3, 1, 2, 4

ב"ה

ממן 14 — עשיתי שאלות בסדר!

שאלה 3 -

א - יהיו $i \le j$, $P_{i,j}$ פולינום האינטרפולציה של הנקודות

$$(x_i, y_i), \dots, (x_j, y_j)$$

נגדיר את הפולינומים הבאים:

$$S(x) = x_i - x_{j+1}$$

$$q(x) = x - x_{j+1}$$

$$r(x) = x - x_i$$

$j = 13$ $i = 10$

יהי $i \le k \le j+1$:

$$\frac{q(x_k) P_{i,j}(x_k) - r(x_k) \cdot P_{i+1,j+1}(x_k)}{S(x_k)} = \frac{(x_k - x_{j+1}) \cdot y_k - (x_k - x_i) \cdot P_{i+1,j+1}(x_k)}{x_i - x_{j+1}}$$

כעת נחלק למקרים. • אם $k = i$ אז:

$$= \frac{(x_k - x_{j+1}) \cdot y_k - \overbrace{(x_k - x_k)}^{0} \cdot P_{i+1,j+1}(x_k)}{x_k - x_{j+1}} = \frac{(x_k - x_{j+1}) \cdot y_k}{x_k - x_{j+1}} = y_k = \boxed{P_{i,j+1}(x_k)}$$

אם $k > i$ אז:

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

לפי הגדרה:

$P_{i+1,j+1}(x_k) = y_k$

בס״ד

$$= \frac{\cancel{x_k y_k} - x_{j+1} y_k - \cancel{x_k y_k} + x_i y_k}{x_i - x_{j+1}} = \frac{-x_{j+1} y_k + x_i y_k}{x_i - x_{j+1}} = \frac{(x_i - x_{j+1}) y_k}{(x_i - x_{j+1})} = y_k = \boxed{P_{i,j+1}(x_k)}$$

אם גם $k=j+1$, באופן דומה נקבל כי הביטוי שווה ל- $P_{i,j+1}(x_k)$.

ולכן הפולינום $\frac{q(x)P_{i,j}(x) - r(x)P_{i+1,j+1}(x)}{s(x)}$ שמעלתו קטנה מ- $(j-i+2)$

(מכיוון שאנו מכפילים את $P_{i,j}$ ואת $P_{i+1,j+1}$ בפולינומים ממעלה 1 - $q(x)$ ו- $r(x)$)

כמו כן הפולינום $P_{i,j+1}$ גם ממעלה קטנה מ- $(j-i+2)$

והראנו שהערכים מתלכדים בערכים x_k עבור $i \le k \le j+1$.

כלומר ב- $(j-i+2)$ ערכים שונים ולכן לפי המשפט היסודי של האלגברה

להם אותו פולינום. והפולינומים $r(x)$, $q(x)$, $s(x)$ אכן מוצדקים

כמבוקש □

ב- תהיו $(x_1,y_1),\dots,(x_n,y_n)$ רשימת הנקודות. ברור שלכל $1 \le k \le n$

מתקיים: $P_{k,k} = y_k$ שכן הישר שעובר בנקודה (x_k,y_k)

הינו הישר האופקי y_k. ולכן זהו הפולינום $P_{k,k}$. כמו כן לכל $1 \le i < j < n$

לפי הנוסחה מתקיים:

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

עבור הפולינומים $s(x), q(x), r(x)$ שהגדרנו.

בס"ד

ולכן הוכחנו את הטענה הבאה:

$$P_{i,j} = \begin{cases} y_i & i=j \\ \dfrac{q(x)P_{i,j-1}(x) - r(x)P_{i+1,j}(x)}{s(x)} & i<j \end{cases}$$

כעת נתאר את האלגוריתם:

```
Interpolation( V̄ = (x_1,y_1),...,(x_n,y_n) ):

Init  M[n,n]
For i ← 1 to n:
      M[i,i] ← y_i

For i ← 2 to n:
      For j ← i+1 to n:
            Let: s(x) ← x_i - x_{j+1} , q(x) ← x - x_{j+1} , r(x) ← x - x_i
            M[i,j] ← (q(x)·M[i,j-1] - r(x)·M[i+1,j]) / s(x)
return  M[1,n]
```

$$M[i,j] \leftarrow \frac{q(x)\cdot M[i,j-1] - r(x)\cdot M[i+1,j]}{s(x)}$$

הערה: התא $M[i,j]$ מכיל את הפולינום $P_{i,j}$.

הוכחה: באינדוקציה על $l=i-j$ כאשר $1\le i,j\le n$.

בסיס: כאשר $l=0 \Leftarrow i=j$. לפי האלגוריתם $M[i,j]=y_j=\underset{i=j}{P_{i,j}}$ כנדרש.

