LAPORAN PRAKTIKUM

METODE NUMERIK

Judul: Sistem Persamaan Linier

DISUSUN OLEH

Sania Bening Nareswara M0519075

PROGRAM INFORMATIKA

FAKULTAS MIPA

UNIVERSITAS SEBELAS MARET

2020

Sistem Persamaan Linier

Persamaan 1

$$\begin{aligned} 2x - 6y - z &= -38 \\ -3x - y + 7z &= -34 \\ -8x + v - 2z &= -20 \end{aligned}$$

Matriks Ax = b

$$\begin{pmatrix} 2 & -6 & -1 \\ -3 & -1 & 7 \\ -8 & 1 & 2 \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} -38 \\ -34 \\ -20 \end{pmatrix}$$

Persamaan 2

$$\begin{aligned} 2a - b + 10c &= -11 \\ 3b - c + 8d &= -11 \\ 10a - b + 2c &= 6 \\ -a + 11b - c + 3d &= 25 \end{aligned}$$

Matriks Ax = b

$$\begin{pmatrix} 2 & -1 & 10 & 0 \\ 0 & 3 & -1 & 8 \\ 10 & -1 & 2 & 0 \\ -1 & 11 & -1 & 3 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} -11 \\ -11 \\ 6 \\ 25 \end{pmatrix}$$

1. Metode Jacobi

```
%Sania Bening Nareswara (M0519075)

function x = IterasiJacobi(A,b,e,max_iter)
exact = [4;8;-2];
tic;
n = size(A,1); %3
% x1,x2,x3...xn = 0
for i = 1:n
  xlama(i) = b(i)/A(i,i);
end

xlama = xlama';
C = -A;
for i = 1:n
  C(i,i) = 0.0;
  C(i,:) = C(i,:)/A(i,i);
  d(i,1) = xlama(i);
end

i = 1;
while (i <= max_iter)
  xbaru = C*xlama + d;
  if(max(abs(xbaru-xlama)) <= e)
    x = xbaru;
    disp('Jacobi method converge');
    toc;
    error = abs(exact - xbaru);
    disp('Error = ');
    fprintf('%.3f\n', error);
    return
  else
    xlama = xbaru;
  end
  fprintf('\n%.0f   %.3f   %.3f    %.3f\n', i,xbaru(1),xbaru(2),xbaru(3));
  i = i + 1;
end
disp('Jacobi method not converge');
toc;
error = abs(exact - xbaru);
disp('Error = ');
fprintf('%.3f\n', error);
```

Fungsi IterasiJacobi dengan parameter A, b, e, dan max_iter disimpan pada variabel x. Terdapat nilai exact dari penyelesaian persamaan yang nantinya akan digunakan untuk mencari nilai error. Kemudian terdapat tic dan toc yang berguna untuk menghitung waktu dari eksekusi program. Size (A,1) akan mengambil nilai dari size A pada kolom ke-1 dan disimpan pada variabel n. Dilakukan perulangan dari 1 hingga n. Tiap perulangan akan mengeksekusi nilai b pada baris ke -i dibagi dengan nilai A pada baris ke-i kolom i dan hasilnya disimpan pada variabel xlama. Nilai xlama akan ditranspose. Kemudian terdapat fungsi perulangan dimulai dari i hingga n. Setiap perulangan akan mengeksekusi nilai C pada baris ke-i kolom i = 0,0. Nilai C pada baris i untuk setiap kolom adalah Nilai C pada baris i untuk setiap kolom dibagi nilai A baris ke-i kolom i. Nilai d baris i kolom i adalah nilai dari xlama baris ke-i. Fungsi perulangan dimulai dari 1 hingga max_iter. Nilai xbaru merupakan nilai C dikali nilai xlama ditambah dengan d. Kemudian terdapat fungsi percabangan jika nilai absolut dari xbaru-xlama kurang dari sama dengan nilai e, maka nilai x=xbaru akan menampilkan output “Jacobi method converge” dan perulangan berhenti. Namun, jika tidak memenuhi, maka akan menampilkan output “Jacobi method not converge”. Perhitungan nilai error menggunakan nilai mutlak dari nilai exact-xlama. Kemudian hasil error akan ditampilkan.

