הפונקציה חסומה ע"י $O(2^n)$.

ד.

הפונקציה לאחר ממואיזציה חסומה ע"י $O(kn)$. במקרה הגרוע, הפונקציה תעבור על כל הרשימה באורך k כמות של n פעמים.

<u>שאלה 6</u>

ב.

הסיבוכיות היא אקספוננציאלית. נראה כי הפונקציה תבצע 3 קריאות רקורסיביות בכל איטרציה במקרה הגרוע ביותר. לכן, נחשוב על מצב בו $s1, s2$ באורך n, k כל אחת בהתאמה, שונות בכל תו במחרוזות שלהן, או במילים אחרות, קוראות ל-3 קריאות רקורסיביות בכל איטרציה. לכן, בהתעלם מהפעולות האריתמטיות שאורכן קבוע, סיבוכיות הפונקציה תהיה קבוע כלשהו בחזקת המחרוזת הארוכה מבין $s1, s2$ שלצורך הדוגמה יהיה n, ותהיה שווה ל-$O(3^n)$.

ד.

ראשית, בכדי ליצור את המטריצה, נקרא לפונקציה $create\_matrix(s1, s2, cnt, m)$ אשר מבצעת 2 קריאות רקורסיביות התלויות באורך המחרוזת $s1$ שנמסמנה ב-n ולכן הן חסומות ע״י $O(2n)$. נסמן את אורך המחרוזת $s2$ ב-k. נשים לב כי ל-$s1$ נוספים שני מרווחים בתחילת המחרוזת, ול-$s2$ מרווח אחד בכדי לייצר מטריצה מתאימה.

לאחר מכן, נוצרת מטריצה מסדר $(n+2) \times (k+1)$ אך למעשה הקריאות הרקורסיביות של פונקציית העזר $distance\_with\_indexes(i, j, m)$ נקראות על מטריצה מסדר $n \times k$ ולכן נקראות $O(nk)$ פעמים.

בפרט, בלי הגבלת הכלליות, נאמר שהפונקציה חסומה ע״י $O(n^2)$.

לכן, בסיכום כלל הקריאות, נראה כי סיבוכיות הזמן של פונקציית $distance\_fast(s1, s2)$ היא:

$$O(2n) + O(n^2) = O(n^2)$$