נניח שנכון לכל $0 < l < n$ כאשר $l = j - i > 0$

כלומר $M[i,j] = P_{i,j}$. $1 \le j,i \le n$.

נתבונן ב- $l+1$ $\Leftarrow$ לכן $j - i > 1$ $\Leftarrow$ $j > i+1$

$$M[i,j] = \frac{q(x)\cdot M[i,j-1] - r(x)\cdot M[i+1,j]}{S(x)} = \frac{q(x)\cdot P_{i,j-1} - r(x)\cdot P_{i+1,j}}{S(x)} = P_{i,j}$$

לפי האלגוריתם כאשר $j \geq i+1$ אנו מבצעים את הלולאה הפנימית

$j-(i+1) = j-i-1 = \underline{l-1}$

$j-1-i = j-i-1 = \underline{\underline{l-1}}$

ולכן ניתן להשתמש בהנחת הא'

לפי נוסחת הנסיגה

והטענה הושלמה.

כמו כן, בכל שלב שבו אנו מחשבים את $M[i,j]$, אנו כבר יודעים את הערכים $M[i,j-1]$, $M[i+1,j]$ שכן אנו מתקדמים לאורך מילוי הטבלה מהאלכסון הראשי עד לתא הימני עליון.

זמן ריצה

ניתן לראות בבירור שהלולאה המקוננת הינה $\Theta(n^2)$ איטרציות. בכל איטרציה יש כפל, חיבור וחילוק של פולינומים בגודל $\Theta(n)$ ולכן נקבל שסה"כ הסיבוכיות היא $\Theta(n^3)$ $\square$

בס"ד

ב-

$-2, -1, 0, 1, 2$ $\qquad P(x) = x + 2x^2 + 3x^3 + 4x^4$

$$P(-2) = -2 + 2\cdot(-2)^2 + 3\cdot(-2)^3 + 4\cdot(-2)^4 = -2 + 8 + -24 + 64 =$$

$$= 46 \to \begin{matrix} x_1, y_1 \\ (-2, 46) \end{matrix}$$

$$P(-1) = -1 + 2 + (-3) + 4 = 1 - 3 + 4 = 2 \to \begin{matrix} x_2, y_2 \\ (-1, 2) \end{matrix}$$

$$P(0) = 0 \to \begin{matrix} x_3, y_3 \\ (0, 0) \end{matrix}$$

$$P(1) = 1 + 2 + 3 + 4 = 10 \to \begin{matrix} x_4, y_4 \\ (1, 10) \end{matrix}$$

$$P(2) = 2 + 8 + 24 + 64 = 98 \to \begin{matrix} x_5, y_5 \\ (2, 98) \end{matrix}$$

$P_{1,1} = 46 \qquad P_{2,2} = 2 \qquad P_{3,3} = 0 \qquad P_{4,4} = 10 \qquad P_{5,5} = 98$

$$P_{1,2} = \frac{(x-(-1))\cdot 46 - (x-(-2))\,2}{-2-(-1)} = \frac{46x + 46 - 2x - 4}{-1} = \frac{44x + 42}{-1} = -44x - 42$$

$$P_{2,3} = \frac{(x-0)\cdot 2 - (x-(-1))\,0}{-1-0} = \frac{2x}{-1} = -2x$$

Let : $s(x) \leftarrow x_i - x_{j+1}$

$, q(x) \leftarrow x - x_{j+1} ,$

$r(x) \leftarrow x - x_i$

$$P_{3,4} = \frac{(x-1)\cdot 0 - (x-0)\cdot 10}{0-1} = \frac{-10x}{-1} = 10x$$

$$P_{4,5} = \frac{(x-2)\cdot 10 - (x-1)98}{1-2} = \frac{10x - 20 - 98x + 98}{-1} = \frac{-88x + 78}{-1} = 88x - 78$$

ב ס"ק

$$P_{1,3} = \frac{(x-0)(-44x-42)-(x-(-2))(-2x)}{-2-0} = \frac{-44x^2-42x+2x^2+4x}{-2}$$

$$= \frac{-42x^2-38x}{-2} = 21x^2+19x$$

$$P_{2,4} = \frac{(x-1)(-2x)-(x-(-1))(10x)}{-1-(1)} = \frac{-2x^2+2x-10x^2-10x}{-2} = \frac{-12x^2-8x}{-2} = 6x^2+4x$$

$$P_{3,5} = \frac{(x-2)(10x)-(x)(88x-78)}{0-2} = \frac{10x^2-20x-88x^2+78x}{-2} = \frac{-78x^2+58x}{-2}$$

$$39x^2-29x$$

$$P_{1,4} = \frac{(x-1)(21x^2+19x)-(x-(-2))(6x^2+4x)}{-2-(1)} = \frac{21x^3+19x^2-21x^2-19x-6x^3-4x^2-12x^2-8x}{-3}$$

$$= \frac{15x^3-18x^2-27x}{-3} = -5x^3+6x^2+9x$$

$$P_{2,5} = \frac{(x-2)(6x^2+4x)-(x-(-1))(39x^2-29x)}{-1-2} =$$

$$= \frac{6x^3+4x^2-12x^2-8x-39x^3+29x^2-39x^2+29x}{-3} = \frac{-33x^3-18x^2+21x}{-3} = 11x^3+6x^2-7x$$

