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:

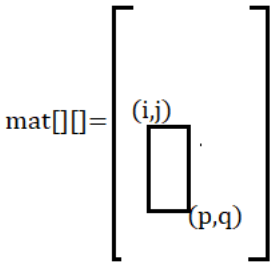
0	-2	-7	0
9	2	-6	2
-4	1	-4	1
-1	8	0	-2

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



חיפוש שלם פסדו-קוד:

```
maxSumSubMatrixExhaustiveSearch(mat[][])
      sum = 0, sumMax = mat[0,0]
      for i=0 to m-1
         for j=0 to n-1
            for p=i to m-1
               for q=j to n-1
                  sum = sumOfSubMatrix(mat,i,j,p,q)
                  if (sumMax < sum) sumMax = sum
               end-for
            end-for
         end-for
      end-for
      return sumMax
end-maxSumSubMatrixExhaustiveSearch

sumOfSubMatrix (mat[][], i, j, p, q) {
      sum=0
      for u=i to p
         for v=j to q
            sum = sum + mat[u][v];
         end-for
      end-for
      return sum
end-sumOfSubMatrix
```

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]=0,h[1],…,h[n-1],h[n]}
כך ש-

```
help[n+1]
help[0] = 0
for (i = 1; i < n+1; i++)
      help[i] = help[i-1] + arr[i-1]
end-for
maxSum = arr[0]
for (int i = 0; i < n+1; i++)
      for (int j = i+1; j < n+1; j++)
            t = help[j] - help[i]
            if (t > maxSum) maxSum = t

return maxSum
```

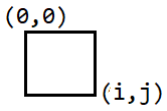
או במילים אחרות

h[i]=a[0]+a[1]+ … +a[i-1], i=1,…, n

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

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

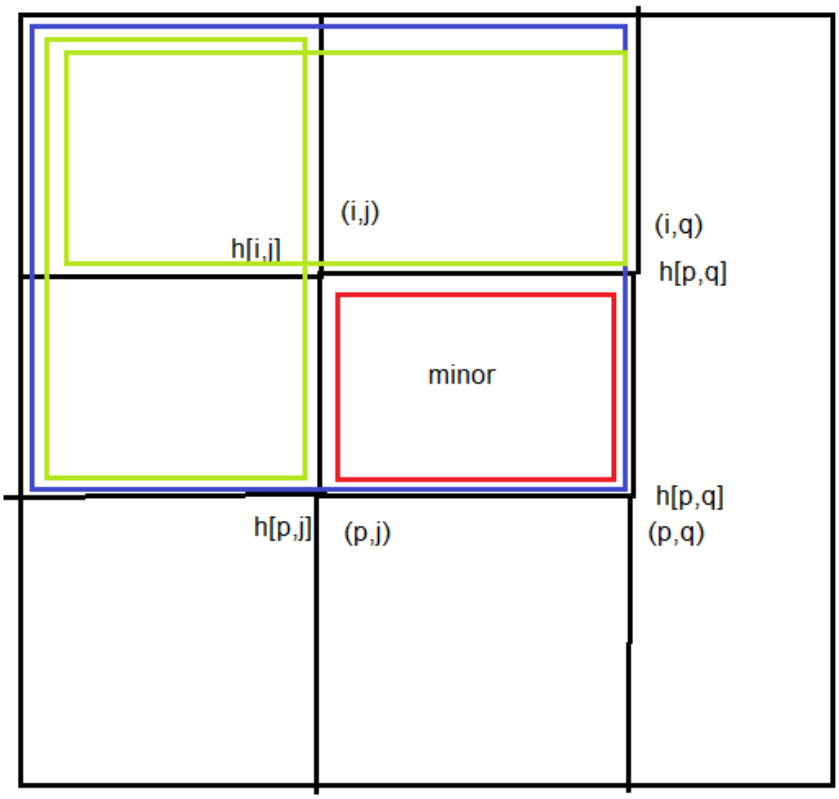
באופן דומה, עבור מטריצה נתונה נגדיר מטריצת עזר h[m,n], כך ש- h[i,j] מהווה סכום איברים של תת-מטריצה ראשית:




```
int[][] helpMatrix(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]

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

      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-helpMatrix
```

לכן סכום איברים של 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
```



ולבסוף:

```
maxSumSubMatrixON4(mat[][]){//O(N^4)
      help[][] = helpMatrix(mat)
      max = help[0][0]
      for i=0 to m-1
            for j=0 to n-1
                  for p=i to m-1
                        for q=j to n-1
                              t = sum_ij_pq(help, i, j, p, q)
                              if (t > max) max = t
                        end-for
                  end-for
            end-for
      end-for
      return max
end-maxSumSubMatrixON4
```

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
```

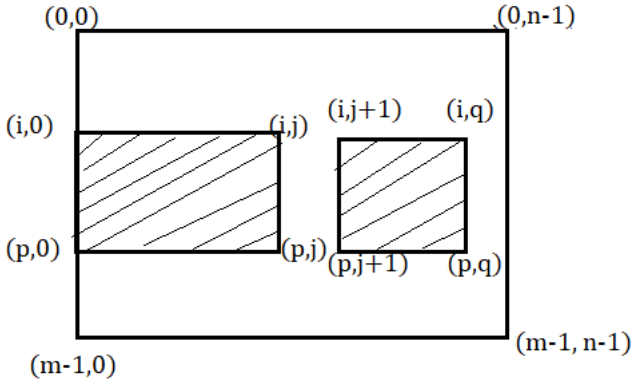
ג) שימוש ב-best

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]
      for i=0; to n-1
          for p=i to n-1
               j = 0
               for q=0 to m-1
                   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
```



סיבוכיות האלגוריתם היא $O(m^2n)$, אבל אותו תרגיל אפשר לעשות לפי עמודות, ולא לפי שורות, לכן סיבוכיות האלגוריתם היא $\boldsymbol{O(\min(m^2n), (mn^2))}$.