Persamaan 1

```
1   4.506   7.976    -2.881

2   4.217   8.315    -1.787

3   3.986   8.037    -1.862

4   3.970   7.972    -2.001

5   3.997   7.990    -2.017

6   4.003   8.002    -2.003

7   4.001   8.001    -1.998

8   4.000   8.000    -1.999

9   4.000   8.000    -2.000

10   4.000   8.000    -2.000
Jacobi method converge
Elapsed time is 0.002737 seconds.
Error =
0.000
0.000
0.000

ans =

    4.0000
    8.0000
   -2.0000
```

Sebelumnya, matriks diubah menjadi diagonally dominant agar penyelesaiannya konvergen.

$$\begin{pmatrix} -8 & 1 & -2 \\ 2 & -6 & -1 \\ -3 & -1 & 7 \end{pmatrix}\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} -20 \\ -38 \\ -34 \end{pmatrix}$$

Pada percobaan ini, error yang digunakan adalah 0.0001 dan max_iter yang digunakan adalah 20. Pada iterasi ke-10 didapatkan hasil konvergen dengan penyelesaian eksaknya, yaitu 4;8;-2. Dengan error 0 dan waktu yang dibutuhkan 0.002737 detik.

Persamaan 2

```
1   1.047   2.602   -0.993   -2.365

2   1.059   2.923   -1.049   -2.475

3   1.102   2.949   -1.019   -2.602

4   1.099   2.990   -1.026   -2.608

5   1.104   2.991   -1.021   -2.624

6   1.103   2.996   -1.022   -2.624

7   1.104   2.996   -1.021   -2.626

8   1.104   2.997   -1.021   -2.626

9   1.104   2.996   -1.021   -2.626
Jacobi method converge
Elapsed time is 0.001629 seconds.
Error =
0.000
0.000
0.000
0.000

ans =

    1.1039
    2.9965
   -1.0211
   -2.6263
```

Sebelumnya, matriks diubah menjadi diagonally dominant agar penyelesaiannya konvergen.

$$\begin{pmatrix} 10 & -1 & 2 & 0 \\ -1 & 11 & -1 & 3 \\ 2 & -1 & 10 & 0 \\ 0 & 3 & -1 & 8 \end{pmatrix}\begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} 6 \\ 25 \\ -11 \\ -11 \end{pmatrix}$$

Pada percobaan ini, error yang digunakan adalah 0.0001 dan max_iter yang digunakan adalah 20. Pada iterasi ke-9 didapatkan hasil konvergen dengan penyelesaian eksaknya, yaitu 1.104;2.996;-1.021. Dengan error 0 dan waktu yang dibutuhkan 0.001629 detik.

2. Metode Gauss-Seidel

```
%Sania Bening Nareswara (M0519075)
function x = IterasiGaussSeidel(A,b,e,max_iter)
exact = [4; 8; -2];
tic;
n = size(A,1);
for i = 1:n
  xlama(i) = b(i)/A(i,i);
end
xlama = xlama';
C = -A;
for i = 1:n
  C(i,:) = C(i,:)/A(i,i);
  d(i,1) = xlama(i);
end
i = 1;
while (i <= max_iter)
  xbaru = C*xlama + d;
  if(abs(xbaru-xlama) < e)
    x = xbaru;
    disp('Gauss Seidel method converge');
    return
  else
    xlama = xbaru;
  end
  fprintf('\n%.0f   %.3f   %.3f    %.3f\n', i,xbaru(1),xbaru(2),xbaru(3));
  i = i + 1;
end
disp('Gauss Seidel method not converge');
toc;
error = abs(exact - xbaru);
disp('Error = ');
fprintf('%.3f\n', error);
```