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

$$\frac{-5x^4+6x^3+9x^2+10x^3-12x^2-18x-11x^4-6x^3+7x^2-22x^3-12x^2+14x}{-4} = \frac{-16x^4-12x^3-8x^2-4x}{-4}$$

$$= \boxed{4x^4+3x^3+2x^2+x}$$

מ"ש

נדב, אני גאה בך! כל הכבוד, תשובה מושלמת!

בס"ד

שאלה 1 -

רעיון: נשתמש בתכנון דינאמי ע"י כך שנפרק מסלול קל לתתי הניתוחי למסלולים הקלים ביותר של התאים המותרים בעמודה הבאה. נגדיר OPT רקורסיבי, נוכיח טענה לגביו שהאלגוריתם יתבסס עליו. האלגוריתם יקבע ערכים למסלולים מהסוף להתחלה. (עמודה ימנית -> שמאלית) - כדי שיהיו לנו בכל שלב את הערכים הנדרשים. כמו כן נגדיר לכל תא את הערך $P(i,j)$ שיסמן מאיזה הגענו אליו "כשהלכנו אחורה" וכך תהיה לנו דרך לעקוב אחרי המסלול הקל ביותר.

הקדמה להוכחת האלגוריתם:

נגדיר:

$OPT(i,j)$ - ערך המסלול המינ' הקל ביותר מהקורדינטה (i,j) עד צומת בעמודה הימנית (עמודה n).

טענה:

$$
\circledast \quad OPT(i,j)=\begin{cases} c(n,j) & i=n \\ \min\{OPT(i+1,j-1),OPT(i+1,j),OPT(i+1,j+1)\}+c(i,j) & 1<j<n,\ 1\le i<n \\ \min\{OPT(i+1,j),OPT(i+1,j+1)\}+c(i,j) & j=1,\ 1\le i<n \\ \min\{OPT(i+1,j),OPT(i+1,j-1)\}+c(i,j) & j=n \end{cases}
$$

הערה: נשים לב שכיסינו את כל האפשרויות עבור ערכי i ו-j כאשר $1\le j,i\le n$.

הוכחת הטענה:

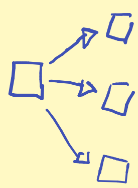
נוכיח באינדוקציה על l כאשר l מוגדר להיות "כמה עמודות נשאר כדי להגיע לעמודה הימנית". כלומר $l=n-i$ כאשר $1\le i<n$. הטענה הינה למעשה $\circledast$.

כאשר $l=0$:

יהי $1\le j\le n$ מספר שורה כלשהו. $l=0 \Leftarrow n-i=0 \Leftarrow \boxed{i=n}$ ולכן אני בעמודה הימנית וערך המסלול הקל ביותר בהו למעשה ערך הוובר שאנו נמצאים בו, כלומר $c(j,n)$ $OPT(i,j)$

נניח שהטענה נכונה עבור $0<l<n-1$. נוכיח שהטענה מתקיימת עבור $l=n-1$:

יהי $1\le j\le n$ מספר שורה.
אם $1<j<n$: אנו נמצאים במצב שאנו יכולים ללכת לעבור הצד הימני לשלושה תאים שונים



נרצה ללכת לתא שערך המסלול ממנו לעמודה הימנית ביותר הינו הקטן ביותר. כלומר נרצה לעבור לתאי $(i+1,j)$ או $(i+1,j-1)$ או $(i+1,j+1)$

מכיוון שתאים אלו נמצאים בעמודה הבאה, לפי הנחת האינדוקציה ($l'=l-1$)

המרחק של התאים הימניים מהסוף

ערך המסלול הזול בתאים אלו הם:

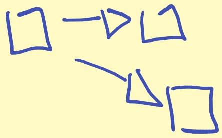
$$OPT(i+1,j-1), OPT(i+1,j), OPT(i+1,j+1)$$

ולכן הערך של המסלול הזול ביותר מהתא שאינו נמצאים הינו:

$$OPT(i,j) = \min\{OPT(i+1,j-1), OPT(i+1,j), OPT(i+1,j+1)\} + \underbrace{c(i,j)}_{\text{עלות התא הנוכחי}}$$

אם $j=1$:

אין בשורה הראשונה ולכן אופציות ההתקדמות הן:



ולכן נרצה להתקדם לתא שמימין והולך לעמודה הימנית ביותר - העלות נמוכה ביותר

כלומר לתאים: $(i+1,j)$ או $(i+1,j+1)$. לפי הנחת האינדוקציה ($l'=l-1$):

ערך המסלול הקל המתאים לתאים אלו בהתאמה הוא $OPT(i+1,j), OPT(i+1,j+1)$.

