Andi Aqil Amanulhaq
06111740000105
PLM Kelas D

Tugas

Bab 11 No. 23

```
A = [2, 2, 1; 0, 1, 2; 1, 1, 3]
b = [2; 1; 3]

C = [A b] % penyelesaian untuk Gauss
D = [A b] % penyelesaian untuk Gauss-Jordan

% Penyelesaian dengan fungsi rref
rref(C)
%

% Penyelesaian manual dengan OBE

%Solve using Gauss
fprintf('Penyelesaian dengan Gauss:')
% Row Operations:
C(1,:) = C(1,:) / 2
C(3,:) = C(3,:) - C(1,:)
C(3,:) = C(3,:) / 2.5

%didapat hasilnya dengan back substitution:
x3 = 0.8
x2 = 1 - 2*x3
x1 = 1 - x2 - 0.5*x3

% Solve Using Gauss Jordan
fprintf('Penyelesaian dengan Gauss-Jordan:')
% Row Operations :
D(1,:) = D(1,:) / 2
D(3,:) = D(3,:) - D(1,:)
D(3,:) = D(3,:) / 2.5
D(1,:) = D(1,:) - 0.5*D(3,:)
D(2,:) = D(2,:) - 2*D(3,:)
D(1,:) = D(1,:) - D(2,:)
fprintf('Hasil dari Gauss-Jordan:')
gj_x1 = D(1,4)
gj_x2 = D(2,4)
gj_x3 = D(3,4)

%Using Inverse of Matrix
inverse_A = inv(A)
sol = inverse_A * b
```

Bab 11 No. 29

```
A = [
    4, -1, 0, 3;
    -2, 3, 1, -5;
    1, 1, -1, 2;
    3, 2, -4, 0;
    ]
b = [10, -3, 2, 4]'

% penyelesaian dengan menggunakan solve
syms x1 x2 x3 x4
col_x1 = x1 * A(:,1)
col_x2 = x2 * A(:,2)
col_x3 = x3 * A(:,3)
col_x4 = x4 * A(:,4)
alg_v = col_x1 + col_x2 + col_x3 + col_x4
fprintf('Hasil:')
[Sx1, Sx2, Sx3, Sx4]  = solve(alg_v(1) == b(1), alg_v(2) == b(2), alg_v(3) == b(3), alg_v(4) == b(4))

% penyelesaian dengan menggunakan metode lain
c = [A b]
rref(c)
% x1 = 2.5581, x2 = 0.4419, x3 = 1.1395, x4 = 0.0698, jika dihitung sama
```

Bab 12 No. 21

```
function output = matsort(X)
% Sort Matrix for any size n x m
% X -- argument in matrix type
    X_size = size(X);
    X_list = reshape(X, 1, X_size(1) * X_size(2));
    output = reshape(sort(X_list), X_size(1), X_size(2));
end

A = [4 5 2; 1 3 6; 7 8 4; 9 1 5]
matsort(A)
```

Bab 12 No. 22

```
function sorted_vector = vectsort(vect, direction)
% Function to sort a vector
% vect -- input argument in vector/list type
% direction -- option argument, 'a' for ascending, 'd' for descending

if direction == 'a'
    sorted_vector = sort(vect, 'ascend');
elseif direction == 'd'
    sorted_vector = sort(vect, 'descend');
end

end

A = [3, 5, 2, 6, 9, 1]

vectsort(A, 'a')
vectsort(A, 'd')
```

Bab 13

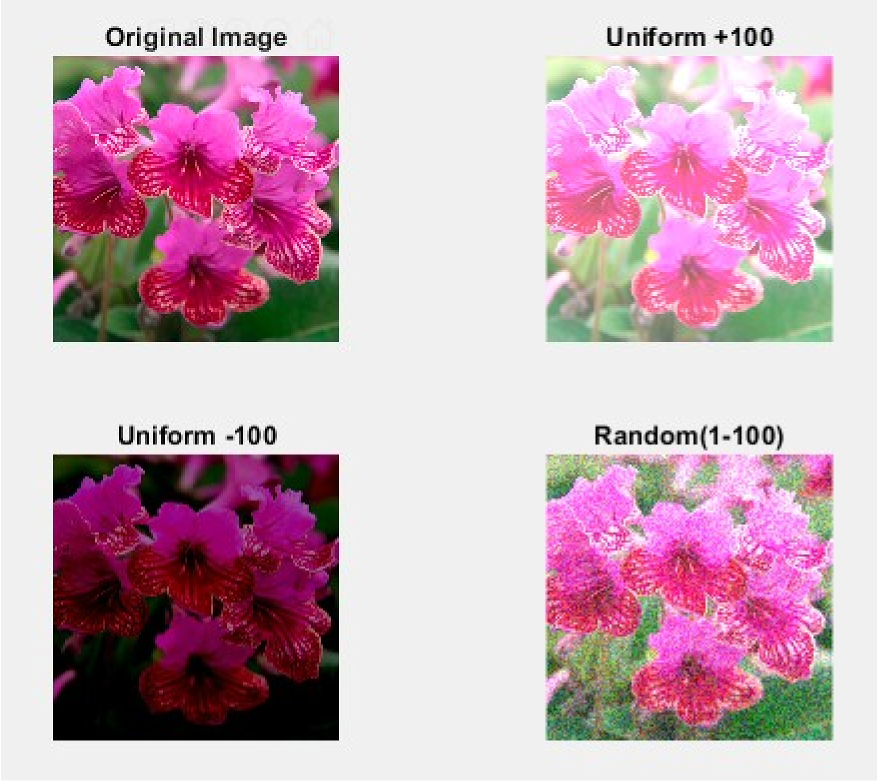
Untuk Bab 13, demonstrasi gambar kebanyakan menggunakan ‘flower.jpg’ yang ada di bawah ini :

