קורס אלגוריתמים 1. שעור 6 חישוב f_n.

אלגוריתם לחישוב f_n - מספר הנמצא בסדרת פיבונצ'י במיקום n בסיבוכיות של $O(log_2 n)$.

סדרת פיבונאצ׳י (Fibonacci) היא הסדרה ששני איבריה הראשונים הם 1, 1 וכל איבר לאחר מכן שווה לסכום שני קודמיו:

$$f_1 = f_2 = 1,\ f_n = f_{n-1} + f_{n-2},\ n = 3,4,\dots \qquad (*)$$

בהתאם לכך, איבריה הראשונים של הסדרה הם: 1,1,2,3,5,8,13,... .

ניקח מטריצה $A = \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$,

$$A^2 = A * A = \begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix}, \qquad A^3 = A^2 * A = \begin{pmatrix} 3 & 2 \\ 2 & 1 \end{pmatrix}, \quad A^4 = A^3 * A = \begin{pmatrix} 5 & 3 \\ 3 & 2 \end{pmatrix},$$

רואים כי מקבלים מספרי פיבונצ׳: $A = \begin{pmatrix} f_2 & f_1 \\ f_1 & 0 \end{pmatrix}$, $A^2 = \begin{pmatrix} f_3 & f_2 \\ f_2 & f_1 \end{pmatrix}$, $A^3 = \begin{pmatrix} f_4 & f_3 \\ f_3 & f_2 \end{pmatrix}$, $A^4 = \begin{pmatrix} f_5 & f_4 \\ f_4 & f_3 \end{pmatrix}$

נוכיח באינדוקציה כי $A^n = \begin{pmatrix} f_{n+1} & f_n \\ f_n & f_{n-1} \end{pmatrix}$.

בסיס אינדוקציה: $A = \begin{pmatrix} f_2 & f_1 \\ f_1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$, השוויון מתקיים.

הנחת אינדוקציה: השוויון מתקיים עבור n : $A^n = \begin{pmatrix} f_{n+1} & f_n \\ f_n & f_{n-1} \end{pmatrix}$.

שלב ההוכחה: יש להוכיח כי השיון מתקיים גם עבור n+1, כלומר כי $A^{n+1} = \begin{pmatrix} f_{n+2} & f_{n+1} \\ f_{n+1} & f_n \end{pmatrix}$.

<u>הוכחה:</u>

$$A^{n+1} = A^n * A = \begin{pmatrix} f_{n+1} & f_n \\ f_n & f_{n-1} \end{pmatrix} * \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} f_{n+1} + f_n & f_{n+1} \\ f_n + f_{n-1} & f_n \end{pmatrix} = \begin{pmatrix} f_{n+2} & f_{n+1} \\ f_{n+1} & f_n \end{pmatrix}$$

השוויון האחרון נובע מנוסחת פיבוני׳ (*). מש״ל.

ידוע לנו כי חישוב x^n ניתן לבצע בסיבוכיות של $O(logn)$. לכן גם חישוב של A^n ניתן לבצע באותה הסיבוכיות.

Psedo-Code:

```
int fibLoop(int n) //O(log(n))
      if (n==1 || n==2) return 1
      n = n-2
      int mat[][] = {{1,1},{1,0}}
      int result[][] = {{1,1},{1,0}}
      while (n != 0)
            if ( (n % 2) != 0 )
                  result = matrixSq2Multi(result, mat);
            end-if
            mat = matrixSq2Multi(mat, mat)
            n = n/2
      end-while
      return result[0][0]

end-fibLoop

int[][] matrixSq2Multi(int[][] m1, int m2[][])//O(1)
      int[][]ans = new int [2][2]
      ans[0][0] = m1[0][0]*m2[0][0] + m1[0][1]*m2[1][0]
      ans[0][1] = m1[0][0]*m2[0][1] + m1[0][1]*m2[1][1]
      ans[1][0] = m1[1][0]*m2[0][0] + m1[1][1]*m2[1][0]
      ans[1][1] = m1[1][0]*m2[0][1] + m1[1][1]*m2[1][1]
      return ans
end-matrixSq2Multi

// recursive solution O(log(n))
int fibRecursion(int n)
      int [][] mat={{1,1}, {1,0}}
      int [][] ans =  fibRecursion(mat, n-1)
      return ans[0][0]
end-fibRecursion

int[][] fibRecursion(int mat[][], int n)
      int [][] A={{1,0}, {0,1}}
      if ( n == 0)  return A
      else if (n%2 == 0)
            return fibRecursion(matrixSq2Multi(mat,mat),n/2)
      else
            return
matrixSq2Multi(mat,fibRecursion(matrixSq2Multi(mat,mat),(n - 1)/2))
      end-if
end-fibRecursion
```