ולכן ערך המסלול הקל מהתא הנוכחי הוא:

$$OPT(i,j) = \min\{OPT(i+1,j), OPT(i+1,j+1)\} + c(i,j)$$

כאשר $j=n$ מראים באופן דומה כי

$$OPT(i,j) = \min\{OPT(i+1,j), OPT(i+1,j-1)\} + c(i,j)$$

והאינדוקציה הושלמה ולכן מתקיימת הטענה (*).

בס״ד

כעת, יש לנו למעשה את התשובה הבאה:

לכל $1 \le i,j \le n$, $OPT(i,j)$ מכיל את הערך של המסלול הקל ביותר מהתא i,j לעבר העמודה n (כאשר ההתקדמות חוקית!).

כל שנותר לנו על מנת לדעת את ערך המסלול הקל ביותר מעמודה שמאלית לימנית זה לעבור על $OPT(1,j)$ כאשר $1 \le j \le n$ ולבחור את התא עם הערך המינימלי ואנו יודעים את ערך המסלול הקל.

על מנת לדעת מהו המסלול עצמו, נגדיר:

$P(i,j)$ - התא הבא בעמודה הבאה במסלול הקל ביותר (כאשר $P(n,j)=null$ מכיוון שזהו התא האחרון).

כדי לדעת מהו המסלול הקל ביותר, נבחר $OPT(1,j)$ שמקיים: $\min_{1 \le j \le n}(OPT(1,j))$ ונרוץ ע״י המצביעים $P(1,j)$ עד שנגיע ל- null וכך נדע את המסלול.

בהגדרת האלגוריתם, נראה איך $P(i,j)$ נבנה. תיאור האלגוריתם בעמוד הבא.

תיאור האלגוריתם:

שריג   פונקציית משקל

Find Path (A, c):

```
Initialize M[n,j] ← c(n,j)   for each j=1,2,...n
Initialize P[n,j] ← null     for each j=1,2,...n
For i←n-1 down to 1 :
    For j←n down to 1:
        if 1<j<n:
            M[i,j] ← min{M[i+1,j-1], M[i+1,j], M[i+1,j+1]} + c(i,j)
            P[i,j] ← cell with minimun cost path from i+1
        if j=1 :
            M[i,j] ← min{M[i+1,j], M[i+1,j+1]} + c(i,j)
            P[i,j] ← cell with minimun cost path from i+1
        if j=n :
            M[i,j] ← min{M[i+1,j], M[i+1,j-1]} + c(i,j)
            P[i,j] ← cell with minimun cost path from i+1
(x,y) ← Find Minimum cell in M[1,j] for each j=1, ... ,n
cell ← (x,y)
While cell is not null do :
        Print cell
        cell ← p(cell)
```

הוכחת נכונות:

תחילה, נשים לב שאנו ממלאים את כל העמודה $i=n$ בהתחלה ולאחר מכן לכל $1 \le j \le n-1$, אנו מחשבים את $M[i,j]$ מאמץ התאים שנמצאו בעמודה גבוהה יותר שכבר חושבה באיטרציה הקודמת. ולכן בכל שלב באלגוריתם יש לנו את הערך המבוקש.

l זה למעשה כמה עמודות נותרו לימין.

<u>טענה:</u> <u>$M[i,j] = OPT(i,j)$. הוכחה:</u>

נניח כי $M[i,j] = OPT(i,j)$. באמצעות אינדוקציה על $l=n-i$, כאשר i זהו העמודה שאנו ממלאים וזהו. <u>כאשר $l=0$</u> $\Leftarrow$ $i=n$. לפי האלגוריתם $M[n,j]=C(n,j)$ $1\le j\le n$ שזהו הערך $OPT(n,j)$ כאשר $i=n$.

נניח שמתקיים עבור $l<m\le n-1$ (כלומר $n-i<m-1$ $\;\; i>n-m+1$) $M[i,j]=OPT[i,j]$ $\forall 1\le j\le n$

נוכיח עבור $l=m$ $\Leftarrow$ ($\Leftarrow$ $n-i=m$ $\Leftarrow$ $\underline{i=n-m}$) מתקיים $M[i,j]=OPT(i,j)$ $\forall 1\le j\le n$

נשים לב שערכו של i קטן לעומת הנחת האינדוקציה ולכן יש לנו כבר את כל העמודה $i+1$ מלואה כבר בטבלה M. לפי האלגוריתם:

