CS1001.py HW4

שאלה 1

סעיף א'

טענה: $\boxed{p(n) = n!}$

הוכחה: נוכיח באינדוקציה על $n \in \mathbb{N}^+$. בסיס ($n = 1$): יש בדיוק דרך אחת לסדר את $[1]$, לכן $p(1) = 1$. יהי $n \geq 2$ טבעי. נניח שיש $n!$ דרכים לסדר את הרשימה $[1,2,\dots,n]$, כלומר $p(n) = n!$, ונשאל כמה דרכים יש לסדר את $[1,2,\dots,n+1]$. נשים לב שאת האיבר החדש ניתן להכניס לרשימה המקורית בכל אחד מהאינדקסים 0 עד $n-1$ (למשל באמצעות המתודה $instert$) או בסוף הרשימה, כלומר $n+1$ אפשרויות. עבור כל אפשרות כזאת ניתן להתייחס לשאר הרשימה כאל רשימה המורכבת מהאיברים 1 עד n, ולפי הנחת האינדוקציה ישנן $n!$ אפשרויות לסדרה. סה"כ ישנן $(n+1) \cdot n! = (n+1)!$ אפשרויות לסידור הרשימה $[1,2,\dots,n+1]$, כלומר $p(n+1) = (n+1)!$. ■

סעיף ב'

טענה: $\boxed{w(n) = 2^{n-1}}$

הוכחה: ראינו בכיתה שה-WC מתקבל כאשר ה-$pivot$ שנבחר הוא האיבר המקסימלי או המינימלי ברשימת הקלט.
ה-$pivot$ שנבחר בפונקציה $\det\_quicksort$ הוא האיבר הראשון ברשימת הקלט. לכן, ברשימה הנתונה lst הסידורים עבורם ריצת הפונקציה היא הארוכה ביותר הם רק מאלו שהאיבר הראשון בהם הוא 1 או n, בפרט שתי אפשרויות.
לאחר הבחירה הזו נקבל שלוש רשימות, אחת ריקה (עבור n ה-$greater$ והפוך), השנייה מכילה את $pivot$ בלבד, והשלישית עם יתר $n-1$ האיברים שנשארו. בשלב הרקורסיבי הבא נרצה שוב שהאיבר הראון ברשימה עם יתר $n-1$ האיברים יהיה המינימלי או המקסימלי, ובפרט שוב יש 2 אפשרויות לכך. מעיקרון הכפל הקומבינטורי, נכפול את שתי האפשרויות בכל שלב רקורסיבי, עד שברשימה עם $k-1$, כאשר k היא כמות האיברים בשלב הרקורסיבי הנוכחי, יישאר איבר יחיד, כלומר סה"כ: $1 \cdot \prod_{k=1}^{n-1} 2 = 2^{n-1}$. ■

סעיף ג'

נחשב את הגבול:

$$\lim_{n\to\infty}\frac{w(n)}{p(n)} = \lim_{n\to\infty}\frac{2^{n-1}}{n!} = \lim_{n\to\infty}\frac{(1+1)^n}{2n!} \underbrace{=}_{\left(\substack{\text{הבינום}\\\text{של}\\\text{ניוטון}}\right)} \lim_{n\to\infty}\frac{\sum_{k=0}^{n}\left(\binom{n}{k}\cdot 1^k \cdot 1^{n-k}\right)}{2n!} = \lim_{n\to\infty}\frac{\sum_{k=0}^{n}\frac{n!}{k!\,(n-k)!}}{2n!}$$

$$= \frac{1}{2}\lim_{n\to\infty}\sum_{k=0}^{n}\frac{1}{k!\,(n-k)!} \underbrace{=}_{\left(\substack{\text{לינאריות}\\\text{הגבול}}\right)} \frac{1}{2}\sum_{k=0}^{n}\lim_{n\to\infty}\frac{1}{k!\,(n-k)!} = 0$$

