תרגיל 4- אלגוריתמים 67504:

שמות: דן קורנפלד, מתן קמינסקי

שאלה 1:

בעיה מס 1:

1. ניסוח בהיר של תתי הבעיות:
לכל $1 \leqslant i, j \leqslant n$ ול- $0 \leqslant m \leqslant n-1$ נמצא את משקל המסלול המינימלי בין v_i ל-v_j המשתמש בלכל היותר m צלעות (נסמן מחיר זה ב- $(f(i, j, m)$
2. נוסחת רקורסיה והסבר לבניתה:

$$f(i,j,m) = \begin{cases} f(i,j,1) \ = \ w(i,j) \rightarrow if \ (i,j) \in E, \ \infty \rightarrow otherwise & m=1 \\ min \ \left\{ \{(f(i,k,m-1) + f(k,j,1)) : \forall k \in V \ ; \ (k,j) \in E, \ \underbrace{f(i,j,m-1)}_{\text{הערך הנמוך ביותר עד } m-1} \right\} & m>1 \end{cases}$$

הסבר: עבור m=1 נגדיר את המרחק המינימלי להיות משקל הצלעות בין 2 הקודקודים, אם אין צלע המשקל הוא ∞. לכל $m > 1$ נגדיר את התא בטבלה להיות הערך המינימלי (המרחק) בין הערך המינימלי שהיה עד $m-1$ צלעות שבו הצלע נגמרת בקודקוד k + הצלע (k, j) לבין הערך הנוכחי שכבר קיים בטבלה עבור המרחק בין 2 הקודקודים.
הערה: לפי בניה זו ניתן להסיק כי לכל $f(i, k, m)$ נקבל את המרחק המינימלי עד m צלעות, מה שמאפשרת ההנחה שכאשר נוסיף את הצלע (k, j) וניקח את המינימלי מבין האפשרויות, בהכרח נקבל את המרחק המינימלי.
3. תיאור של הטבלה, סדר המילוי שלה ואופן חילוץ הפתרון:
נבנה את הטבלה מ$m = 0$ עד $m = n-1$, עבור כל תא (i, j) נרוץ על כל $1 \leqslant i \leqslant n$ עבור כל $1 \leqslant j \leqslant n$, וכך נמלא כל תא בטבלה לפי נוסחת הרקורסיה ולפי התאים שקדמו לתא הנוכחי.
אופן חילוץ הפתרון: עבור כל 2 קודקודים (i, j) מהגדרת הפתרון, התא $f(i, j, n-1)$ יכיל את המרחק המינימלי עבור המרחק מ-i ל-j
4. ניתוח זמן ריצה: עבור כל תא בטבלה לוקח לנו $O(|E|)$ עלות חישוב, היות ואנו בודקים בין כל הצלעות הרלוונטיות המגיעות לקודקוד j, כך עבור n^3 תאים, לכן עלות כוללת של $O\left(n^3|E|\right)$

בעיה מס 2:

1. ניסוח בהיר של תתי הבעיות:
לכל $1 \leqslant i, j \leqslant n$ ו-$0 \leqslant k \leqslant n$ נמצא את המשקל המינימלי של מסלול בין v_i ל- v_j שמשתמש רק בקודקודים $\{v_1, ..., v_k\}$ כקודקודי ביניים במסלול. נסמן מחיר זה ב- $g(i, j, k)$ כאשר $g(i, j, 0)$ אומר שהמסלול בין v_i ל-v_j לא משמש בשום קודקוד ביניים.
2. נוסחת רקורסיה והסבר לבניתה:

$$g(i,j,k) = \begin{cases} g(i,j,0) \ = \ w(i,j) \xrightarrow{if} (i,j) \in E, \ \infty \rightarrow otherwise & k=0 \\ min \ \left\{ \{(g(i,z,k-1) + g(z,j,k-1)) : z = v_k, \ \underbrace{g(i,j,k-1)}_{\text{הערך הנמוך ביותר עד } k-1} \right\} & k>0 \end{cases}$$

הסבר: עבור $k=0$ נגדיר את המרחק המינימלי להיות משקל הצלעות בין 2 הקודקודים, אם אין צלע המשקל הוא ∞. לכל $k>0$ אנו יודעים שה-$k-1$ חישב את מסלול המינימלי שמכיל את קודקודי הביניים $\{v_1,\ldots.,v_{k-1}\}$ בלבד. לכן, ברצונינו כעת "לכפות" את המסלול המינימלי מ-i ל-j שבהכרח מכיל את קודקוד v_k, בנוסף לקבוצה של קודקודי הביניים. לכן, נמצא את המרחק המינימלי מ-$v_k \to i$ ואת המרחק המינימלי מ-$j \to v_k$ הדבר מחייב מסלול מינימלי שמכיל את v_k וזהו מסלול **יחיד ולא ריבוי אפשרויות**.