<u>if $1<j<n$:</u>
$M[i,j] \leftarrow \min\{M[i+1,j-1], M[i+1,j], M[i+1,j+1]\} + c(i,j)$
<u>if $j=1$:</u>
$M[i,j] \leftarrow \min\{M[i+1,j], M[i+1,j+1]\} + c(i,j)$
<u>if $j=n$:</u>
$M[i,j] \leftarrow \min\{M[i+1,j], M[i+1,j-1]\} + c(i,j)$

$M[i,j] \triangleleft\!\!-$

$= \uparrow$ הנחת האינדוקציה

<u>if $1<j<n$:</u>
$M[i,j] \leftarrow \min\{OPT(i+1,j-1), OPT(i+1,j), OPT(i+1,j+1)\} + c(i,j)$
$=$ <u>if $j=1$:</u>
$M[i,j] \leftarrow \min\{OPT(i+1,j), OPT(i+1,j+1)\} + c(i,j)$
<u>if $j=n$:</u>
$M[i,j] \leftarrow \min\{OPT(i+1,j), OPT(i+1,j-1)\} + c(i,j)$

$= \uparrow$ לפי טענה (*) $\;\; OPT(i,j)$

בס"ד

ולכן האינדוקציה הושלמה.

לכן $M[i,j] = OPT(i,j)$. כעת כל שעלינו לעשות זה למצוא את המינימום:

$$\min_{1 \le j \le n} \{ M[1,j] \}$$

וזהו למעשה הערך של המסלול הקל ביותר במטריצה. כמו כן האלגוריתם שומר בטבלה P את התאים בעמודה הבאה במסלול הקל ביותר. ולכן לאחר שמצאנו את התא שממנו מתחילים את המסלול הקל ביותר, כל שעלינו לעשות זה לעבור בין המצביעים החל מהתא הראשון, עד שנגיע ל- NULL. (כלומר אם נניח בשלילה ש- $P(i,j)$ אינו התא הבא במסלול הקל ביותר, נקבל שיש תא חוקי X אחר שממנו ממשיך המסלול החוקי הקל ביותר. כלומר נקבל כי $M[X] < M[P(i,j)]$ אבל זה כמובן סתירה, כי האלגוריתם נכון ונבדק <u>מינימום</u> כאשר הוא מעדכן את $P[i,j]$).

סיבוכיות:

מבחינת מקום - שתי טבלאות בגודל $n \times n$ $\leftarrow$ $\Theta(n^2)$.

זמן - הלולאות המקוננות רצות ב- $\Theta(n^2)$ כאשר כל איטרציה הינה בזמן <u>קבוע</u>.

מציאת המינימום של העמודה הראשונה - $\Theta(n)$.

הדפסת המסלול הינה $\Theta(n)$, מכיוון שלכל $A(i,j)$ מתקיים שהעוקב שלו $P(A(i,j))$ נמצא בעמודה $i+1$ ולכן ישנם $\Theta(n)$ עוקבים עד לערך NULL. ולכן סה"כ $\Theta(n^2)$

שאלה 2 -

רעיון האלגוריתם:

בתחילה נמיין את כל התיבות מהתיבה הרחבה ביותר עד לתיבה הכי פחות רחבה. נגדיר OPT(i) כערך של הגובה של המגדל היציב הגבוה ביותר שניתן לבנות מ- $1,\dots,i$ התיבות הרחבות ביותר. בעזרת OPT

לכן, נייצר אלגוריתם תכנון דינאמי, שבו נמלא מערך חד מימדי בערכים של הגבהים הגבוהים ביותר של מגדל יציב, כאשר התא ה-i במערך מייצג את ערכו של המגדל היציב הגבוה ביותר שנבנה בעזרת התיבות $1,\dots,i$ הרחבות.

לאחר מכן, נשחזר את התיבות ע"י סריקה לאחור של המערך שמילאנו,

ונקבל את התיבות.

המשך בדף הבא!

בס״ד

נסמן:

B_1 התיבה בעלת הרוחב הגדול ביותר

B_n התיבה בעלת הרוחב הקטן ביותר. כלומר:

$W(B_n) < W(B_{n-1}) < \dots < W(B_1)$

נגדיר:

$OPT(i)$ = ערך של הגובה המירבי של מגדל יציב המורכב מתיבות $B_1, \dots, B_n$ הרחבות ביותר

$1 \le i \le n$

טענה:

$$OPT(i) = \begin{cases} h(B_1) & \boxed{i=1} \\ \max\left\{ OPT(i-1),\ h(B_i) + \max\limits_{\substack{1 \le k \le i-1 \\ \text{and} \\ l(B_k) > l(B_i)}} \{ OPT(k) \} \right\} & \boxed{1 < i \le n} \end{cases}$$

הוכחת הטענה:

באינדוקציה על i כאשר i מ״צ את המיקום של התיבה ה-i-ית הכי רחבה.

