<u>שאלה 1 :</u>

מסלולים מזעריים בשריג עם מחירים על קדקודים , האלגוריתם:

לצורך חישוב ומציאת המסלול המזערי נשתמש בשני מערכי עזר (דו ממדיים בגודל n X n)

מערך עזר M(i,j) אשר יכיל את ערכי המסלולים המזעריים עבור תתי-המערכים כל תא M(i,j) יכיל את ערך המסלול המזערי עלו.

מערך עזר S(i,j) אשר יכיל את אינדקס של התא בתת-המערך ממנו 'הגענו' – בכדי שנוכל לשחזר את המסלול המזערי

התוצאה תיוצג ע"י מערך חד-ממדיי P[] שמיצג את המסלול המזערי אשר האינדקס שלו מיצג את השורה במערך C והערך את העמודה של הקדקוד במערך C אשר נמצא במסלול המזערי

<u>בשלב ראשון:</u> נעבור ונאתחל את השורה הראשונה במערכים M בערכי הקדקודים המקוריים ו S באינדקס של כול תא

<u>בשלב השני:</u> נעבור על כל השורות בסדר עולה ועל כול התאים ונחשב את ערכי השורה של M כאשר הנוסחה היא:

$$M[i,j] = C[i,j] + \min(M[i-1,j-1]\ ,M[i-1,j]\ ,M[i-1,j+1])$$

כלומר המינימום מאחד משלושת הקדקודים שיכולים להתחבר לתא המחושב ועד ערך התא

כמו כן נשמור בתא *S[i,j]* את האינדקס של התא המינימלי שבחרנו (*j / j-1 /j+1*)

<u>בשלב השלישי:</u> נחשב את הערך המינימלי בשורה האחרונה *n* , זהו הערך של המסלול המינימלי , נשמור גם את האינדקס של התא בעל הערך הזה (h).

<u>בשלב רביעי:</u> נשחזר את המסלול המזערי, נתחיל בסופו *P[n]* בו נציב את *h* ונעבור על המערך *P* לחור כאשר הנוסחה היא

$$P[i] \leftarrow S[i+1, P[i+1]]$$

כאשר *i* מתחיל מ *n-1* עד 1 .

<u>נכונות:</u> כיוון שהאלגוריתם נכון עבור סריג בעל שורה אחת (מציאת קדקוד המינימום) וברור כי הוא נכון עבור שתי שורות (זו פעולת מיון של שלושה ערכים וחיבורם ושוב קיבלנו שורה אחת) אזי ניתן להראות באינדוקציה לדוגמה כי הוא היה נכון לכול מספר של שורות.

<u>חישוב הסיבוכיות:</u>

בשלב האתחול האלגוריתם מבצע $2n$ פעולות פשוטות --> $O(n)$

בלולה הראשית מבצעים nXn פעולת של – שתי השוואות ושתי הצבות --> $O(n^2)$

בשלב מציאת המינימום מבצעים מעבר בודד על השורה העליונה --> $O(n)$

בשלב שיחזור המסלול מבצעים מעבר על תא בכול שורה --> $O(n)$

כלומר הסיבוכיות היא:

$$\Theta(n^2)$$

<u>האלגוריתם:</u>

Input: Array C[1..n, 1..n] containing s matrix of the Lattice vertex's values

Intermediate : Minimum-path matrix\array M[1..n, 1..n] values , Path matrix\array S[1..n, 1..n] that contain the information regarding the path – the index of the source (from wat cell in the previous row)

Result: P[] array of the vertex indexes of the minimum path , k the minimal path value

MINIMAL_PATH (C[], n)

```
for i ← 1 to n
    S[1, i] ← i          // Initialize the S array first line
    M[1, i] ← C[1, i]
for i ← 2 to n
    for j ← 1 to n
        if M[i − 1, j − 1] ≤ M[i − 1, j]
            if M[i − 1, j − 1] ≤ M[i − 1, j + 1]
                M[i, j] ← C[i, j] + M[i − 1, j − 1]
                S[i, j] ← j − 1
            esle
                M[i, j] ← C[i, j] + M[i − 1, j + 1]
                S[i, j] ← j + 1
        esle
            if M[i − 1, j] ≤ M[i − 1, j + 1]
                M[i, j] ← C[i, j] + M[i − 1, j]
                S[i, j] ← j
            esle
                M[i, j] ← C[i, j] + M[i − 1, j + 1]
                S[i, j] ← j + 1
k ← M[n, 1]
h ← 1
for i ← 1 to n
    if M[n, i] < k
        k ← M[n, i]
        h ← i
P[n] ← h
for i ← n − 1  downto  1
    P[i] ← S[i + 1, P[i + 1]]
return k, P[]
```