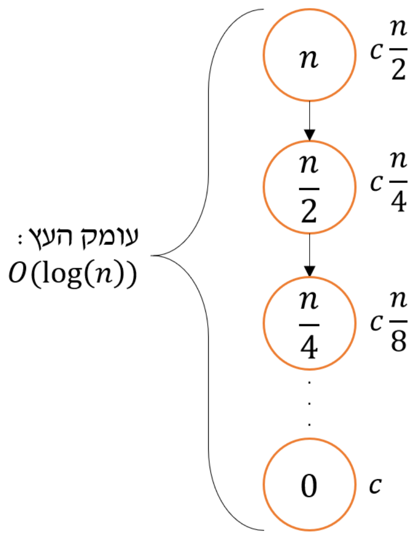
בגבול זה אנו מחשבים למעשה את השכיחות היחסית של הסידורים של הרשימה $[1,2,\dots,n]$ שאם יינתנו ל-$\det\_quicksort$ כקלט, יניבו את זמן הריצה הגרוע ביותר, מתוך כלל הסידורים האפשריים. הגבול של שכיחות יחסית זו הוא 0 עבור $n \to \infty$, וממנו ניתן להסיק שהסיכוי לכך שזמן הריצה של האלגוריתם יהיה גרוע ביותר ככל ש-n גדל, הוא **קטן** יותר ויותר.

שאלה 2

סעיף א'

1- עץ הרקורסיה עבור $key = 8$, $lst = [1,2,3,4,5,6,7,8]$, כאשר כל צומת מייצג את גודל הקלט באותו שלב - אורך הרשימה:

2- ניעזר בעץ רקורסיה עבור $key = 0$, $lst = [1,2,\dots,n]$, בדומה ל-1:



קלט זה מייצג את ה-WC משום שהמפתח לא נמצא ברשימה ולכן מספר הקריאות הרקורסיביות יהיה המקסימלי (בכל מקרה אחר היינו נכנסים לתנאי עצירה בטרם אורך הרשימה מתאפס). בכל שלב רקורסיבי הקלט שלנו קטן בכחצי עד שגודלו 0. לכן, בבירור ישנן כ-$\log(n)$ קריאות רקורסיביות. באשר לעבודה שנעשית בכל צומת לא כולל הקריאה הרקורסיבית, העבודה הדומיננטית מתבצעת ע"י ה-$slicing$ שבקריאה הרקורסיבית, שגודלו כמחצית מגודל הקלט באותה צומת, ולכן זוהי גם כמות העבודה של פעולה זו (העבודה של $slicing$ היא כאורכו). שאר העבודה בכל צומת היא קבועה וכוללת גישה לאורך רשימה, השמה, גישה לאיבר ברשימה לפי אינדקס ובדיקת תנאים בוליאניים. סך העבודה על כן היא סכימת כמות העבודה שכל צומת תורם:

$$T(n) = \sum_{i=1}^{\log(n)} \frac{cn}{2^i} = cn \sum_{i=1}^{\log(n)} \left(\frac{1}{2}\right)^i \underbrace{=}_{\text{(סכום סדרה הנדסית)}} cn \frac{\frac{1}{2} \cdot \left(1 - \left(\frac{1}{2}\right)^{\log(n)}\right)}{1 - \frac{1}{2}}$$

$$= cn - cn \cdot \frac{1}{n} = O(n)$$

כלומר סה"כ: $\boxed{\boldsymbol{O(n)}}$.

סעיף ב'

הקוד לא מחזיר פלט נכון רק כאשר המפתח נמצא ברשימה והתוכנית מבצעת לפחות קריאה רקורסיבית אחת מה-$else$, שכן אז בהימצא אותו מפתח התוכנית לא תיקח בחשבון את החיתוך/כים משמאל, ותחזיר אינדקס שגוי.

דוגמה מייצגת (הפונקציה הועתקה אחד לאחד מקובץ השאלות):

```
In[3]: rec_slice_binary_search([1, 2, 3, 4, 5], 4)
Out[3]: 0
```

הפלט התקין שהיינו מצפים לקבל הוא 3.