כאשר $i=1$, המגדל היציב הכי גבוה שניתן לבנות מ-$\{B_1\}$ זה המגדל שבנוי מ-B_1 בלבד. ולכן גובהו $h(B_1)$. כלומר $OPT(1)=h(B_1)$ אכן הטענה נכונה.

נניח שעבור $i<n$ מתקיימת הטענה, נוכיח עבור $i+1 \le n$:

נתבונן בסידרה: $\underset{w_1 > w_2 > \dots > w_{i+1}}{(B_1, B_2, \dots, B_i, B_{i+1})}$ יהי Θ קבוצת התיבות שמרכיבה את המגדל היציב שתיבותיו מהסדרה ↑. כעת ישנן שתי אופציות. 1) $B_{i+1} \in \Theta$ 2) $B_{i+1} \notin \Theta$

בס״ד

אם $B_{i+1} \in \Theta$, ברור שהתיבה B_{i+1} בראש המגדל, שכן היא התיבה עם הרוחב הקטן ביותר בסדרה $(B_1,\dots,B_{i+1})$ והמגדל הינו <u>יציב</u>.

כמו כן כל תיבה "מתחת" מחויבת להיות באורך גדול יותר מאשר $l(B_{i+1})$.

ולכן נקבל שהגובה של המגדל הינו $h(B_{i+1})$ ועוד <u>הגובה של המגדל היציב הגבוה ביותר שניתן לבנות מהתיבות $(B_1,\dots,B_i)$ שבראש מגדל זה ישנה תיבה B_j המקיימת: $l(B_j) > l(B_{i+1})$ (שכן זה מגדל יציב).</u>

$1 \le j \le i$

אך לפי הנחת האינדוקציה, גובה של מגדל יציב זה הינו: $OPT(B_j)$ (שכן $j \le i$).

$1 \le j \le i$

כלומר:

$$OPT(B_{i+1}) = h(B_{i+1}) + \max_{\substack{1 \le j \le i \\ \text{and} \\ l(B_j) > l(B_{i+1})}} \{OPT(B_j)\}$$

אם $B_{i+1} \notin \Theta$ אזי ברור שהמגדל היציב בנוי מהתיבות $(B_1,\dots,B_i)$

אך ערכו של גובה המגדל היציב הגבוה ביותר לפי הנחת האינדוקציה הוא $OPT(B_i)$.

ולכן:

$$opt(B_i) = \max\Big\{ opt(i-1),\ h(B_i) + \max_{\substack{1 \le k \le i-1 \\ \text{and} \\ l(B_k) > l(B_i)}} \{opt(k)\} \Big\}$$

והאינדוקציה הושלמה והטענה נכונה.

בס״ד

נאמר שהגדרנו את ה- OPT, האלגוריתם נבנה באופן פשוט:

קבוצת התיבות

```
Tower( B ):

 Sort B by width in desc order
 Let M[0...n] be the memoization table
 M[1] <- h(B1)
 for i <- 2 to n:
      M[i] <- max { M[i-1],  max      { h(bi) + M[j]   where l(bj) < l(bi) } }
                            1<=j<i

 // Print the Boxes:
 i <- n
 while ( i > 0 ) do:
      if M[i] = M[i-1]
           then  i <- i-1
      else
           print bi
           Scan M[] back until find j  such that: M[j] = M[i] - h(bi)
           i <- j
 return M[n]
```

הוכחת נכונות:

נוכיח כי לכל $1 \le i \le n$: $M[i] = OPT(i)$.

ע״י אינדוקציה על האינדקס.

בסיס $j=1$: לפי האלג׳ $M[1] = h(b_1) = OPT(1)$

המשך הבא:

בס"ד

נניח שלכל $i<n$: $M[i]=OPT(i)$

לכן עבור $i+1$:

$$M[i+1] \xleftarrow{\text{לפי האלג'}} \max\Big\{m[i],\ \max_{1\le j<i+1}\{h(b_{i+1})+m[j] \ \text{ where }\ l(b_j)<l(b_{i+1})\}\Big\}$$

$$\underset{\text{הנחת האינדוקציה}}{=} \max\Big\{opt(i),\ h(B_{i+1})+\max_{\substack{1\le k\le i \\ \text{and} \\ l(B_k)>l(B_{i+1})}}\{opt(k)\}\Big\} = OPT(i+1)$$

והאינדוקציה הושלמה.

כמו כן, בכל שלב של הלולאה, החישוב של $M[i]$ מתבסס על $M[j]$ כאשר $j<i$, מכיוון שבתחילת האלג' איתחלנו את $M[1]$ לכן לאורך כל הלולאה אני מחשבים את $M[i]$ על סמך תאים שכבר חישבנו.

כמו כן, השיפור שאנו מבצעים נכון מכיוון שהוכחנו את הגדרת ה-OPT (ניתן לראות בקלות מה הערך של OPT לפי ערכים קודמים).