<u>שאלה 2 :</u>

נניח כי המחרוזת מיוצגת ע"י מערך P באורך n

נגדיר שני מערכים דו ממדיים שיעזרו בחישוב הפלינדרום המקסימלי במחרוזת

הראשון $M[1..n, 1..n]$ אשר בו נחזיק את גודל הפלינדרום הגדול ביותר בתת המחרוזת מ i ל j (במחרוזת $P[i]...P[j]$) $M[i,j]$

השני $S[1..n, 1..n]$ אשר בו נחזיק את האינדקס ב P של התו הראשון בפלינדרום הגדול ביותר בתת המחרוזת

נגדיר את הפונקציה למילוי המערכים:

$$M[i,j] = \begin{cases} 1 & ,i=j \\ \begin{Bmatrix} 1 & ,P[i] \neq P[j] \\ 2 & ,P[i] = P[j] \end{Bmatrix} & i=j-1 \\ \left.\begin{cases} j-i+1 & , \quad P[i]=P[j] \text{ and } M[i+1,j-1]=j-i-1 \\ M[i+1,j-1] \end{cases}\right\} & i+1<j \end{cases}$$

$$S[i,j] = \begin{cases} i & ,i=j \\ \begin{Bmatrix} i & ,P[i] \neq P[j] \\ i & ,P[i] = P[j] \end{Bmatrix} & i=j-1 \\ \left.\begin{cases} i & , \quad P[i]=P[j] \text{ and } M[i+1,j-1]=j-i-1 \\ S[i+1,j-1] \end{cases}\right\} & i+1<j \end{cases}$$

<u>האלגוריתם:</u>

האלגוריתם יעבור על המערכים M ו S וימלא אותם כך שבסופו של תהליך נוכל לקבל את הערכים של $M[1,n]$ ו $S[1,n]$

שהם הפלינדרום שאנו מחפשים , בכדי לבצע את התהליך באופן יעיל נמלא תחילה את האלכסון המרכזי של המערכים: את $M[i,i]$ נמלא ב 1 – תאים המיצגים פלינדרום באורך 1 , $S[i,i]$ נמלא ב i – תאים המיצגים פלינדרום באורך תו במקום i

לאחר מכן נחשב את האלכסון שמעל האלכסון המרכזי – זהו האלכסון של $M[i,i+1]$ נמשיך ונמלא את האלכסונים עד שנחשב את $M[1,n]$ ו $S[1,n]$ שזו התוצאה הנדרשת.

<u>חישוב הסיבוכיות:</u>

בשלב האתחול האלגוריתם מבצע לולאה של n פעולות פשוטות --> $O(n)$

בלולה הראשית על k מבצעים n פעמיים

לולה שנייה על i מבצעים n פעמיים ($n-k+1$)

בתוך הלולאות מתבצעות שתי השוואות ושתי השמות כלומר $O(1)$ פעולות

כיוון שהלולאות מוכלות אחת בשניים הסיבוכיות היא:

$$\Theta(n^2)$$

האלגוריתם:

Input: Array P[1..n] containing s the character string , n – the number of characters

Intermediate data:

- max palindrome size array M[1..n, 1..n] contain the maximum palindrome size in the sub-string from i to j
- max palindrome start index array S[1..n, 1..n] contain the maximum palindrome starting index in the sub-string from i to j

Result: the size of the maximum palindrome in the string P[] (in M[1,n]) and the first char in the palindrome (P[S[1,n]])

MAX_PALINDROME (P[], n)

$for\ i \leftarrow 1\ to\ n$

$\quad S[i,i] \leftarrow i$ // Initialize the S array first line

$\quad M[i,i] \leftarrow 1$

$for\ k \leftarrow 2\ to\ n$

$\quad for\ i \leftarrow 1\ to\ n-k+1$

$\quad\quad j \leftarrow i+k-1$

$\quad\quad if\ i = j-1$

$\quad\quad\quad if\ P[i] \neq P[j]$

$\quad\quad\quad\quad M[i,j] \leftarrow 1$

$\quad\quad\quad\quad S[i,j] \leftarrow i$

$\quad\quad\quad esle$

$\quad\quad\quad\quad M[i,j] \leftarrow 2$

$\quad\quad\quad\quad S[i,j] \leftarrow i$

$\quad\quad esle$

$\quad\quad\quad if\ (P[i] = P[j])\ and\ (M[i+1,j-1] = j-i-1)$

$\quad\quad\quad\quad M[i,j] \leftarrow j-i+1$

$\quad\quad\quad\quad S[i,j] \leftarrow i$

$\quad\quad\quad esle$

$\quad\quad\quad\quad M[i,j] \leftarrow M[i+1,j-1]$

$\quad\quad\quad\quad S[i,j] \leftarrow S[i+1,j-1]$

Return M[1,n] , S[1,n]