שאלה 3

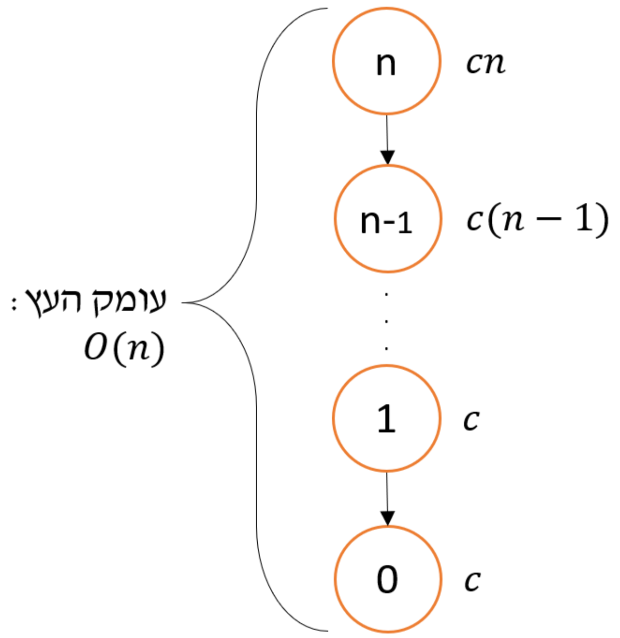
סעיף א'

נוכיח באינדוקציה על $n \in \mathbb{N}$.
בסיס ($n = 0$): המטריצה $had(0) = [[0]]$ מקיימת את התכונה באופן ריק, שכן התכונה מציבה תנאי לכל שורה מלבד השורה העליונה, ואילו במטריצה זו אין שורה מלבד העליונה.
צעד: יהי $n \geq 2$ טבעי. נניח שבמטריצה $had(n-1)$, שגודלה $2^{n-1} \times 2^{n-1}$, כל שורה מלבד השורה העליונה מכילה מספר זהה של אפסים ואחדות. ננתח את המבנה הרקורסיבי $had(n)$: "זוהי מטריצת בלוקים עם שני בלוקים עליונים של $had(n-1)$" – כלומר כל תת-רשימה (המייצגת שורה) בתחום האינדקסים 0 עד $2^{n-1} - 1$ ברשימה $had(n)$, מהווה למעשה שרשור של כל שורה ב-$had(n-1)$ לעצמה בהתאמה (באותו סדר שמופיע ב-$had(n-1)$). נשים לב שבתחום האינדקסים 0 עד $2^{n-1} - 1$ ברשימה $had(n)$, כל שורה מלבד השורה העליונה מכילה מספר זהה של אפסים ואחדות, מכיוון שלפי הנחת האינדוקציה כל חצי רשימה (שמהווה רשימה ב-$had(n-1)$ ושורשרה לעצמה לכדי תת-רשימה באותו מיקום ב-$had(n)$) מכילה מספר זהה של אפסים ואחדות מלבד השורה העליונה – הכפלה ב-2 של רשימה בינארית שומרת על היחסים בין האפסים לאחדות. "ושני בלוקים תחתונים של $had(n-1)$ כאשר בבלוק הימני-תחתון ערכי המטריצה מתהפכים" – כלומר כל תת-רשימה (המייצגת שורה) בתחום האינדקסים 2^{n-1} עד $2^n - 1$ ברשימה $had(n)$, מהווה למעשה שרשור של כל רשימה (שורה) ב-$had(n-1)$ לרשימה עם הערכים הבינאריים ההפוכים של אותה שורה, מימין לשמאל בהתאמה, באותו סדר שמופיע ב-$had(n-1)$. נשים לב שבתחום האינדקסים $2^{n-1} + 1$ עד $2^n - 1$ ברשימה $had(n)$, כל שורה מכילה מספר זהה של אפסים ואחדות, מכיוון שלפי הנחת האינדוקציה כל חצי שמאלי של רשימה (שמהווה רשימה ב-$had(n-1)$) מכילה מספר זהה של אפסים ואחדות מלבד השורה העליונה (מיד נטפל גם בשורה 2^{n-1} ב-$had(n)$), ולכן גם כל חצי ימני של רשימה יכיל מספר זהה של אפסים ואחדות – היפוך רשימה בינארית בה מספר האפסים והאחדות זהה לא משנה את שוויון הכמויות שלהם במצב ההפוך.