3. תיאור של הטבלה, סדר המילוי שלה ואופן חילוץ הפתרון:
נבנה את הטבלה מ$k=0$ עד $k=n$, עבור כל תא (i,j) נרוץ על כל $1 \leqslant i \leqslant n$ עבור כל $1 \leqslant j \leqslant n$, וכך נמלא כל תא בטבלה לפי נוסחת הרקורסיה ולפי התאים שקדמו לתא הנוכחי.
אופן חילוץ הפתרון: עבור כל 2 קודקודים (i,j) מהגדרת הפתרון, התא $f(i,j,n)$ יכיל את המרחק המינימלי עבור המרחק מ-i ל-j
4. ניתוח זמן ריצה: עבור כל תא (i,j) סיבוכיות הזמן תהיה $O(1)$ היות ובכל חישוב תא אנו בודקים נתונים קבועים והאיטרציה הקודמת, זאת אנו עושים עבור n^2 תאים בטבלה באיטרציה ה-k, ומאחר ויש n טבלאות שכאלו סיבוכיות זמן הריצה הכוללת תהיה $O\left(n^3\right)$

מי יותר עדיף:

1. היות וזמן הריצה של בעיה 2 (פלויד ורשל) בסיבוכיות זמן ריצה קטנה יותר, פתרון זה יעיל יותר ■

שאלה 2:

1. נציע את האלגוריתם הבא למציאת תת המחרוזות הפלינדרומים באורך מקסימלי.
2. ראשית, נקבל מחרוזת S, ונבצע עליה $reverse$, נגדיר את מחרוזת זו בתור S'.
3. עתה, קיבלנו 2 מחרוזות, ואנו רוצים לבדוק את התת מחרוזת המקסימלית ביניהן, לכן לבצע את האלגוריתם הדינמי מהתרגול:
   (a) אוסף תתי הבעיות: מציאת תמ"א X^i ו-Y^i לכל $1 \leqslant i \leqslant n$, $1 \leqslant j \leqslant n$.
   (b) נוסחת רקורסיה: נסמן את האורך של תמ"א של X^i ושל Y^i ב- $f(i, j)$:
   $$f(i,j) = \begin{cases} 0 & i = 0 \ \vee j = 0 \\ f(i-1, j-1) + 1 & x_i = y_j \\ max\left\{f(i-1, j), f(i, j-1)\right\} & x_i \neq y_j \end{cases}$$
   (c) מילוי טבלה: נבנה טבלה M בגודל $n \times n$. נמלא את הטבלה לפי נוסחת הרקורסיה הבאה:
      i. נאתחל $k = 0$
      ii. נמלא את $f(k, j)$ מ-$j = 0$ עד $j = n$ אחרי שנסיים את האיטרציה עבור j נעדכן $k := k + 1$
      iii. נעצור כאשר $k = n + 1$.

   שחזור הפתרון:
      i. בזמן מילוי הטבלה נשמור מצביעים אל התא ממנו חושב הפתרון
      ii. נעבור על הטבלה מלמעלה למטה: נתחיל מהתא (n, n), ונלך לפי שמירת המצביעים שהגדרנו. בכל פעם שהתא (i, j) מצביע לתא $(i-1, j-1)$ נוסיף את האות הנוכחית למחרוזת משמאל.
      iii. נחזיר לבסוף את המחרוזת שהתקבלה.

