Nomor 1

```
1;
function x = backwardSub(U, b)
  % This solves Ux = b, U is unit upper triangular matrix
  [n n] = size(U); x = zeros(n, 1); x(n) = b(n);
  for i = (n - 1):-1:1
    x(i) = b(i) - U(i, i+1:n) * x(i+1:n);
  endfor
endfunction

function x = forwardSub(L, b)
  % This solves Lx = b, L is unit lower triangular matrix
  [n n] = size(L);  x = zeros(n, 1);  x(1) = b(1);
  for i = 2:n
    x(i) = b(i) - L(i, 1:i-1) * x(1:i-1);
  endfor
endfunction

function x = diagSub(D, b)
  % This solves Dx = b, D is a diagonal matrix
  [n n] = size(D);  x = zeros(n, 1);
  for i = 1:n
    x(i) = b(i)/D(i, i);
  endfor
endfunction

function [A] = getSymmetric(n)
  % Generate a symmetric matrix with uniform distribution
  A = zeros(n, n);
  for i=1:n
    for j=1:n
      A(i, j) = rand();
      if(i > j)
        A(i, j) = A(j, i);
      endif
    endfor
  endfor
endfunction

function [L D] = LDLT(A)
   % Input: A, symmetric matrices
   % Output L and D such that A = L D L'
   [n n] = size(A);
   L = zeros(n, n);
   D = zeros(n);
   for k=1:n
     D(k) = A(k, k) - sum(L(k,1:k-1).^2.*D(1:k-1));
     for i=1:n
       % Cari pembaginya
       L(i, k) = (A(i, k) - sum(L(i, 1:k-1).*L(k, 1:k-1).*D(1:k-1)))/D(k);
     endfor
   endfor
   D = D(1:n,1);
```

```
endfunction

function [L D P] = LDLTwP(A)
   % Input: A, symmetric matrices
   % Output L and D such that A = L D L'
   [n n] = size(A);
   p = 1:n;
   L = zeros(n, n);
   D = zeros(n);
   for k=1:n
     % Extract maximum diagonal of submatrix k .. n : k .. n
     [M pos] = max(diag(A(k:n,k:n)))
     % p is the relative position
     pos += k - 1;

     % Swap pos and k
     tmp = p(pos);
     p(pos) = p(k);
     p(k) = tmp;

     % Swap Diagonal p and k
     tmp = A(pos,1:n)
     A(pos, 1:n) = A(k, 1:n);
     A(k, 1:n) = tmp;

     tmp = A(1:n, pos);
     A(1:n, pos) = A(1:n, k);
     A(1:n, k) = tmp;

     % Swap Diagonal p and k
     tmp = L(pos,1:n)
     L(pos, 1:n) = L(k, 1:n);
     L(k, 1:n) = tmp;

     tmp = L(1:n, pos);
     L(1:n, pos) = L(1:n, k);
     L(1:n, k) = tmp;

     D(k) = A(k, k) - sum(L(k,1:k-1).^2.*D(1:k-1));
     for i=1:n
       % Cari pembaginya
       L(i, k) = (A(i, k) - sum(L(i, 1:k-1).*L(k, 1:k-1).*D(1:k-1)))/D(k);
     endfor
   endfor
   D = D(1:n,1);


   P = zeros(n, n);
   for i=1:n
     P(i, p(i)) = 1
   endfor

endfunction

function [x] = solve(A, b)
  [L D P] = LDLTwP(A);
  disp(L);
  disp(D);
```

```
  disp(P);
  disp(L * diag(D) * L');
  disp(P * A * P');
  b = P * b;
  z = forwardSub(L, b);
  y = diagSub(diag(D), z);
  w = backwardSub(L', y);
  x = P' * w;
endfunction

A = [1, 2, 3, 4; 2, 5, 6, 7; 3, 6, 8, 9; 4, 7 , 9, 0];
b = [4; 1; 5; 8];

A = getSymmetric(3);
b = rand(3, 1);
disp(solve(A, b));
disp(A\b);
```

Jika kita diberikan persamaan $Ax = b$, dan diketahui dekomposisi $A = LDL^\intercal$, maka bisa kita tuliskan $LDL^\intercal x = b$. Pertama, kita akan menyelesaikan dari bagian terluarnya.

- Carilah z sehingga $Lz = b$. Perhatikan bahwa disini $DL^\intercal x = z$.
- Carilah y sehingga $Dy = z$. Perhatikan bahwa disini $L^\intercal x = y$.
- Carilah x sehingga $L^\intercal x = y$. Perhatikan bahwa x adalah solusi yang kita inginkan.

Perhatikan bahwa L ialah matriks **unit segitiga bawah**, artinya entri pada diagonal utamanya berisi satu, kita bisa buat fungsi backward dan forward substitution untuk L dan $L^\intercal$.

```
function x = backwardSub(U, b)
  % This solves Ux = b, U is unit upper triangular matrix
  [n n] = size(U); x = zeros(n, 1); x(n) = b(n);
  for i = (n - 1):-1:1
    x(i) = b(i) - U(i, i+1:n) * x(i+1:n);
  endfor
endfunction

function x = forwardSub(L, b)
  % This solves Lx = b, L is unit lower triangular matrix
  [n n] = size(L);  x = zeros(n, 1);  x(1) = b(1);
  for i = 2:n
    x(i) = b(i) - L(i, 1:i) * x(1:i);
  endfor
endfunction

function x = diagSub(D, b)
  % This solves Dx = b, D is a diagonal matrix
  [n n] = size(D);  x = zeros(n, 1);
  for i = 1:n
    x(i) = b(i)/D(i, i);
  endfor
endfunction
```

Bagaimana kasusnya untuk $PAP^\intercal = LDL^\intercal$? Pertama secara intuitif bisa kita tuliskan $A = P^{-1}LDL^\intercal(P^\intercal)^{-1}$. Perhatikan bahwa $PP^\intercal = I \iff P^\intercal = P^{-1}$.

$$\textbf{Proof}$$

$$(PP^\intercal)_{ij} = \begin{cases} 1 & P_{ik} \land P^\intercal_{kj} \iff i = j \\ 0 & \text{otherwise} \end{cases}$$

Akan kita selesaikan persamaan sebagai berikut.

$$\begin{aligned} P^{-1}LDL^\intercal(P^\intercal)^{-1}x &= b \\ LDL^\intercal(P^\intercal)^{-1}x &= Pb \\ LDL^\intercal(P^\intercal)^\intercal x &= Pb \\ LDL^\intercal Px &= Pb \end{aligned}$$

Untuk kemudahan komputasi, lakukan $b := Pb$

- Carilah z sehingga $Lz = b$. Perhatikan bahwa disini $DL^\intercal Px = z$.
- Carilah y sehingga $Dy = z$. Perhatikan bahwa disini $L^\intercal Px = y$.
- Carilah w sehingga $L^\intercal w = y$. Perhatikan bahwa disini $Px = w$.
- Carilah x sehingga $x = P^\intercal w$.

Total Flops:

- $LDL' = \frac{n^3}{3}$ (Setengah dari LU)
- Backward substitution = Forward Substitution = $\frac{n(n-1)}{2}$.
- Diagonal = n.
- Perkalian matriks permutasi (Kasus diagonal pivoting) n^3 (?)

Nomor 2