באשר לשורה 2^{n-1} ב-$had(n)$, גם היא מהווה למעשה שרשור של הרשימה הראשונה ב-$had(n-1)$ לרשימה עם הערכים הבינאריים ההפוכים שלה. מכיוון שהיא מכילה אפסים בלבד, היא משורשרת לרשימה עם אחדות בלבד באותו האורך, ולכן גם שורה זו מקיימת את התכונה שבשאלה. כיסינו את כל השורות של $had(n)$, 2^n, וראינו שהתכונה מתקיימת כלשונה.

סעיף ג'

ניעזר בעץ רקורסיה כאשר כל צומת מייצג את גודל הקלט באותו שלב רקורסיבי ו-n הוא המספר הטבעי עבורו מטריצת $Hadamard$ בגודל $2^n \times 2^n$. ליד כל צומת מצוינת כמות העבודה:



בכל קריאה רקורסיבית, ערכו של n קטן ב-1 עד שמגיע ל-0, לכן בבירור ישנם בדיוק $n + 1$ שלבים רקורסיביים (כולל הראשון).
נגדיר את k להיות גודל הקלט בצומת בתחום $1 \leq k \leq n$.
באשר לכמות העבודה בכל צומת, מלבד הצומת 0, מחושב הביטוי $pow(2, k-1)$ ולפי ההנחיה בתרגיל, ניתן להניח שהעבודה היא $O(k)$. לאחר מכן נעשית השמה (זמן קבוע $O(1)$), ונבדקים תנאים בוליאניים עבור משתנים השמורים בזיכרון, ולכן ההנחה היא שבדיקת התנאים מתבצעת בזמן קבוע. בחלק מהקריאות הרקורסיביות מתבצעים חישובים מסוג חיסור. i ו-j ערכם לכל היותר $2^k - 1$. כלומר צריך לכל היותר k ביטים לייצגם. באשר ל-$critic\_val$, המשתנה שמצביע על $pow(2, k-1)$, ערכו בדיוק 2^{k-1}, כלומר צריך בדיוק k ביטים לייצגו. ראינו שפעולת חיסור בין שני מספרים בני k ביטים כל אחד עולה $O(k)$. את החיסור הזה נעשה פעם, פעמיים או אפס פעמים כתלות באיזה תנאי בוליאני מתקיים.
ישנה גם פעולת חיסור נוספת, אבל היא מתרחשת בזמן קבוע כי היא $0 - 1$ או $1 - 1$, בפרט לא תלויה ב-k. בכל מקרה, כמות העבודה הכוללת בכל צומת היא $O(k)$. הצומת $k = 0$ תורמת לנו זמן קבוע בלבד שכן רק בודקת תנאי ומחזירה 0. לקבלת כמות העבודה הכוללת נסכום על הצמתים:

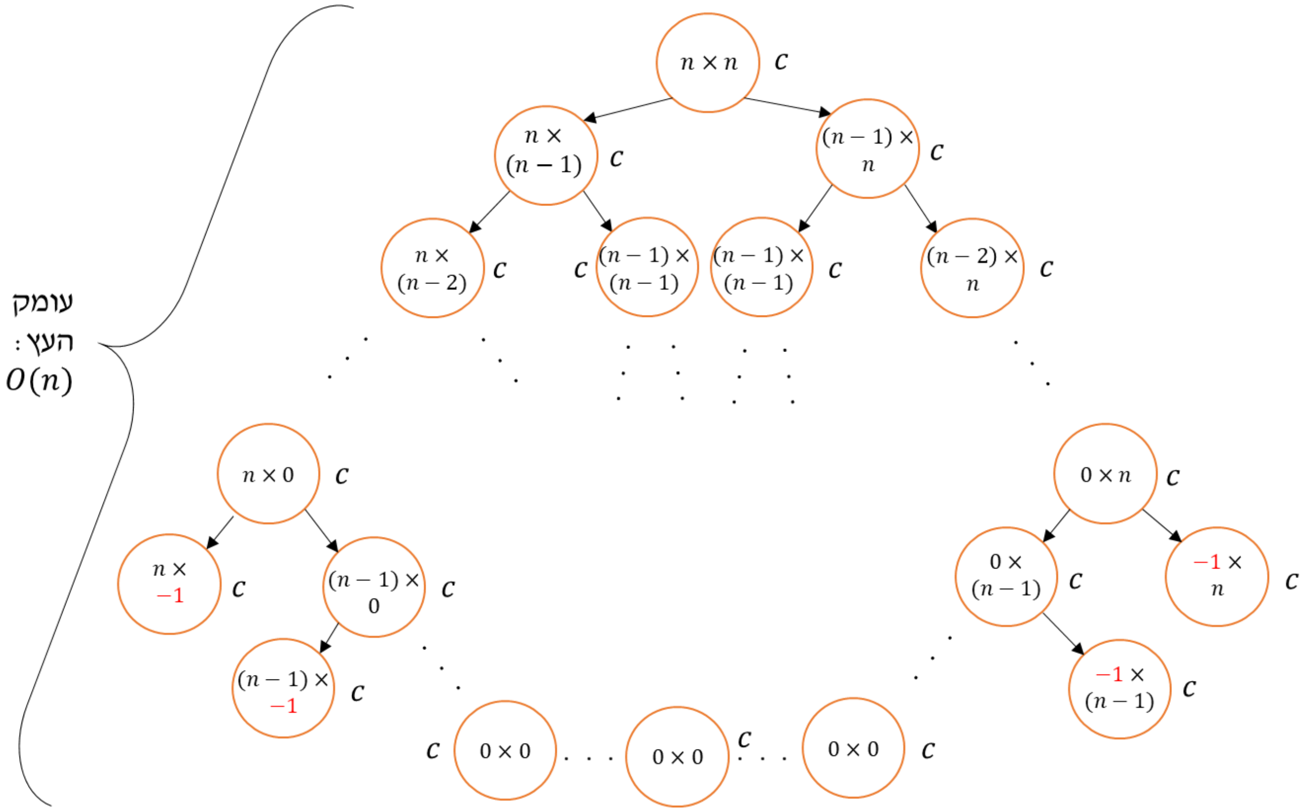
$$T(n) = c + \sum_{k=1}^{n} c \cdot k = c + c \cdot \frac{(1+n) \cdot n}{2} = \boldsymbol{O(n^2)}$$

כלומר זמן ריצה **ריבועי** ב-n.

שאלה 4

סעיף ב'

ניתוח סיבוכיות זמן הריצה תלוי מאוד באיך שהמטריצה נראית. נוכל לנתח את סיבוכיות הזמן של מטריצת האפס בגודל $n \times n$, שמהווה את ה-WC, כי עבורה נפתחות הכי הרבה קריאות כי יש הכי הרבה מסלולים. ניעזר בעץ רקורסיה כאשר כל צומת מייצג את גודל המטריצה באותו שלב רקורסיבי, וליד כל צומת מצוינת כמות העבודה :



נעיר שהעלים בהם מופיע -1, מייצגים את השלבים הרקורסיביים בהם i או j חורגים מקצות המטריצה, נכנה עלים אלה "עלים חורגים". בכל צומת מתבצעת עבודה קבועה, שכן היא כוללת לכל היותר בדיקת תנאים בוליאניים על אורך המטריצה, גישה לאיבר ברשימה מקוננת לפי אינדקס, חיבור וחיסור עם 1, שינוי $in-place$ של איבר ברשימה מקוננת, וחיבור מספר (r) עם 0 – כל אלה פעולות שעולות $O(1)$. בנוסף, בפונקציית המעטפת מתבצעת עבודה של $O(n^2)$ באתחול (השמת אפסים) המטריצה שכל תא בה מחזיק את מספר המסלולים השונים שעוברים בתא הזה, מכיוון שהאתחול מתבצע בלולאות מקוננות, כאשר שתי הלולאות ב"ת ורצות על טווח של n. זוהי העבודה המשמעותית ביותר בפונקציית המעטפת והיא חד פעמית. נותר לספור את כמות הצמתים. נשים לב שכל מסלול משורש לעלה (שאינו חורג) מייצג למעשה מסלול אפסים אפשרי במטריצה. מכיוון שזוהי מטריצת האפס, שאלה דומה היא כמה מסלולים קיימים מהקצה הימני העליון של המטריצה לקצה השמאלי התחתון, והיא שקולה קומבינטורית לשאלה של $counting\ paths$ שראינו בתרגול. נניח לרגע שכל צעד ימינה שונה זה מזה וכן כל צעד למטה שונה זה מזה. ישנם $(2n)!$ סידורים אפשריים, כי מספר הצעדים הכולל הוא $2n$. אבל מכיוון שכל הצעדים ימינה זהים צריך לחלק גודל זה ב-$n!$, מספר הצעדים ימינה. כנ"ל לגבי הצעדים למטה, וקיבלנו את $\binom{2n}{n}$. ראינו בתרגול את קירוב סטרלינג ולכן : $\binom{2n}{n} \simeq \frac{4^n}{\sqrt{2}\sqrt{n}} = O\left(\frac{4^n}{\sqrt{n}}\right)$, ובפרט אקספוננציאלי ב-$n$. לכן גם זמן הריצה אקספוננציאלי ב-n.