   (d) זמן ריצה: גודל הטבלה הוא $(n+1)(n+1)$. מילוי כל תא לוקח $O(1)$ (בודקים 3 תאים סמוכים). לכן נקבל סה"כ $O\left(n^2\right)$, בנוסף היפוך המחרוזת בסיבוכיות זמן של $O(n)$
4. הוכחת נכונות:
   ברגע שאנחנו מבצעים את ה$reverse$ עבור S', כל איבר יופיע בצורה "שקופה" בתא ה-$n-i$ עבור התא ה-i. לכן, ברגע שמפעילים אלגוריתם למציאת תת סדרה משותפת, תת הסדרה תופיע גם במחרוזת S וגם ב-S', ולכן מהגדרת הפלינדרום (מחרוזת הניתנת לקריאה מהסוף להתחלה ולהפך באותו צורה) נמצא בהכרח פלינדרום, מהאלגוריתם אנו מוצאים את התת מחרוזת הארוכה ביותר, לכן בהכרח גם נמצא באותו אופן את המחרוזת הפלינדרום הארוכה ביותר ■

שאלה 3:

1. נציע אלגוריתם דינמי הפותר את הבעיה:
2. האלגוריתם מקבל קלט גרף ו- $S = s_1 ... s_k$ כך ש $s_i \in \sum$, $\forall i \; 1 \leqslant i \leqslant k$, תווית כלשהי ממרחב תוויות סופי, כמו כן כל צלע מקבלת משקל כלשהו מפונקצית המשקל, ותווית מ- $\sum$.
3. עיבוד מקדים: נעבור על כל צלעות הגרף, ונגדיר רשימה של רשימות, כך שכל רשימה תכיל את הצלעות מאותו תווית.
4. הגדרת הטבלה: יפורט בהמשך
5. תיאור האלגוריתם: נתחיל ממקרה הבסיס ומשם נמשיך:
   (a) מקרה בסיס: במצב זה נגדיר את $S = s_k$, נעבור על כל הקודקדים מהם יוצא צלע עם תווית s_k, נשמור בתא ה-(k, j) (k הוא סוג התווית, j מספר הקודקוד) את הערך המקסימלי מבין הצלעות עם התווית s_k (ייתכן שיש מספר צלעות כאלו).
   (b) בכל איטרציה, נחזור אחורה בקלט הנתון $S = s_{i+1} ... s_k$, ונוסיף לקלט את התווית s_i. עד עתה, האלגוריתם סרק את כל ה $s_{i+1} ... s_k$ ועתה האלגוריתם יחפש את כל הקודקודים בעלי הצלעות s_i, כך שקודקודי היעד מילאו בטבלה את הערך $(i + 1, v)$ כלומר כל הקודקודים שהאלגוריתם התייחס לסימן s_{i+1}. האלגוריתם יסתכל על כל קודקוד מהם יוצא הצלע s_i וישמור את המקסימום מבין כל האפשרויות המתאימות, כלומר:
$$f(j,i) = \begin{cases} 0 & S = \varnothing \\ max\left\{w(e_{(i,v)}) + f(j+1,v)\right\} & else \end{cases}$$
      i. $e_{(i,v)} \in E$ וגם $e_{(i,v)}$ שייך לקבוצת כל הצלעות היוצאות מ-i, ובעלות הסימן $j \in \sum$.
      j- סימן בתוך S.
      i- קודקוד כלשהו בתהליך
      v- קודקוד יעד כלשהו
      $j + 1$- סימן עוקב ב-S.
   (c) כלומר, בכל איטרציה האלגוריתם מוצא את המסלול המקסימלי בגרף כאשר התוויות של צלעות המסלול מתאימות לתת הקלט S.
   (d) באיטרציה האחרונה, האלגוריתם מוצא את המסלול המקסימלי של צלעותיו הן בסדר הקלט S, כלומר בסיום האלגוריתם אם קיים מסלול תקין, הוא יהיה בתא ה- $(1, v_0)$, אחרת לא קיים מסלול כזה.
   (e) ★ - בכל שלב אם לא קיימת צלע מתאימה בתהליך, האלגוריתם יחזיר $-\infty$.
6. תיאור האלגוריתם באופן דינמי:
   (a) ניסוח בהיר של תתי הבעיות: לכל $s_j \in \sum$, נמצא את המסלול המקסימלי עבור צלעות המתאימות ל-$S = s_j ... s_k$ בגרף הנתון
   (b) נוסחת רקורסיה והסבר לבניתה:

נוסחת הרקורסיה: $f(j,i) = \begin{cases} 0 & S = \varnothing \\ max\left\{w(e_{(i,v)}) + f(j+1, v)\right\} & else \end{cases}$

i. $e_{(i,v)} \in E$ וגם $e_{(i,v)}$ שייך לקבוצת כל הצלעות היוצאות מ-i, ובעלות הסימן $j \in \sum$.
j- סימן בתוך S.
i- קודקוד כלשהו בתהליך
v- קודקוד יעד כלשהו
$j+1$- סימן עוקב ב-S.

ii. הסבר לבניה: ההסבר מפורט לעיל בהרחבה, בכל איטרציה נחפש את המסלול המתאים הגדול ביותר, עבור האיטרציה ה-i, נוריד את כל התוויות מ-$s_1, .., s_i$ כך שישאר הקלט $S = s_{i+1} \ldots s_k$.
במקרה הבסיס (כאשר הקלט ריק) נחזיר 0, כדי שלאחר מכן האלגוריתם יחזיר את משקל הצלע המתאימה.