Fungsi IterasiGaussSeidel dengan parameter A, b, e, dan max_iter disimpan pada variabel x. Terdapat nilai exact dari penyelesaian persamaan yang nantinya akan digunakan untuk mencari nilai error. Kemudian terdapat tic dan toc yang berguna untuk menghitung waktu dari eksekusi program. Size (A,1) akan mengambil nilai dari size A pada kolom ke-1 dan disimpan pada variabel n. Dilakukan perulangan dari 1 hingga n. Tiap perulangan akan mengeksekusi nilai b pada baris ke -i dibagi dengan nilai A pada baris ke-i kolom i dan hasilnya disimpan pada variabel xlama. Nilai xlama akan ditranspose. Kemudian terdapat fungsi perulangan dimulai dari i hingga n. Setiap perulangan akan mengeksekusi nilai C pada baris ke-i kolom i = 0,0. Nilai C pada baris i untuk setiap kolom adalah Nilai C pada baris i untuk setiap kolom dibagi nilai A baris ke-i kolom i. Nilai d baris i kolom i adalah nilai dari xlama baris ke-i. Fungsi perulangan dimulai dari 1 hingga max_iter. Nilai xbaru merupakan nilai C dikali nilai xlama ditambah dengan d. Kemudian terdapat fungsi percabangan jika nilai absolut dari xbaru-xlama kurang dari sama dengan nilai e, maka nilai x=xbaru akan menampilkan output "Jacobi method converge" dan perulangan berhenti. Namun, jika tidak memenuhi, maka akan menampilkan output "Jacobi method not converge". Perhitungan nilai error menggunakan nilai mutlak dari nilai exact-xlama. Kemudian hasil error akan ditampilkan.

Persamaan 1

```
>> IterasiGaussSeidel(A,b,e,max_iter)

1   2.006   1.643    1.976

2   0.205   5.030    -5.739

3   4.358   2.329    1.688

4   -1.989   5.176    -4.345

5   6.222   1.218    -0.625

6   -3.414   7.293    -1.391

7   7.173   -1.866    -3.887

8   -3.935   11.238    1.838

9   7.380   -6.523    -6.776

10   -4.002   16.446    4.150

11   7.520   -12.138    -8.372

12   -4.444   22.373    5.004

13   8.490   -18.355    -8.570

14   -6.142   28.947    4.729

15   11.078   -25.449    -8.083

16   -9.738   36.822    4.338

17   15.756   -34.457    -8.108

18   -15.536   47.394    5.081

19   22.690   -47.087    -9.826

20   -23.620   62.621    7.967
Gauss Seidel method not converge
Elapsed time is 0.000884 seconds.
Error =
27.620
54.621
9.967
```

Sebelumnya, matriks diubah menjadi diagonally dominant terlebih dahulu agar konvergen. Pada percobaan ini, error yang digunakan adalah 0.0001 dan max_iter yang digunakan adalah 20. Pada iterasi ke-20 hasilnya masih belum konvergen dengan penyelesaian eksaknya, yaitu -23.620;62.621;7.967. Dengan error 27.620;54.621;9.967 dan waktu yang dibutuhkan 0.000884 detik.

Persamaan 2

```
>> IterasiGaussSeidel(A, b, e, max_iter)

1   0.447   0.330   0.107   -0.990

2   0.164   2.264   -1.264   -0.495

3   0.915   0.044   0.357   -1.886

4   -0.382   2.859   -1.636   0.539

5   1.595   -0.916   0.898   -3.191

6   -1.266   4.286   -2.409   2.272

7   2.776   -2.967   1.990   -5.555

8   -2.871   7.188   -3.942   5.541

9   4.979   -7.046   4.136   -10.105

10   -5.910   12.903   -6.936   11.889

11   9.188   -15.041   8.308   -18.969

12   -11.753   24.077   -12.750   24.273

13   17.311   -30.652   16.408   -36.271

14   -23.058   45.882   -24.036   48.441

15   33.053   -61.102   32.135   -70.027

16   -44.991   88.399   -45.956   95.582

17   63.622   -120.462   62.694   -135.851

18   -87.607   171.269   -88.565   187.486

19   123.047   -236.144   122.113   -264.158

20   -170.484   332.747   -171.437   366.601
Gauss Seidel method not converge
Elapsed time is 0.001655 seconds.
Error =
171.588
329.750
170.416
369.227
```

Sebelumnya, matriks diubah menjadi diagonally dominant terlebih dahulu agar konvergen. Pada percobaan ini, error yang digunakan adalah 0.0001 dan max_iter yang digunakan adalah 20. Pada iterasi ke-20 hasilnya masih belum konvergen dengan penyelesaian eksaknya, yaitu -170.484;332.747;-171.437;366.601. Dengan error 171.588;329.750;170.416;369.227 dan waktu yang dibutuhkan 0.001655 detik.