עבור מטריצה בגודל 10×10, גם אז ב-WC הניתוח דומה למה שעשינו לעיל עבור מטריצה בגודל $n \times n$ – זמן הריצה של כל צומת אינו תלוי בגודל הקלט וישנם $\binom{20}{10} = 184{,}756$ מסלולים אפשריים. אם זמן הריצה אינו תלוי בגודל הקלט, הרי שהסיבוכיות היא $O(1)$.

שאלה 5

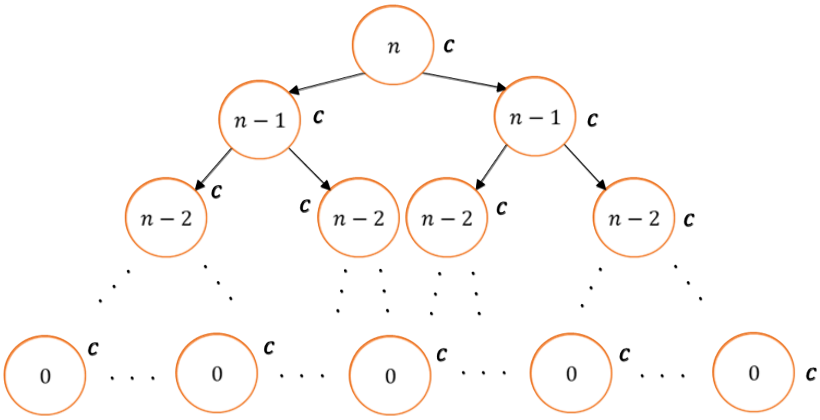
סעיף ב'

4. הסיבוכיות היא **אקספוננציאלית**, כלומר קיים קבוע $c > 1$ כך שסיבוכיות הזמן של הפונקציה היא $O(c^n)$.

סיבוכיות זמן הריצה כתלות באורך הקלט תלויה בגודלו של k. ה-WC מתקבל עבור $k \geq \sum_{i=0}^{n-1} w_i$ (שאז כל הקומבינציות נבדקות בלי להגיע לתנאי עצירה על $k < 0$).

נתבונן בעץ הרקורסיה של ה-WC:

בתוך כל צומת בעץ מצוין אורך הרשימה הרלוונטי לשלב הרקורסיבי הנוכחי, ולידו מצוינת כמות העבודה.
ראשית, בכל צומת נבדקים תנאים בוליאניים, מבוצעות פעולות אריתמטיות, הוספה או שינוי $in-place$ של איבר ברשימה וגישה לאיבר ברשימה לפי אינדקס. כל אלה פעולות שמבוצעות בזמן קבוע (לגבי הפעולות האריתמטיות – זוהי ההנחה בתרגיל). סה"כ כל צומת תמיד $O(1)$. לכן סיבוכיות הזמן תיקבע לפי כמות הצמתים בעץ הרקורסיה.



זהו עץ בינארי מלא שאנו מכירים ולפי סכום סדרה הנדסית, מספר הצמתים הוא בדיוק $2^{n+1} - 1$, ולכן הסיבוכיות היא $O(2^n)$.

