שעור 4 מציאת תת-מטריצה מלבנית בעלת סכום איברים גדול ביותר.

בעיה זו נמצאת בשימוש נרחב ביישומים כגון זיהוי תבניות, עיבוד תמונה, ניתוח רצף ביולוגי כריית מידע או כריית נתונים (Data mining).

<u>דוגמה</u>: עבור מטריצה נתונה:

0	-2	-7	0
9	2	-6	2
-4	1	-4	1
-1	8	0	-2

תת-מטריצה בעלת סכום גדול ביותר (15) היא

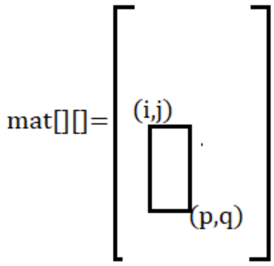
9	2
-4	1
-1	8

א) חיפוש שלם

בהינתן מטריצה mat[m,n] בעלת m שורות ו- n עמודות צריך לעבור על כל תתי מטריצות מלבניות.

תת-מטריצה מלבנית שקואורדינטות של פינה שמאלית עליונה הן `(i,j)` וקואורדינטות של פינה ימנית תחתונה הן `(p,q)` מתקבלת ק חיתוך של שני פסים: פס אנכי בין עמודות `i` ו- `j` ופס אופקי שבין עמודות `p` ו- `q` (`i≤j, p≤q`). מספר אפשרויות לבנות "פס" של שורות הוא $\binom{m}{2}=\frac{m(m-1)}{2}$

מספר אפשרויות לבנות "פס" של עמודות הוא $\frac{n(n-1)}{2}=\binom{n}{2}$. סה"כ מספר תת-מטריצות מלבניות הוא $O(n^2m^2)=\frac{m(m-1)}{2}*\frac{n(n-1)}{2}$. סיבוכיות חישוב סכום איברי המלבן היא $O(m*n)$, לכן סיבוכיות של חיפוש שלם היא $\boldsymbol{O(n^3m^3)}$.



ב) שימוש במטריצת עזר

קודם כל נביא דוגמה של שימוש במערך עזר למערך רגיל.

נתון מערך

`a[]={a[0],a[1],…,a[n-1]}` ,

נגדיר מערך עזר

`h[] ={h[0],h[1],…,h[n-1]}`

כך ש-

```
h[0]=a[0],
h[1]=h[0]+a[1],
      … ,
h[i]=h[i-1]+a[i],
     … ,
h[n-1]=h[n-2]+a[n-1]
```

או במילים אחרות

```
h[i]=a[0]+a[1]+ … +a[i]
```

לכן סכום של איברי קטע במערך a שווה להפרש של שני איברים במערך h:

```
a[p]+a[p+1]+ … +a[q-1]+a[q] = h[q] – h[p].
```