(c) תיאור של הטבלה, סדר המילוי שלה ואופן חילוץ הפתרון:

i. תיאור: הטבלה מכילה $n \times k$ תאים, עבור כל תא $(i,j) : i \in \sum, j \in [n]$, הטבלה תשמור את המרחק המקסימלי מהקודקוד j וסדר התוויות הינו: $s_i \ldots s_k$/

ii. סדר מילוי הטבלה: נתחיל ממקרה הבסיס בו $S = s_k$, לאחר כל השמת התאים הרלוונטים, נעבור לקלט $S = s_{k-1}s_k$ ונחזור חלילה עד $S = s_1, .., s_k$ ובכל שלב ושלב נשים את הערכים המתאימים בתאים בטבלה כמפורט לעיל.

iii. אופן חילוץ הפתרון: אם קיים מסלול תקין, הוא יהיה בתא ה- $(1, v_0)$, ונחזיר את ערכו, אחרת לא קיים מסלול כזה.

(d) ניתוח זמן ריצה: $O(n \cdot k + |V| + |E|) = O(|V| \cdot k + |V| + |E|) = O(k \cdot n + |E|)$:

i. כדי לגשת בצורה יעילה לכל קודקוד לפי הסימן, עברנו על כל הקודקודים והצלעות כדי לשים אותם ברשימה של הרשימות בעלות של $O(|V| + |E|)$

ii. בכל איטרציה, האלגוריתם עובר לכל היותר על כל הקודקודים (בהם יש צלע עם התווית המתאימה), כלומר סיבוכיות זמן של $O(n) = O(|V|)$

iii. כך נבצע k פעמים, כלומר סיבוכיות זמן של $O(n \cdot k) = O(|V| \cdot k)$

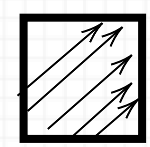
שאלה 4:

1. נציע את האלגוריתם הבא: עבור מטריצה נתונה M

(a) נגדיר טבלה השומרת בכל תא 2 ערכים: כמות ה-1-ים הרצופים מימין ומלמטה לתא הנוכחי, בהתאמה. (a, b), a עבור הערכים מלמטה, b עבור הערכים מימין
כמו כן, עבור כל תא נשמור את הערך 0 עבור גודל המטריצה המקסימלית היוצאת מהתא.

(b) עבוד מטריצה נתונה בגודל $n \times m$ נאתחל את העמודה ה-$m-1$ והשורה ה-$n-1$ לשמור את הערכים: במידה וערך הטבלה הינו 1, נשמור עבור כל תא את הערך $(1,1)$ אחרת את הערך $(0,0)$.

(c) נתחיל למלא את הטבלה באופן הבא: נמלא את הטבלה בצורה אלכסונית לפי האלכסונים המשניים כלומר:



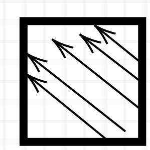
(d) עבור כל תא $$T(i,j) = \begin{cases} (T(i+1,j)[a]+1, T(i,j+1)[b]+1) & M(i,j)=1 \\ (0,0) & M(i,j)=0 \end{cases}$$

הערה: נזכיר $T(i+1,j)[a]$ עבור הערך המתאים לרצף של העמודות (מטה), ו$[b]$ בהתאמה עבור ה"ימינה".