3. Eliminasi Gauss Jordan

```
%Sania Bening Nareswara (M0519075)
function x = EliminasiGaussJordan(A,b)
exact = [4; 8; -2];
tic;
n = size(A,1);
for i = 1 : n-1 % i=1:2
    [pivot,k] = max(abs(A(i:n, i))) % max(abs(A(1:3, 1)))
    if (k > 1)
        temp1 = A(i, :);
        temp2 = b(i, :);
        A(i,:) = A(i+k-1,:); % A(3,:)
        b(i,:) = b(i+k-1,:);
        A(i+k-1,:) = temp1;
        b(i+k-1,:) = temp2;
    end
    for h = i+1 : n
        m = A(h,i)/A(i,i);
        A(h,:) = A(h,:) - m*A(i,:);
        b(h,:) = b(h,:) - m*b(i,:);
    end
end
for i = n:-1:2
    for h = i-1:-1:1
        m = A(h,i)/A(i,i);
        A(h,:) = A(h,:)-m*A(i,:);
        b(h,:) = b(h,:)-m*b(i,:);
    end
end
for i = 1:n
    x(i,:) = b(i,:)/A(i,i);
end
toc;
error = abs(exact -x);
disp('Error = ');
fprintf('%.3f\n', error);
```

Fungsi IterasiGaussJordan dengan parameter A dan b disimpan pada variabel x. Terdapat nilai exact dari penyelesaian persamaan yang nantinya akan digunakan untuk mencari nilai error. Kemudian terdapat tic dan toc yang berguna untuk menghitung waktu dari eksekusi program. Size (A,1) akan mengambil nilai dari size A pada kolom ke-1 dan disimpan pada variabel n. Dilakukan perulangan dari 1 hingga n-1. Terdapat nilai pivot yang merupakan nilai maksimun dari mutlak A baris I hingga n kolom i, letak pivot ditunjukkan dengan nilai k. Terdapat percabangan, jika k lebih dari 1 maka nilai A baris ke-i akan diganti dengan nilai A baris ke i+k-1 dan nilai B baris ke-i akan diganti dengan nilai B baris ke i+k-1. Terdapat perulangan nilai h dari i+1 hingga n. Nilai m adalah nilai dari matriks A baris ke h kolom ke i dibagi dengan matriks A baris ke i kolom ke i. Nilai dari matriks A baris ke h di semua kolom adalah A baris ke h di semua kolom dikurangi dengan m kali A baris ke i di semua kolom. Sedangkan nilai matriks B baris ke h di semua kolom adalah nilai B baris ke h di semua kolom dikurangi dengan m kali B baris ke i di semua kolom. Setiap perulangan akan mengeksekusi nilai C pada baris ke i kolom i = 0.0. Nilai C pada baris i untuk setiap kolom adalah nilai C pada baris i untuk setiap kolom dibagi nilai A baris ke i kolom i. Nilai d baris i kolom i adalah nilai dari xlama baris ke i. Perhitungan nilai error menggunakan nilai mutlak dari nilai exact-xlama. Kemudian hasil error akan ditampilkan.

Persamaan 1

```
>> EliminasiGaussJordan(A, b)

pivot =

     8


k =

     1


pivot =

    5.7500


k =

     1

Elapsed time is 0.005230 seconds.
Error =
0.000
0.000
0.000

ans =

     4
     8
    -2
```

Matriks tidak diubah menjadi diagonally dominant terlebih dahulu. Pada percobaan ini, error yang digunakan adalah 0.0001 dan tidak menggunakan iterasi. Pada saat program dieksekusi, hasilnya konvergen dengan penyelesaian eksaknya, yaitu 4;8;-2. Dengan error 0 dan waktu yang dibutuhkan 0.005230 detik.

Persamaan 2

```
>> EliminasiGaussJordan(A, b)

pivot =

    10


k =

     1


pivot =

   10.9000


k =

     1


pivot =

    9.5413


k =

     1
```

```
Elapsed time is 0.002070 seconds.
Error =
0.000
0.001
0.000
0.000

ans =

    1.1039
    2.9965
   -1.0211
   -2.6263
```

Matriks tidak diubah menjadi diagonally dominant terlebih dahulu. Pada percobaan ini, error yang digunakan adalah 0.0001 dan tidak menggunakan iterasi. Pada saat program dieksekusi, hasilnya konvergen dengan penyelesaian eksaknya, yaitu 1.1039;2.9965;-1.0211;-2.6263. Dengan error 0 dan waktu yang dibutuhkan 0.002070 detik.