ניתוח זמן

מיון - $\Theta(n\lg n)$

לולאות מילוי הטבלה - ישנן $\Theta(n)$ איטרציות. כל איטרציה מחפשת מקסימום מתוך $\Theta(i)$ ערכים שמילאנו עד כה. ולכן הלולאה עולה $\Theta(n^2)$.

בס"ד

<u>שיחזור התיבות</u>- לולאות ה-while לוקחות לכל היותר n איטרציות

מכיוון ש-i יורד בכל איטרציה לפחות בערך אחד.

כמו כן "יתכן" ש-i "ילך יותר "רחוק" כאשר

מחפשים את הערך המבוקש.

ככה ש"ילך אחורה" רק יהיו פחות איטרציות בעתיד.

$\Theta(n)$

סה"כ $\Theta(n^2)$ □

15/20

שאלה 4-

א) האלגוריתם מחשב לכל $v \in V$ את משקל המסלול הקל ביותר מהצומת r עד V.

נוכיח באינדוקציה על מספר האיטרציות של הלולאה החיצונית את הטענה הבאה:

יהי l מספר האיטרציות. בסוף האיטרציה ה-l, $A[v]$ מכיל את משקלו של המסלול הקל ביותר המורכב לכל היותר l קשתות מהצומת r עד לצומת V.

עבור $l=0$:

האלגוריתם מעדכן $A[v=r] \leftarrow 0$ וזהו אכן משקלו של המסלול הקל ביותר המכיל לכל היותר $l=0$ קשתות. כמו כן לכל $v \neq r$ האלגוריתם מעדכן $A[v] \leftarrow \infty$ שזהו ערכו של "המסלול הריק" (לכל היותר $l=0$ קשתות).

נניח שנכון עבור $k<l$ נוכיח שמתקיים עבור l.

יהי $v \in V$.

יהי $P_{r,v}$ המסלול הקל ביותר בין r ל-v שאורכו לכל היותר l (יתכן שלא קיים מסלול כזה ואז אורכו ∞)

אם $|P_{r,v}| < l$ אזי לפי הנחת האינדוקציה באיטרציה קודמת התעדכן $A[v]$ לערך של המשקל של המסלול $P_{r,v}$ (ולו השתנה כי זהו הקל ביותר). וכמובן שהתנאי $A[v] > A[u] + c_e$ לא יתקיים ולכן $A[v]$ לא משתנה.

דרוש הסבר מדוע התנאי לא יתקיים. זה בהחלט לא ברור.

אם $|P_{r,v}| = l$: נתבונן בצומת הלפני אחרונה במסלול זה.

בס״ד

נסמנה ב-u. המסלול $P_{r,v}\setminus\{u,v\}$ הינו בן $l-1$ הקשתות ולכן
עפ״י הנחת האינדוקציה $A[u]$ מכיל את משקל מסלול זה (שהרי
המסלול $P_{r,v}\setminus\{u,v\}$ הינו המסלול הקל ביותר בין r לבין u שאורכו
קל אורך מושר $l-1$ קשתות. נניח בשלילה שלא,
כלומר קיים מסלול $p^*\neq P_{r,v}\setminus\{u,v\}$ קל יותר. אזי המסלול $p^*\cup\{u,v\}$
הינו קל יותר מאשר $P_{r,v}$. בסתירה להנחה ש-$P_{r,v}$ המסלול הקל ביותר באורך l.
כמו כן ברור שלא קיים מסלול קל יותר בין r לבין u עם פחות הקשתות
מאשר $l-1$ כי אז היינו שוב מגיעים לסתירה להנחה שהמסלול הקל
ביותר בין r לבין v הינו מאורך l.)

כעת, מכיוון ש-$A[v]$ כרגע (לפני בדיקת התנאי) מכיל משקל של מסלול קל ביותר באורך
קטן ממש מ-l, ברור שמתקיים $A[v] > A[u]+C_e$ שכן

אני חולק עלייך ואני חושב שזה לא ברור שזה מתקיים.

לפי הנחתנו המסלול הקצר[a] שאורכו קטן שווה ל-l הוא בדיוק באורך
l והערך של מסלול זה הוא $A[u]+C_e$ ולכן $A[v] > A[u]+C_e$
ואז מתקיים העדכון $A[v]\leftarrow A[u]+C_e$ והטענה מתקיימת.

כמו כן אם לא קיים מסלול שאורכו קטן שווה ל-l אז נחלק
לשני מקרים:

- אם אין ל-v שכנים ברור שיתקיים $A[v]\leftarrow\infty$ שכן
עפ״י האלגוריתם אין קשת שתכניס אותנו לתנאי.

