Zadanie 1a:

Macierz podana w zadaniu jest macierzą symetryczną, trójdiagonalną, dodatnio określoną, a więc jej czynnik Cholesky'ego ma tylko dwie niezerowe diagonale, a koszt jego wyliczenia jest O(N), dlatego do rozwiązania tego przykładu zastosowałem faktoryzację Cholesky'ego uwzględniając strukturę macierzy. Następnie rozwiązanie otrzymujemy poprzed rozwiązanie dwóch układów równań:

1) Cy=b – forward substitution

2) C^tx=y – backward substitution

```
#include <bits/stdc++.h>
#include <iostream>
using namespace std;

const int N = 7;

int main()
{
   int matrix[][N] =  {{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} };

   int b[N] = {1, 2, 3, 4, 5, 6, 7};
   double y[N];
   double x[N];

   double lower[N][N]; // macierz C dolnotrójkątna
   memset(lower, 0, sizeof(lower));

   //************************************
   // FAKTORYACJA CHOLESKY'EGO

   for (int i = 0; i < N; i++) {

      if(i == 0) {
         lower[0][0] = sqrt(matrix[0][0]);
         continue;
      }
            lower[i][i-1] = matrix[i][i-1] / lower[i-1][i-1];
            lower[i][i] = sqrt(matrix[i][i] - pow(lower[i][i-1], 2));

   }



   cout << "Macierz C: " << endl;
   for (int i = 0; i < N; i++) {
      for (int j = 0; j < i+1; j++)
         printf("%f ", lower[i][j]);
      printf("\n");
   }
   cout << endl << endl;

   //  A = C(C^t)x = b
   //  (C^t)*x = y //backsubstitution
   //  Cy = b //forward substitution


   // Obliczanie y
   double sum = 0;
   y[0] = b[0]/lower[0][0];

   for(int i = 1; i < N; ++i) {
```

```
        sum = lower[i][i-1] * y[i-1];
        y[i] = (b[i]-sum) / lower[i][i];
    }


    //  Obliczanie x
    sum = 0;
    x[N-1] = y[N-1]/lower[N-1][N-1];

    for(int i = N-2; i >= 0; --i) {
        sum = lower[i+1][i] * x[i+1];
        x[i] = (y[i]-sum) / lower[i][i];
    }

    for(int i = 0; i < N; ++i)
        cout << "x" << i+1 <<": " << x[i] << endl;
    cout << endl;

    return 0;
}
```

```
Macierz C:
2.000000
0.500000 1.936492
0.000000 0.516398 1.932184
0.000000 0.000000 0.517549 1.931875
0.000000 0.000000 0.000000 0.517632 1.931853
0.000000 0.000000 0.000000 0.000000 0.517638 1.931852
0.000000 0.000000 0.000000 0.000000 0.000000 0.517638 1.931852


x1: 0.166789
x2: 0.332842
x3: 0.501841
x4: 0.659794
x5: 0.858984
x6: 0.904271
x7: 1.52393
```

Program wypisuje wyliczony czynnik Cholesky'ego oraz wyliczone rozwiązania układu równań.

Zadanie 1b:

Podana w zadaniu macierz A_1 jest trójdiagonalna z dodatkowymi niezerowymi elementami znajdującymi się w rogach macierzy.
Tworzymy macierz A odejmując od macierzy A_1 macierz będącą iloczynem diadycznym dwóch wektorów: uv^T, gdzie $u = v = [1\ 0\ 0\ 0\ 0\ 0\ 1]^T$.
W pierwszym kroku dokonujemy faktoryzacji macierzy A algorytmem Cholesky'ego (uwzględniającym strukurę macierzy, która jest taka sama jak struktura macierzy z zadania 1a). Następnie do znalezienia rozwiązań stosujemy algorytm Shermana – Morrisona podany na wykładzie (wykład nr 3, slajd 35). Zastosowane rozwiązanie pozwala rozwiązać problem w czasie liniowym.

```
#include <bits/stdc++.h>
#include <iostream>
using namespace std;
```

```
const int N = 7;

int main()
{
   int n = 7;
   int matrix[][N] ={   {3, 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, 3} };
   int u[N] = {1,
               0,
               0,
               0,
               0,
               0,
               1};

   int v[N] = {1, 0, 0, 0, 0, 0, 1};  // v^t


   int b[N] = {1, 2, 3, 4, 5, 6, 7};

   double y[N];
   double z[N];
   double q[N];

   //************************************
   // FAKTORYACJA CHOLESKY'EGO
   double lower[N][N];
   memset(lower, 0, sizeof(lower));

   for (int i = 0; i < N; i++) {

      if(i == 0) {
         lower[0][0] = sqrt(matrix[0][0]);
         continue;
      }
            lower[i][i-1] = matrix[i][i-1] / lower[i-1][i-1];
            lower[i][i] = sqrt(matrix[i][i] - pow(lower[i][i-1], 2));

   }


   cout << "Macierz C: " << endl;
   for (int i = 0; i < n; i++) {
      for (int j = 0; j < i+1; j++)
         printf("%f ", lower[i][j]);
      printf("\n");
   }
   cout << endl << endl;

   //  A = C(C^t)z = b
   //  (C^t)*z = y
   //  Cy = b


   //*****************************************
   //OBLICZANIE Az = b

   // Obliczanie y
   double sum = 0;
   y[0] = b[0]/lower[0][0];

   for(int i = 1; i < N; ++i) {

      sum = lower[i][i-1] * y[i-1];
      y[i] = (b[i]-sum) / lower[i][i];
```

```
}

//  Obliczanie z
sum = 0;
z[N-1] = y[N-1]/lower[N-1][N-1];

for(int i = N-2; i >= 0; --i) {
    sum = lower[i+1][i] * z[i+1];
    z[i] = (y[i]-sum) / lower[i][i];
}

for(int i = 0; i < N; ++i)
    cout << "z" << i+1 <<": " << z[i] << endl;
cout << endl;



//*********************************************
//OBLICZANIE Aq = u

// Obliczanie y
 sum = 0;
 y[0] = u[0]/lower[0][0];

 for(int i = 1; i < N; ++i) {

     sum = lower[i][i-1] * y[i-1];
     y[i] = (u[i]-sum) / lower[i][i];
 }

 //  Obliczanie q
 sum = 0;
 q[N-1] = y[N-1]/lower[N-1][N-1];

 for(int i = N-2; i >= 0; --i) {
     sum = lower[i+1][i] * q[i+1];
     q[i] = (y[i]-sum) / lower[i][i];
 }

 for(int i = 0; i < N; ++i)
     cout << "q" << i+1 <<": " << q[i] << endl;
 cout << endl;


 //*************************************************
// w = z - q[ (v^t)z / (1 + (v^t)q) ]

 //***
 // (v^t)z  - licznik
 double vtz = 0;

 for(int i = 0; i < N; ++i)
     vtz += v[i] * z[i];


 //***
 // 1 + (v^t)q - mianownik
 double vtq = 0;

 for(int i = 0; i < N; ++i)
     vtq += v[i] * q[i];
 ++vtq;

 //***
 // ulamek * q
 for(int i = 0; i < N; ++i) {
     q[i] *= vtz;
     q[i] /= vtq;
 }
```

```
    //***
    //obliczanie w
    double w[N];

    for(int i = 0; i < N; ++i)
        w[i] = z[i] - q[i];



    for(int i = 0; i < N; ++i)
        cout << "w" << i+1 <<": " << w[i] << endl;
    cout << endl;
   return 0;
}
```

```
Macierz C:
1.732051
0.577350 1.914854
0.000000 0.522233 1.930615
0.000000 0.000000 0.517970 1.931763
0.000000 0.000000 0.000000 0.517662 1.931845
0.000000 0.000000 0.000000 0.000000 0.517640 1.931851
0.000000 0.000000 0.000000 0.000000 0.000000 0.517638 1.652892


z1: 0.2281
z2: 0.315699
z3: 0.509103
z4: 0.647887
z5: 0.899347
z6: 0.754723
z7: 2.08176

q1: 0.366197
q2: -0.0985915
q3: 0.028169
q4: -0.0140845
q5: 0.028169
q6: -0.0985915
q7: 0.366197

w1: -0.260163
w2: 0.447154
w3: 0.471545
w4: 0.666667
w5: 0.861789
w6: 0.886179
w7: 1.5935
```

Program wypisuje wyliczony czynnik Cholesky'ego macierzy A, wektory z i q oraz będący rozwiązaniami układu macierzowego wektor w.