1. Rozwiązać układ równań

$$\begin{bmatrix} 4 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 4 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 4 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 4 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \end{bmatrix} \quad (1)$$

Zadanie rozwiązane jest za pomocą metody Givensa. Została ona użyty z uwagi na to że macierz jest rzadka i trój-diagonalna. Zadziałanie 6 razy macierzą Givensa na macierz A powoduje selektywne wyzerowanie elementów pod diagonalą. Dzięki temu możemy użyć metody backsubstitution do wyznaczenia rozwiązań x1...x7.

```
import numpy asn
import math as M
import copy
import sys
a=[1.]*(6)
b = [4.0]*(7)
C=n.diag(a,-1)+n.diag(b, 0) + n.diag(a, 1)
BB=[1, 2, 3, 4, 5, 6, 7]
Solves=[0, 0, 0, 0, 0, 0, 0]
for x in range(7):
    A = copy.copy(C)
    B=copy.copy(BB)
    r = M.sqrt((M.pow(A[x, x],2)) + 1)
    c = A[x, x] / r
    s = 1 / r
    if(x<6):
        C[x + 1, x + 1] = -s * A[x, x + 1] + c * A[x + 1, x + 1]
        C[x, x] = c * A[x, x] + s * A[x + 1, x]
        C[x, x + 1] = c * A[x, x + 1] + s * A[x + 1, x + 1]
        C[x + 1, x] = -s * A[x, x] + c * A[x + 1, x]
        BB[x] = c * B[x] + s * B[x + 1]
        BB[x+1]=-s*B[x]+c*B[x+1]
    if(x<5):
        C[x + 1, x + 2] = -s * A[x, x + 2] + c * A[x + 1, x + 2]
        C[x, x + 2] = s * A[x + 1, x + 2]
Solves[6]=B[6]/A[6,6]
Solves[5]=(B[5]-(A[5,6]*Solves[6]))/A[5,5]
for x in range(4,-1,-1):
    Solves[x] = (B[x] - (A[x, x+1] * Solves[x+1]+A[x,x+2]*Solves[x+2])) /
A[x, x]
print("Solves")
for i in range(7):
    sys.stdout.write("x%d"%(i+1))
    sys.stdout.write("=%f"%Solves[i])
    print
```

Rozwiązania:
x1=0.166789
x2=0.332842
x3=0.501841
x4=0.659794
x5=0.858984
x6=0.904271
x7=1.523932