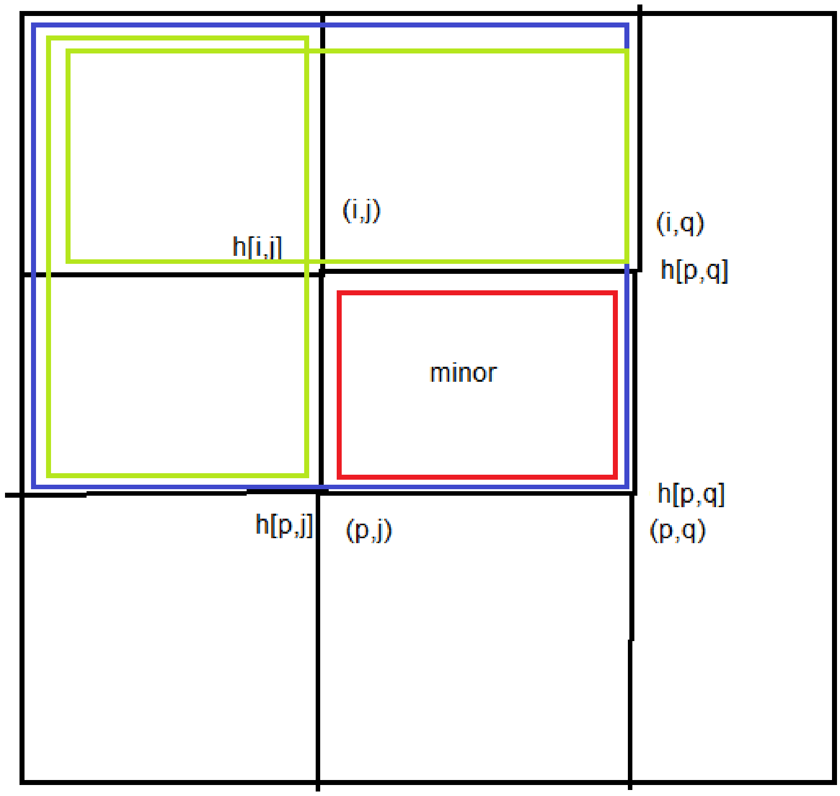
באופן דומה, עבור מטריצה נתונה נגדיר מטריצת עזר `h[m,n]`:

```
int[][] getHelpMatrix(mat[][]
      h[m,n]

      h[0,0] = m[0,0]
      for i=1 to m-1
          h[i][0] = h[i-1][0] + m[i][0], i=0,1,…,m-1

      for j=1 to n-1
         h[0][j] = h[0][j-1] + m[0][j], j=0,1,…,n-1

      for i=1 to m-1
         for j=1 to n-1
            h[i,j] = m[i,j] + h[i-1,j] + h[i,j-1] - h[i-1,j-1]
      return help
end-getHelpMatrix
```



לכן סכום של איברי תת-מטריצה ניתן לחשב ב- O(1):

```
int sum_ij_pq(help[][], i, j, p, q)
   if (i==0 and j==0) sum = h[p,q]
   else if (i==0 and j>0)  sum = h[p,q] - h[p,j-1]
   else if (i>0 and j==0)  sum = h[p,q] - h[i-1,q]
   else sum = help[p,q] - help[p,j-1] - help[i-1,q] + help[i-1,j-1]
   return sum
end_sum_ij_pq
```

סיבוכיות של בניית מטריצת עזר היא $O(mn)$, מעבר על כל תתי-מטריצות $O(n^2m^2)$, חישוב סכום איברים של תת-מטריצה $O(1)$, לכן הסיבוכיות של האלגוריתם היא

$$O(mn) + O(n^2m^2) + O(1) = \boldsymbol{O(n^2m^2)}$$

מטריצת עזר למטריצה שבדוגמה שלעיל היא:

```
0   -2  -9    -9
9    9  -4    -2
5    6  -11,  -8
4,  13, -4,   -3
```

ג) שימוש ב-best

נשתמש באותה מטריצת עזר כמו בסעיף ב) לחישוב סכום איברי של תת-מטריצות..

נגדיר פס שורות בין שורה i לשורה p ונחשב את סכומים של תתי-מטריצות הנמצאות באותו פס שורות בין עמודות 0 ל-j. כמו ב-**best** , ברגע שנקבל סכום שלילי מאפסים את הסכום ועוברים לסכום מחדש מעמודה j+1. בכול חישוב סכום מעדכנים את הסכום המקסימאלי.

פסדו-קוד של אלגוריתם:

```
int matrixSuperBestON3(int[][] mat){//O(N^3)
      help[][] = getHelpMatrix(mat)
      max = help[0][0], t=0
      for i=0; to m-1
           for p=i to m
                j = 0
                for q=0 to q<n
                    t = sum_ij_pq(help, i, j, p, q)//O(1)
                    if (t < 0) j = q + 1
                    else if (t>max) max = t
                end-for
             end-for
      end-for
      return max;
end-matrixSuperBestON3
```