- אם יש ל-v שכן u, ברור שכל מסלול בין r לבין
u יהיה באורך לכל הפחות l. אבל באינדוקציה

הקודמות (שקטנות ממש מ-l) לפי הנחת האינדוקציה מתקיים $A[u]\leftarrow\infty$ (שכן זהו ערך המשקל כאשר אין מסלולים באורך שקטן מ-$(l-1)$). לכן אורכו של $A[v]$ לא ישתנה והוא ישאר כמו האתחול הראשוני שהוא ∞ וזה תואם משקלו של המסלול הקל ביותר שאורכו קטן מ-l (שהרי אין כזה).

והאינדוקציה הושלמה.

<u>טענה</u>: יהי $v\in V$. אם קיים מסלול בין s לבין v אזי קיים מסלול קל ביותר המכיל לכל היותר $n-1$ קשתות (<u>בהנחה</u> קיים מסלול קל ביותר)

<u>הוכחה</u>: נניח שקיים מסלול בין s לבין v. נניח בשלילה שהמסלול הקל ביותר מכיל לפחות n קשתות. $\Rightarrow$ ישנו קודקוד u שמופיע פעמיים במסלול זה (שכן ישנם n קודקודים, ובמסלול כזה מופיעים לפחות $n+1$ קודקודים) ולכן קיים מעגל C במסלול. מכיוון שהקשתות שוליות משקל אי שלילי, לכן ניתן "למחוק" מעגל זה ולהגיע למסלול עם משקל קטן שווה וגם פחות קשתות. סתירה. ולכן הטענה נכונה.

כעת, לפי הטענה שהוכחנו באינדוקציה ברור שבסוף הצעד לאיטרציה n, אזי כבר לכל $v\in V$ מעודכן משקלו של מסלול קל ביותר שמכיל לכל היותר $n-1$ הקשתות ואם אין מסלול כזה, $A[v]=\infty$. לכן באיטרציה ה-n-ית לא יתבצע שום עדכון

בס״ד

ולכן הוכחנו למעשה שהאלגוריתם מעדכן ב- $A[v]$ את משקל המסלול הקל ביותר מ- r אל v (וזה ∞ אם לא קיים מסלול).

ב' וג' מושלם

ב) בסעיף לעיל הראינו כי $B(n)=n$.

נגדיר גרף בעל n קודקודים שעבורו יתבצעו n איטרציות:

$$G_n = (V, E) \quad \text{where:} \quad \begin{array}{l} V = \{ v_i \mid 1 \le i \le n \} \\ E = \{ (v_i, v_{i-1}) \mid 2 \le i \le n \} \end{array}$$

$C_e = \text{any value} > 0$

סדר לקסיקוגרפי

כלומר:

$$v_n = r \to v_{n-1} \to v_{n-2} \to \cdots \to v_2 \to v_1$$

נקבע $v_n = r$.

נגדיר $Lex(u,v)$ כ"ערך לקסיקוגרפי" ולכן לכל $2 \le i < j \le n$:

$$Lex[(v_i, v_{i-1})] < Lex[(v_j, v_{j-1})]$$

ולכן הלולאה הפנימית תסרוק קודם את (v_2, v_1) ולאחר מכן $(v_{3,2})$ וכן הלאה. ולכן בכל ריצה שלמה של הלולאה הפנימית, רק הקודקוד שאחרי הראשון יתעדכן (שכן בהתחלה כל השאר מאותחלים ב-∞ למעט הראשון) כך "ימשך" לאורך $n-1$ איטרציות חיצוניות עד שבאיטרציה ה-n-ית לא יתבצע שום שינוי והאלג' יסתיים.

ג- נגדיר:

$G'_n = (V', E')$ Where $V' = \{v_i \mid 1 \le i \le n\}$

$E' = \{(v_i, v_{i+1}) \mid 1 \le i < n\}$

נתבונן $r = v_1$ $\qquad c_e = \text{any value} > 0$

סדר לקסיקוגרפי $\Rightarrow$

(כלומר: $r = v_1 \to v_2 \to v_3 \cdots v_{n-1} \to v_n$)

לפי הגדרת Lex בסעיף ב: לכל $1 \le i < j < n$:

$$Lex[(v_i, v_{i+1})] < Lex[(v_j, v_{j+1})]$$

ולכן הסריקה שהלולאה הפנימית תבצע הינה תתחיל מהקשת (v_1, v_2) כאשר $A[v_1] = 0$, $A[v_2] = \infty$ ולכן העדכון יתבצע לאורך b המסלול (שכן ההתקדמות הינה לפי הסדר הלקסיקוגרפי) ובסוף הלולאה יהיה הערך של המסלול הקצר ביותר (למעשה יש מסלול יחיד בין r לבין כל $v \neq r$). לכן באיטרציה השנייה של הלולאה החיצונית לא יתבצע שום עדכון והאלג' יסתיים. כמו כן:

$$|E'| = |E|$$