4. Eliminasi Gauss

```
%Sania Bening Nareswara (M0519075)

function x = EliminasiGauss (A,b)
exact = [4;8;-2];
tic;
[n,l] = size(A);
for i = 1 : n-1,
    [pivot,k] = max(abs(A(i:n, i)));
    if (k > 1)
        temp1 = A(i, :);
        temp2 = b(i, :);
        A(i,:) = A(i+k-1,:);
        b(i,:) = b(i+k-1,:);
        A(i+k-1,:) = temp1;
        b(i+k-1,:) = temp2;
    end
    for (h = i+1 : n),
        m = A(h,i)/A(i,i);
        A(h,:) = A(h,:) - m*A(i,:);
        b(h,:) = b(h,:) - m*b(i,:);
    end
end
x(n,:) = b(n,:) / A(n,n);
for (i = n:-1:1),
    x(i,:) = (b(i,:)-A(i,i+1:n)*x(i+1:n,:)) / A(i,i);
end
toc;
error = abs(exact - x);
disp('Error = ');
fprintf('%.3f\n', error);
```

Persamaan 1

```
>> EliminasiGauss(A, b)
Elapsed time is 0.017263 seconds.
Error =
0.000
0.000
0.000

ans =

     4
     8
    -2
```

Matriks tidak diubah menjadi diagonally dominant terlebih dahulu. Pada percobaan ini, error yang digunakan adalah 0.0001 dan tidak menggunakan iterasi. Pada saat program dieksekusi, hasilnya konvergen dengan penyelesaian eksaknya, yaitu 4;8;-2. Dengan error 0 dan waktu yang dibutuhkan 0.017263 detik.

Persamaan 2

```
>> EliminasiGauss(A, b)
Elapsed time is 0.000150 seconds.
Error =
0.000
0.001
0.000
0.000

ans =

    1.1039
    2.9965
   -1.0211
   -2.6263
```

Matriks tidak diubah menjadi diagonally dominant terlebih dahulu. Pada percobaan ini, error yang digunakan adalah 0.0001 dan tidak menggunakan iterasi. Pada saat program dieksekusi, hasilnya konvergen dengan penyelesaian eksaknya, yaitu 1.1039;2.9965;-1.0211;-2.6263. Dengan error 0 dan waktu yang dibutuhkan 0.000150 detik.

5. Faktorisasi LU

```
%Sania Bening Nareswara (M0519075)

function x = LU_Solusi (L,U,b)
exact = [4;8;-2];
tic;
[n,m] = size(L);
z = zeros(n,1);
x = zeros(n,1);
z(1) = b(1)/L(1,1);
for i = 2:n,
    z(i) = (b(i) - L(i, 1:i-1)*z(1:i-1)) / L(i,i);
end
x(n) = z(n)/U(n,n);
for i = n-1:-1:1,
    x(i) = (z(i) - U(i,i+1:n)*x(i+1:n)) / U(i,i);
end
toc;
error = abs(exact - x);
disp('Error = ');
fprintf('%.3f\n', error);
```

```
%Sania Bening Nareswara (M0519075)

function [L,U] = Doolittle (A)
tic;
[n,m] = size(A);
U = zeros (n,n);
L = eye(n);
for k = 1:n,
    U(k,k) = A(k,k) - L(k, 1:k-1)*U(1:k-1,k);
    for j = k+1:n,
        U(k,j) = A(k,j) - L(k, 1:k-1)*U(1:k-1,j);
        L(j,k) = (A(j,k) - L(j, 1:k-1)*U(1:k-1,k))/U(k,k);
    end
end
toc;
```

Persamaan 1

```
>> [L,U] = Doolittle(A)
Elapsed time is 0.000106 seconds.

L =

    1.0000         0         0
   -0.2500    1.0000         0
    0.3750    0.2391    1.0000


U =

   -8.0000    1.0000   -2.0000
         0   -5.7500   -1.5000
         0         0    8.1087

>> LU_Solusi(L, U, b)
Elapsed time is 0.000078 seconds.
Error =
0.000
0.000
0.000

ans =

     4
     8
    -2
```

Pada percobaan ini, error yang digunakan adalah 0.0001 dan tidak menggunakan iterasi. Selain itu, matriks tidak diubah menjadi diagonally dominant tetapi dibentuk matriks L dan U dahulu. Hasil setelah program dieksekusi adalah 4;8;-2. Dengan error 0 dan waktu yang dibutuhkan 0.000078 detik.