(e) נעבור על כל אלכסון, מהתא $(n-1, m-1)$ עד התא $M(0,0)$ כמפורט לעיל

(f) עתה, לכל תא שמרנו 2 ערכים המהווים את הרצף של ה-1-ים מימין ומלמטה לכל תא

(g) עכשיו, האלגוריתם יעבור על כל אלכסון ראשי,



(h) נתחיל בסריקת האלכסונים מהתא מימין למעלה באיור, כלפי האלכסון התחתון (שמאל למטה)

(i) עבור כל תא, נבדוק את ערכו בטבלה, נחלק למקרים:

i. אם $M(i,j)=0$: נדלג על התא (לא יכול להיות מטריצה ריבועית כלשהי)

ii. אם $a, b \geqslant 1$: נתקדם לתא ה- $(i-1, j-1)$ (באלכסון) ונבדוק האם בתא מתקיים $a, b \geqslant 2$, נמשיך כך כל עוד $(i-k, j-k)$ נמצא בגבולות המטריצה וגם מתקיים כי $a, b \geqslant k$. ברגע ש$a < k \ \vee b < k$ הרצף לא מתקיים, ונשמור בטבלה את המקסימום מבין הערך הנוכחי בטבלה לבין ה-k.

(j) הערה: עבור כל אלכסון נסרוק ונקצה counter=0 כל עוד אנחנו עומדים בתנאי $ii(i)$ (הנ"ל)

נגדיל את ה-counter ב-1, ברגע שנגיע למצב בו התנאי לא מתקיים נעדכן את הטבלה בערך המקסימלי עבור המטריצה, ונאתחל חזרה את ה-counter ל-0, נמשיך עם תהליך סריקת האלכסון עד שנגיע לסופו, ונעבור לאלכסון הבא.

(k) נעבור את הטבלה, נוציא את ה-k המקסימלי המהווה את הגודל המטריצה המקסימלית ונחזיר אותו כמבוקש

1. ניסוח של תתי הבעיות:

לכל (i,j) נחפש את מטריצת ה-1-ים המקסימלית.

2. נוסחת רקורסיה והסבר לבניתה:

$$T(i,j)=\begin{cases}(0,0) & M(i,j)=0\\ (T(i+1,j)[a]+1, T(i,j+1)[b]+1) & if:\ M(i,j)=1\ and\ not:\\ (0, T(i,j+1)[b]+1) & i+1>m\ \wedge j+1\leqslant n\\ (T(i+1,j)[a]+1, 0) & i+1\leqslant m\ \wedge\ j+1>n\\ (1,1) & else\end{cases}$$

עבור כל תא בטבלה הנתונה נסמן בטבלה שבנינו את הערכים $(0,0)$, עבור כל תא עם ערך 1 הפרדנו למקרים כך שנרצה לספור את כמות הערכים הרציפים של 1-ים מימין ומלמטה עבור כל תא, בפירוט מפורט מעלה.

3. תיאור של הטבלה, סדר המילוי שלה ואופן חילוץ הפתרון:

במפורט באלגוריתם, אנו שומרים בכל תא 3 ערכים: את כמות הפעמים ש-1 מופיע ברצף מלמטה ומימין לתא, אותם נאתחל בסריקה על האלכסונים המשניים (מפורט מעלה באלגוריתם בפירוט).

כמו כן, לאחר מכן נשמור בטבלה עבור כל תא את גודל המטריצה המקסימלית היוצאת מימין-מטה.

אופן חילוץ הפתרון: נעבור על כל תא בטבלה ונחזיר את הערך המקסימלי המהווה את גודל המטריצה המקסימלית.

4. ניתוח זמן ריצה:

בניית הטבלה ואיתחולה: $O(nm)$, מעבר על המטריצה באלכסונים (פעמיים)$O(nm)$, מציאת הערך המקסימלי $O(nm)$ לכן סיבוכיות הזמן הריצה הינו $O(nm)$

שאלה 5:

1. נציע את האלגוריתם הבא לפתרון הבעיה:
   (a) נגדיר מטריצה A בגודל $M \times n$ (M עבור הזמנים, n עבור הנתיבים), כך שכל עמודה תהווה הנתיב המתאים, וכל שורה תהווה יחידת הזמן הרלוונטית.
   (b) נעבור על כל זמני כניסה ויציאת רכב ונשים 1 עבור התאים בהם הרכב תופס את הנתיב.
   (c) כמו כן, נגדיר טבלה עבור האלגוריתם, בגודל $M \times n$ נתאחל את העמודה ה-n ל-0, (M עבור הזמנים, n עבור הנתיבים), את השורה האחרונה m נאתחל ל-∞ אם במקום הנ"ל במטריצה יש רכב (תא עם ערך 1), אחרת נאתחל את הערך להיות n פחות העמודה שבה אנו נמצאים (היות ונתון ש M היא יחידת הזמן הגדולה ביותר שבה יהיה רכב בנתיבים, אזי כל יחידת זמן שאחריה תהיה ריקה ולכן יהיה ניתן להגיע באלכסון ישיר ליעד).
   (d) נמלא את הטבלה באופן הבא: נתחיל בתא ה-$(m-1,n-1)$, אם הערך שבמטריצה הוא 1, נערדכן את הערך ל∞, אחרת נחשב את הערך המינימלי מבין:
   $min\left\{\underbrace{(x+1,y)}_{\text{להישאר בנתיב}}, \underbrace{(x+1,y+1)}_{\text{לחזור נתיב אחד אחורה}}, \underbrace{(x+1,y-1)}_{\text{להתקדם לנתיב קדימה}}\right\} + 1$, כך נמשיך עד שנסיים את השורה ה-$m-1$ ונעבור לשורה ה-$m-2$ מהקורדינטה $(m-2,n-1)$ וכך הלאה עד מילוי הטבלה.
   הערה: נתאים את החישובים לגבולות הטבלה, כך שלא נחשב כאשר הערכים יוצאים מגבולותיה
   (e) ★ - אם הגענו לקורדינטה שבה אינדקס העמודה גדול מאינדקס השורה, נגדיר את ערך זה להיות ∞, היות והחתול לא יכול להגיע בזמן פחות מ(i,i) לנתיב האחרון.
   (f) לבסוף נחזיר את הערך שבתא $(1,1)$.
2. <u>ניסוח בהיר של תתי הבעיות:</u> הזמן המינימלי שהחתול יעבור מנתיב i ביחידת זמן j לנתיב האחרון (n)
3. נוסחת רקורסיה והסבר לבניתה:

$$f(i,j) = \begin{cases} f(i,n) \ = \ 0 & j = n \\ f(M,j) = n-j & i = M \ \wedge \ A[M,j] \neq 1 \\ f(i,j) = \infty & A[i,j] = 1 \\ min\left\{\underbrace{f(x+1,y)}_{\text{להישאר בנתיב}}, \underbrace{f(x+1,y+1)}_{\text{לחזור נתיב אחד אחורה}}, \underbrace{f(x+1,y-1)}_{\text{להתקדם לנתיב קדימה}}\right\} + 1 & 1 \leqslant i,j \leqslant n-1 \\ f(i,j) = \infty & j > i \ (\text{עמודות} < \text{שורות}) \end{cases}$$

הסבר לבנייתה מפורט לעיל

4. תיאור של הטבלה, סדר המילוי שלה ואופן חילוץ הפתרון:
   טבלה בגודל $M \times n$ כך שכל תא בטבלה יכיל את הזמן המינימלי שהחתול יעבור מנתיב i ביחידת זמן j לנתיב האחרון (n), סדר המילוי מתבצע לפי האלגוריתם המפורט לעיל (מהיחידת זמן האחרונה (השורה העליונה בטבלה) כלפי מטה, כך שכל שורה ממלאים מימין לשמאל (מהאינדקס העמודה n-1 לאינדקס 0).
   אופן חילוץ הפתרון: ניגש ונחזיר את התא בטבלה ה- $(1,1)$.
5. זמן ריצה: בניית המטריצה: $O(nMN)$ הסבר: נעבור על כל המכוניות עבור כל הנתיבים והזמנים שלהם ונבצע השמה במטריצה במקומות הרלוונטים.
   בניית הטבלה: $O(nM)$ עבור כל תא בטבלה סיבוכיות זמן הריצה היא $O(1)$ (מספר סופי וידוע מראש

של חישובים), כך נבצע כל תא בטבלה ולכן זה סיבוכיות הזמן.
כמו כן, אנו מקצים 2 טבלאות בגודל $M \times n$ לכן סיבוכיות המקום היא $O(nM)$.
לכן אנו עומדים בדרישות השאלה שסיבוכיות הזמן היא:$O(nMN)$ וסיבוכיות המקום הינו: $O(nM)$
במבוקש ■

1. ניסוח בהיר של תתי הבעיות
2. נוסחת רקורסיה והסבר לבניתה
3. תיאור של הטבלה, סדר המילוי שלה ואופן חילוץ הפתרון
4. ניתוח זמן ריצה
5. הוכחת נכונות (אם מבקשים)