סעיף ד'

סיבוכיות זמן הריצה לא משתנה אסימפטוטית במקרה הגרוע, כלומר $\boldsymbol{O(2^n)}$, שכן במקרה זה הסכום של כל שני ערכים ברשימה W עד למקום מסוים n תיתן ערך k שונה, ולכן אף זוג סדור (n,k) לא יחזור על עצמו במילון ויעילות הממואיזציה תהיה חסרת משמעות ואפילו פחות טובה בשל תיחזוק המילון. למעשה, אלא אם מתכננים את W היטב כך שתהיינה מגוון אפשרויות ליצירת אותו k עבור אותו n בכמה שיותר מקומות בקלט (ואז יהיו שימושים ניכרים יותר במילון), יעילות הממואיזציה לא תורגש. מבדיקה על רשימות W שונות בעלות ערכים אקראיים ניכר שיפור מסוים אך לא גבוה במיוחד ובוודאי שלא אסימפטוטית.

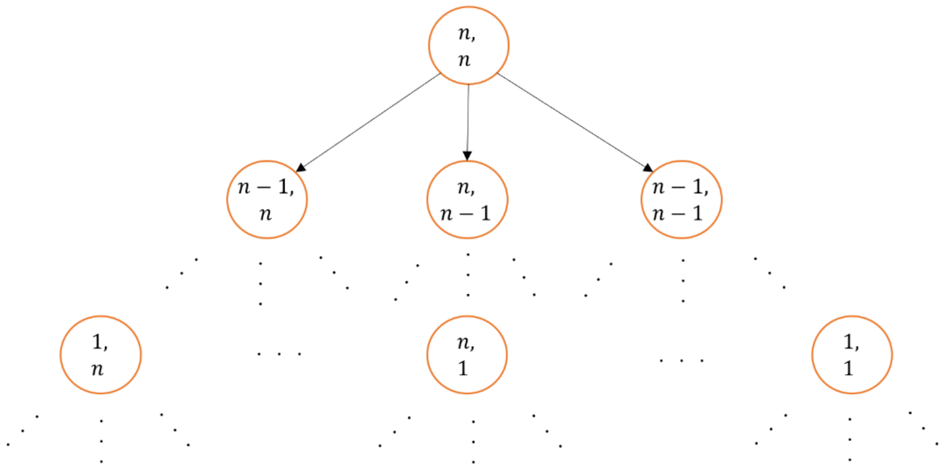
שאלה 6

סעיף ב'

8. הסיבוכיות היא **אקספוננציאלית**, כלומר קיים קבוע $c > 1$ כך שסיבוכיות הזמן של הפונקציה היא $O(c^n)$.

טענה: זמן הריצה **לכל הפחות** $\boldsymbol{3^n}$.

נצייר עץ רקורסיה ונציין בתוך כל צומת את הגודל הרלוונטי של כל אחת מהמחרוזות (הגודל הרלוונטי נקבע באמצעות האינדקסים $i1, i2$). ניקח כדוגמה את המקרה בו אורך שתי המחזורות הוא n – ואין תו זהה בשתיהן.