<u>שאלה 4 :</u>

נניח כי סדרת המספרים מיוצגת ע"י מערך A באורך n

נגדיר שני מערכים דו ממדיים שיעזרו בחישוב תת-הסדרה העולה שאורכה המקסימלי

הראשון M[1..n, 1..n] אשר בו נחזיק את אורך תת-הסדרה העולה הגדלה ביותר בתת הסדרה מ i ל j (הסדרה A[i]...A[j]) M[i,j]

השני S[1..n, 1..n] אשר בו נחזיק את האינדקס ב A של האיבר הראשון בתת הסדרה העולה הגדולה ביותר בתת הסדרה i j

נגדיר את הפונקציה למילוי המערכים:

$$M[i,j] = \begin{cases} 1 & ,i = j \\ \begin{Bmatrix} 2 & ,A[i] < A[j] \\ 1\ , & A[i] \geq A[j] \end{Bmatrix} & i = j - 1 \\ \begin{Bmatrix} M[i+1,j] + 1 \quad , & A[i] < A[i+1]\ and\ S[i+1,j] = i+1 \\ M[i,j-1] + 1 \quad , & A[j-1] < A[j]\ and\ S[i,j-1] + M[i,j-1] = j-1 \end{Bmatrix} & i+1 < j \end{cases}$$

$$S[i,j] = \begin{cases} i & ,i = j \\ \begin{Bmatrix} i & ,A[i] < A[j] \\ j\ , & A[i] \geq A[j] \end{Bmatrix} & i = j - 1 \\ \begin{Bmatrix} i \quad , & A[i] < A[i+1]\ and\ S[i+1,j] = i+1 \\ S[i,j-1] \quad , & A[j-1] < A[j]\ and\ S[i,j-1] + M[i,j-1] = j-1 \end{Bmatrix} & i+1 < j \end{cases}$$

<u>האלגוריתם:</u>

האלגוריתם יעבור על המערכים M ו S וימלא אותם כך שבסופו של תהליך נוכל לקבל את הערכים של M[1,n] ו S[1,n]

שהם הערכים של תת-הסדרה העולה שאנו מחפשים , בכדי לבצע את התהליך באופן יעיל נמלא תחילה את האלכסון המרכזי של המערכים: את M[i,i] נמלא ב 1 – תאים המיצגים תת-מערכים באורך 1 , S[i,i] נמלא ב i – תאים המיצגים תת-מערכים באורך 1 במקום ה- i

לאחר מכן נחשב את האלכסון שמעל האלכסון המרכזי – זהו האלכסון של M[i,i+1] נמשיך ונמלא את האלכסונים עד שנחשב את M[1,n] ו S[1,n] שזו התוצאה הנדרשת.

<u>חישוב הסיבוכיות:</u>

בשלב האתחול האלגוריתם מבצע לולאה של n פעולות פשוטות -- > $O(n)$

בלולה הראשית על k מבצעים n פעמיים

לולה שנייה על i מבצעים n פעמיים (n-k+1)

בתוך הלולאות מתבצעות שתיים או שלוש השוואות ושתיים או שלוש השמות כלומר $O(1)$ פעולות

כיוון שהלולאות מוכלות אחת בשניים הסיבוכיות היא:

$$\Theta(n^2)$$

הערה: אני ראה אפשרות לבצע את האלגוריתם בצורה רקורסיבית (הפרד המשול) ב $O(nlogn)$ אך אני לא יודע אם זה נקרא תכנון דינמי ...

ע"י חלוקת המערך לחצי וקריאה רקורסיבית עד ש מגיעים למערך של שניים או אחד ואז ניתן להגדיר את עורך תת-הסדרה העולה ומיקומה , בחזרה מהרקורסיה יש לבצע איחוד של התוצאות (בחירת הסדרה הגדולה או חיבור ביניהן אם ניתן)

כיוון שזמן המיזוג אינו תלוי בגודל , זמן הריצה היה $O(nlogn)$

האלגוריתם:

Input: Array A[1..n] containing s the numbers series , n – the number of numbers in the series

Intermediate data:

- max up series size array M[1..n, 1..n] contain the maximum sb-series size in the up sub- series from i to j
- max up series start index array S[1..n, 1..n] contain the maximum up series starting index in the sub-series from i to j

