2 O. Dobierając odpowiedni algorytm (wybór należy uzasadnić!), rozwiązać układ równań

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

Zadanie rozwiązane jest przy pomocy algorytmu Shermana Morrisona.
Algorytm ten został wybrany ponieważ macierz jest symetryczna, rzadka, dodatnio określona i oczywiście trój-diagonalna z dwoma jedynkami w rogach macierzy co jest głównym powodem wyboru algorytmu.

```
import numpy as np
import sys as s
import copy as c

A = np.diag([1.0]*(6), -1) + np.diag([4.0]*7, 0) + np.diag([1.0]*(6), 1)
A[6][0]=1; A[0][6]=1

L=np.zeros(shape=[7,7])
U=np.zeros(shape=[7,7])
uv=np.zeros(shape=[7,7])
uv[0][0]=1; uv[6][6]=1
A=A-uv
U[0][0]=A[0][0]
L[0][0]=1.;

for i in range(0,6):
   U[i][i+1]=A[i][i+1]
   L[i+1][i+1]=1.0;
   L[i+1][i]=(A[i+1][i]/U[i][i])
   U[i+1][i+1]=A[i+1][i+1]-L[i+1][i]*U[i][i+1]

u=np.array([1.,0.,0.,0.,0.,0.,1.])
b=np.array([1.,2.,3.,4.,5.,6.,7.])
u_temp=c.copy(b)
b_temp=c.copy(u)

Z=[0]*7
```

```
Q=[0]*7
Solves=np.array([0.,0.,0.,0.,0.,0.,0.])

for n in range(1,7):
   u_temp[n]=b[n]-((L[n][n-1])*(u_temp[n-1]))
   b_temp[n]=u[n]-((L[n][n-1])*(b_temp[n-1]))

for i in range(6,-1,-1):
   if(i<6):
      Z[i]=(u_temp[i]-((A[i][i+1])*(Z[i+1])))/U[i][i]
      Q[i]=(b_temp[i]-((A[i][i+1])*(Q[i+1])))/U[i][i]
   else:
      Z[6]=u_temp[6]/U[6][6]
      Q[6]=b_temp[6]/U[6][6]

for i in range(7):
   Solves[i]=Z[i]-(np.dot(u,Z)*Q[i])/((1+np.dot(u,Q)))

print("Solves")
for i in range(7):
   s.stdout.write("x%d"%(i+1))
   s.stdout.write("=%f"%Solves[i])
   print
```

**Rozwiązania:**

x1=-0.260163
x2=0.447154
x3=0.471545
x4=0.666667
x5=0.861789
x6=0.886179
x7=1.593496