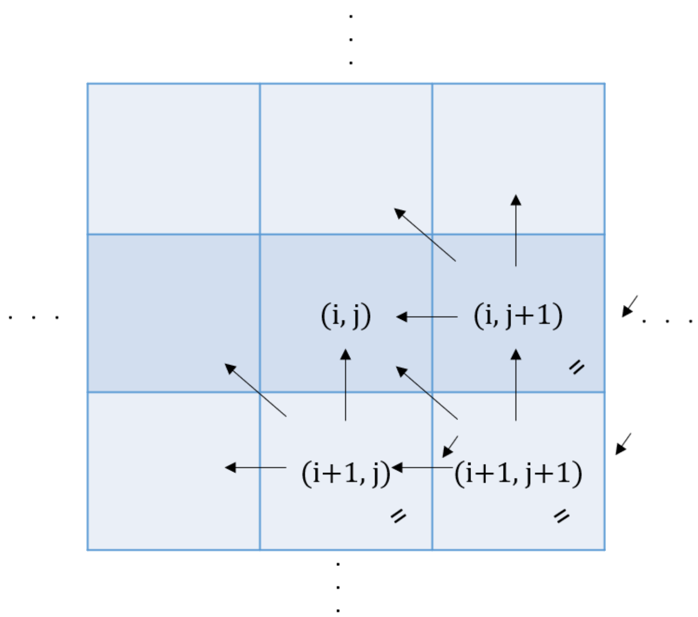


נשים לב שבכל רמה בעץ אורך כל מחרוזת יורד לכל היותר ב-1, ולכן לא משנה איך נרד בעץ, לאחר $n-1$ רמות אורך כל מחרוזת תמיד חיובי. לכן בכל $n-1$ הרמות הראשונות, כל צומת יתפצל ל-3 ולא יגיע לתנאי עצירה, ולכן ב-n הרמות הראשונות העץ מלא, ולכן גודל העץ $\leq 3^n$. לכן גם זמן הריצה הוא לכל הפחות 3^n.

סעיף ד'

טענה: זמן הריצה - $\boxed{\boldsymbol{O(n^2)}}$

נניח לחומרה שאורך שתי המחזורות הוא n – ואין תו זהה בשתיהן. נעשה את הניתוח באמצעות טבלה בגודל $(n+1) \times (n+1)$, כאשר (i, j) יסמן את השלב הרקורסיבי בו החלק הרלוונטי של הרשימה $s1$ הוא באורך i והחלק הרלוונטי של הרשימה $s2$ הוא באורך j. ידוע שגודל הטבלה הוא $O(n^2)$ ושזמן החישוב של כל תא הוא לכל היותר c (קבוע), שהרי בכל שלב רקורסיבי מלבד הקריאות הרקורסיביות מבוצעות פעולות אריתמטיות עבורן אנו מניחים שרצות בזמן קבוע, בדיקות תנאים בוליאניים וחיפוש, השמה ושליפה ממילון, שלשם הפשטות מניחים שפעולות אלו על המילון הן $O(1)$ ולא רק בממוצע. אם נדע מהו מספר הפעמים לכל היותר שביקרנו בכל תא, נדע מהו זמן הריצה לכל היותר.
נסתכל על התא הכללי (i, j) ונחשוב מי יכול היה לקרוא לו:



רק $(i+1,j)$, $(i,j+1)$ או $(i+1,j+1)$ יכולים לקרוא ל-(i,j). בנוסף, אם מחשבים את (i,j) לא חוזרים אל שלושתם, כי בטבלה ניתן להתקדם רק שמאלה, למעלה או באלכסון, ולכן בפרט עד שלא נסיים את החישוב של (i,j), לא נגיע לאף אחד מהם. כעת נניח שהגענו ל-(i,j) מ-$(i+1,j)$. ברגע שסיימנו לחשב את (i,j), צריך לחשב גם את $(i+1,j-1)$ ואת $(i,j-1)$, ומהם לא נגיע ל-(i,j) בגלל כיוון ההתקדמות בטבלה. כשנחשב אותם יהיה לנו את $(i+1,j)$ באמצעות המינימום שלהם. בהמשך מתישהו נרצה לחשב גם את $(i,j+1)$ ולכן נצטרך לחשב את $(i-1,j+1)$, את (i,j) ואת $(i-1,j)$, כאשר מהשניים החדשים לא נגיע ל-(i,j), וברגע שנחשב אותם יהי לנו את הערך של $(i,j+1)$ לפי המינימום מביניהם. וזהו הערכים של $(i+1,j)$ ו-$(i,j+1)$ נמצאים במילון ולא נחשבם יותר. ולכן כאשר בהמשך נרצה לחשב את $(i+1,j+1)$, נגיע רק ישירות ממנו ל-(i,j) ואז גם הערך של $(i+1,j+1)$ יהי במילון ולא נחשבו יותר, ולכן אף פעם לא נצטרך לחשב את (i,j) יותר כי הם היחידים שיכולים לקרוא לו והם במילון.

<u>מסקנה</u>: מבקרים ב-(i,j) לכל היותר שלוש פעמים ולכן זמן הריצה הוא לכל היותר מספר התאים בטבלה כפול מספר הפעמים שביקרנו בכל תא לכל היותר כפול זמן החישוב של כל תא בטבלה, כלומר $O(n^2)$.