Result: the size of the maximum up series in the series A[] (in M[1,n]) and the first number in the up series (A[S[1,n]])

MAX_PALINDROME (P[], n)

$for\ i \leftarrow 1\ to\ n$

$S[i,i] \leftarrow i$ // Initialize the S array first line

$M[i,i] \leftarrow 1$

$for\ k \leftarrow 2\ to\ n$

$for\ i \leftarrow 1\ to\ n-k+1$

$j \leftarrow i+k-1$

$if\ i = j-1$

$if\ A[i] \geq A[j]$

$M[i,j] \leftarrow 1$

$S[i,j] \leftarrow j$

$esle$

$M[i,j] \leftarrow 2$

$S[i,j] \leftarrow i$

$esle$

$if\ (A[i] < A[i+1])\ and\ (S[i+1,j] = i+1)$

$M[i,j] \leftarrow M[i+1,j]+1$

$S[i,j] \leftarrow i$

$esle$

$M[i,j] \leftarrow M[i+1,j]$

$S[i,j] \leftarrow S[i+1,j]$

$if\ (A[j-1] < A[j])\ and\ (S[i,j-1] + M[i,j-1] = j-1)$

$M[i,j] \leftarrow M[i,j]+1$

Return M[1,n] , S[1,n]

<u>שאלה 3 א':</u>

"לפי מקורות זרים" (Wikipedia) הפולינומים אשר יקימו את נוסחת הנסיגה:

$$P_{i,j+1}(x) = \frac{q(x)P_{i,j}(x) - r(x)P_{i+1,j+1}(x)}{s(x)}$$

הם:

$$q(x) = x_{j+1} - x \ , \ r(x) = x_i - x \ , \ s(x) = x_{j+1} - x_i$$

נציב: עבור $0 \le i \le j \le n$

$$P_{i,j+1}(x) = \frac{(x_{j+1} - x\ )P_{i,j}(x) - (x_i - x)P_{i+1,j+1}(x)}{x_{j+1} - x_i}$$

בשינוי קטן של הכתיבה נקבל:

$$\begin{cases} P_{i,i}(x) = y_i & , \quad 0 \le i \le n \\ P_{i,j}(x) = \dfrac{(x_j - x\ )P_{i,j-1}(x) + (x - x_i)P_{i+1,j}(x)}{x_j - x_i} & , \ 0 \le i < j \le n \end{cases}$$

<u>שאלה 3 ב':</u> אלגוריתם לחישוב (מקדמי) פולינום האינטרפולציה של n נקודות

<u>האלגוריתם</u>: נכין מערך עזר דו-ממדי M בגודל nXn של מערכים V בגודל n , כל מערך כזה (V) מייצג את המקדמים של הפולינום האינטרפולציה של הנקודות $(x_j, y_j) \ldots\ldots (x_i, y_i)$, הפולינום המבוקש היה בתא M(0,n) .

בכדי להגיע לחישוב תא זה בצורה היעילה ביותר נחשב רק את התאים שמעל לאלכסון M[i,i] , החישוב יתבצע לאורך האלכסונים M[i,i+k] בכדי שהחישוב של $P_{i,j}(x)$ היה מבוסס על פולינומים שחושבו כבר : $P_{i+1,j}(x)$, $P_{i,j-1}(x)$

שלב 1: נמלא את האלכסון המרכזי של המערך כך שהמקדם (0) של הפולינום ב M[i,i] היה y_i

שלב שני: נתחיל למלא את המערך לאורך האלכסונים M[i,i+k] כאשר i מתחיל ב 0 ומסיים ב n-1 , וk מתחיל ב 1 ומסיים ב n .

בכול תא נחשב את הפולינום שלו $P_{i,j}(x)$ חישוב כולל הכפלה של שני פולינומים ידועים (חושבו לפני כן) <u>בקבוע</u> (סיבוכיות $O(n)$) והכפלה של שני פולינומים ב x ('הזזה' $O(n)$) ולבסוף חיבור לפולינום אחד (סיבוכיות $O(n)$) .

החישוב מסתיים כאשר מחשבים את התא האחרון M[0,n] תא זה מכיל את הפולינום המבוקש.