Persamaan 2

```
>> [L,U] = Doolittle(A)
Elapsed time is 0.000121 seconds.

L =

    1.0000         0         0         0
   -0.1000    1.0000         0         0
    0.2000   -0.0734    1.0000         0
         0    0.2752   -0.0817    1.0000


U =

   10.0000   -1.0000    2.0000         0
         0   10.9000   -0.8000    3.0000
         0         0    9.5413    0.2202
         0         0         0    7.1923

>> LU_Solusi(L, U, b)
Elapsed time is 0.000071 seconds.
Error =
0.000
0.001
0.000
0.000

ans =

    1.1039
    2.9965
   -1.0211
   -2.6263
```

Pada percobaan ini, error yang digunakan adalah 0.0001 dan tidak menggunakan iterasi. Selain itu, matriks tidak diubah menjadi diagonally dominant tetapi dibentuk matriks L dan U dahulu. Hasil setelah program dieksekusi adalah 1.1039;2.9965;-1.0211;-2.6263. Dengan error 0 dan waktu yang dibutuhkan 0.000071 detik.

A. Hasil persamaan menggunakan metode Gauss, Gauss Jordan, Gauss Seidel, Jacobi, LU Solution.

Metode	Persamaan 1
Jacobi	4 ; 8 ; -2
Gauss Seidel	-23.620 ; 62.621 ; 7.967
Gauss Jordan	4 ; 8 ; -2
Gauss	4 ; 8 ; -2
Faktorisasi LU	4 ; 8 ; -2

Metode	Persamaan 2
Jacobi	1.1039 ; 2.9965 ; -1.0211; -2.6263
Gauss Seidel	-170.484 ; 332.747 ; -171.437 ; 366.601
Gauss Jordan	1.1039 ; 2.9965 ; -1.0211; -2.6263
Gauss	1.1039 ; 2.9965 ; -1.0211; -2.6263
Faktorisasi LU	1.1039 ; 2.9965 ; -1.0211; -2.6263

B. Error dari tiap metode

Metode	Persamaan 1	Persamaan 2
Jacobi	0	0
Gauss Seidel	27.620;54.621;9.967	171.588;329.750;170.416;369.227
Gauss Jordan	0	0
Gauss	0	0
Faktorisasi LU	0	0

C. Perbandingan proses iterasi dari segi running time, banyak iterasi, konvergensi.

Persamaan 1

Metode	Running Time (s)	Jumlah Iterasi	Konvergensi
Jacobi	0.002737	10	konvergen
Gauss Seidel	0.000884	20	tidak konvergen
Gauss Jordan	0.005230	0	konvergen
Gauss	0.017263	0	konvergen
Faktorisasi LU	0.000078	0	konvergen

Persamaan 2

Metode	Running Time (s)	Jumlah Iterasi	Konvergensi
Jacobi	0.001629	9	konvergen
Gauss Seidel	0.001655	20	tidak konvergen
Gauss Jordan	0.002070	0	konvergen
Gauss	0.000150	0	konvergen
Faktorisasi LU	0.000071	0	konvergen

Berdasarkan perbandingan dari penyelesaian sistem persamaan linear menggunakan 5 metode, dapat disimpulkan bahwa metode yang paling cepat berdasarkan running time-nya adalah metode Faktorisasi LU. Metode yang memiliki iterasi adalah metode Jacobi (10 iterasi dan 9 iterasi) dan Gauss Seidel (20 iterasi). Metode yang tidak konvergen adalah metode Gauss Seidel. Kemudian, metode yang paling baik untuk digunakan adalah metode Faktorisasi LU karena hasilnya konvergen, tidak memerlukan iterasi, dan running time-nya cepat.