הסיבוכיות: האלגוריתם מבצע שתי לולאות בסדר גודל של n ולכן מבצע $O(n^2)$ פעולת חישוב של תת פולינומים

כיוון שסיבוכיות החישוב של כל תת פולינום היא $O(n)$ הסיבוכיות הכוללת שנקבל תהיה: $\boldsymbol{O(n^3)}$

<u>שאלה 3 ג':</u> $P(x) = x + 2x^2 + 3x^3 + 4x^4$

נציב את הערכים ונחשב:

$$P(-2) = x + 2x^2 + 3x^3 + 4x^4 = -2 + 8 - 24 + 64 = 46$$

$$P(-1) = x + 2x^2 + 3x^3 + 4x^4 = -1 + 2 - 3 + 4 = 2$$

$$P(0) = x + 2x^2 + 3x^3 + 4x^4 = 0$$

$$P(1) = x + 2x^2 + 3x^3 + 4x^4 = 1 + 2 + 3 + 4 = 10$$

$$P(2) = x + 2x^2 + 3x^3 + 4x^4 = 2 + 8 + 24 + 64 = 98$$

נחשב את המערך שלנו : נמלא את האלכסון בערכי הנקודות הידועות

46	-42, -44	0, 19, 21	0, 9, 6, -5	0,1,2,3,4
	2	0, -2	0, 4, 6	0, -7, 6, 11
		0	0, 10	0, -29, 39
			10	-78, 88
				98

נחשב את M[0,1] :

$$P_{0,1}(x) = \frac{(x_j - x\ )P_{i,j-1}(x) + (x - x_i)P_{i+1,j}(x)}{x_j - x_i} = \frac{(-1 - x\ )(46) + (x - -2)(2)}{-1 - -2} = \frac{-44x - 42}{1}$$

נחשב את M[1,2] :

$$P_{1,2}(x) = \frac{(x_2 - x\ )P_{i,j-1}(x) + (x - x_1)P_{i+1,j}(x)}{x_j - x_i} = \frac{(0 - x\ )(2) + (x - -1)(0)}{0 - -1} = \frac{-2x}{1}$$

נחשב את M[2,3] :

$$P_{2,3}(x) = \frac{(x_3 - x\ )P_{i,j-1}(x) + (x - x_i)P_{i+1,j}(x)}{x_j - x_i} = \frac{(1 - x\ )(0) + (x - 0)(10)}{1 - 0} = \frac{10x}{1}$$

נחשב את M[3,4] :

$$P_{3,4}(x) = \frac{(x_4 - x\ )P_{i,j-1}(x) + (x - x_3)P_{i+1,j}(x)}{x_4 - x_3} = \frac{(2 - x\ )(10) + (x - 1)(98)}{2 - 1} = \frac{88x - 78}{1}$$

נחשב את M[0,2] :

$$P_{0,2}(x) = \frac{(x_2 - x\ )P_{0,1}(x) + (x - x_0)P_{1,2}(x)}{x_2 - x_0} = \frac{(0 - x\ )(-42 - 44x) + (x - -2)(-2x)}{0 - -2} = \frac{38x + 42x^2}{2}$$

נחשב את M[1,3] :

$$P_{1,3}(x) = \frac{(x_3 - x\ )P_{1,2}(x) + (x - x_1)P_{2,3}(x)}{x_3 - x_1} = \frac{(1 - x\ )(-2x) + (x - -1)(10x)}{1 - -1} = \frac{8x + 12x^2}{2}$$

נחשב את M[2,4] :

$$P_{2,4}(x) = \frac{(x_4 - x\ )P_{2,3}(x) + (x - x_2)P_{3,4}(x)}{x_4 - x_2} = \frac{(2 - x\ )(10x) + (x - 0)(-78 + 88x)}{2 - -0} = \frac{-58x + 78x^2}{2}$$

נחשב את M[0,3] :

$$P_{0,3}(x) = \frac{(x_3 - x\ )P_{0,2}(x) + (x - x_0)P_{1,3}(x)}{x_3 - x_0} = \frac{(1 - x\ )(19x + 21x^2) + (x - -2)(4x + 6x^2)}{1 - -2}$$
$$= 9x + 6x^2 - 5x^3$$

נחשב את M[1,4] :

$$P_{1,4}(x) = \frac{(x_4 - x\ )P_{1,3}(x) + (x - x_1)P_{2,4}(x)}{x_4 - x_1} = \frac{(2 - x\ )(4x + 6x^2) + (x - -1)(-29x + 39x^2)}{2 - -1}$$
$$= -7x + 6x^2 + 11x^3$$

נחשב את M[0,4] :

$$P_{0,4}(x) = \frac{(x_4 - x\ )P_{0,3}(x) + (x - x_0)P_{1,4}(x)}{x_4 - x_0} = \frac{(2 - x\ )(9x + 6x^2 - 5x^3) + (x - -2)(-7x + 6x^2 + 11x^3)}{2 - -2}$$
$$= \mathbf{x + 2x^2 + 3x^3 + 4x^4}$$

וזהו אכן הפולינום הנדרש!