Bab 13 No. 14

```
A = double(imread('flower','jpg'));
%B = imread('cat','png');

red_mean = mean(mean(A(:,:,1)))
green_mean = mean(mean(A(:,:,2)))
blue_mean = mean(mean(A(:,:,3)))

mean_color = [red_mean, green_mean, blue_mean]
%image(mean_color) %rata-ratanya jika dikombinasi akan berwarna kuning

red_std = std(std(A(:,:,1)))
green_std = std(std(A(:,:,2)))
blue_std = std(std(A(:,:,3)))
color_std = [red_std, green_std, blue_std]
```

Bab 13 No. 15

```
image = imread('flower.jpg');
new1 = image + 100;
new2 = image - 100;
random = uint8(randi(100,size(image)))
new3 = image + random;


subplot(2,2,1), imshow(image)
title('Original Image')
subplot(2,2,2), imshow(new1)
title('Uniform +100')
subplot(2,2,3), imshow(new2)
title('Uniform -100')
subplot(2,2,4), imshow(new3)
title('Random(1-100)')
```



Gambar di atas adalah hasil running dari program, dapat dilihat dengan penambahan nilai random akan menimbulan efek ‘noise’ pada gambar. Sedangkan dengan penambahan yang sama rata hanya akan mengubah ‘Brightness’ pada gambar.

Bab 13 No. 16

```
A = imread('flower','jpg');

maximum = max(max(max(A)))
minimum = min(min(min(A)))
```

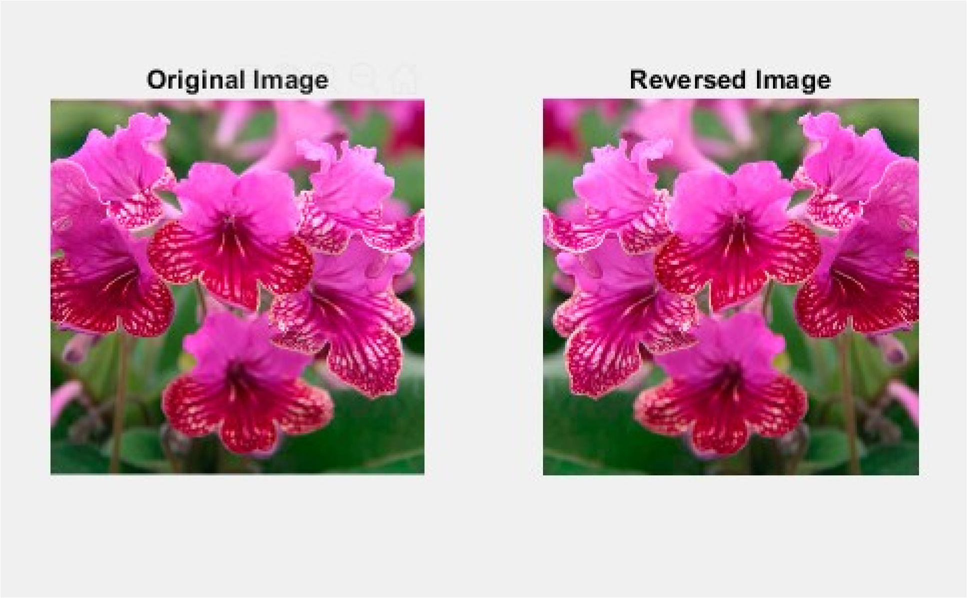
Bab 13 No. 17

```
orig = randi([0, 255], 4)
fin = orig + randi([-10,10],4)

mean(mean(orig))
mean(mean(fin))
```

Bab 13 No. 18

```
I1 =imread('flower.jpg');  %Mengkonversi gambar ke bentuk matrix
[rc,h] = size(I1); %Mengambil size dari matrix
Inew(:,:,:) = I1(:,rc:-1:1,:); %Mereverse image dengan mengkonstruksi matriks dari
belakang

% Melakukan Plotting

% Menggunakan Image biasa
figure(1)
subplot(2,1,1)
image(I1);
```

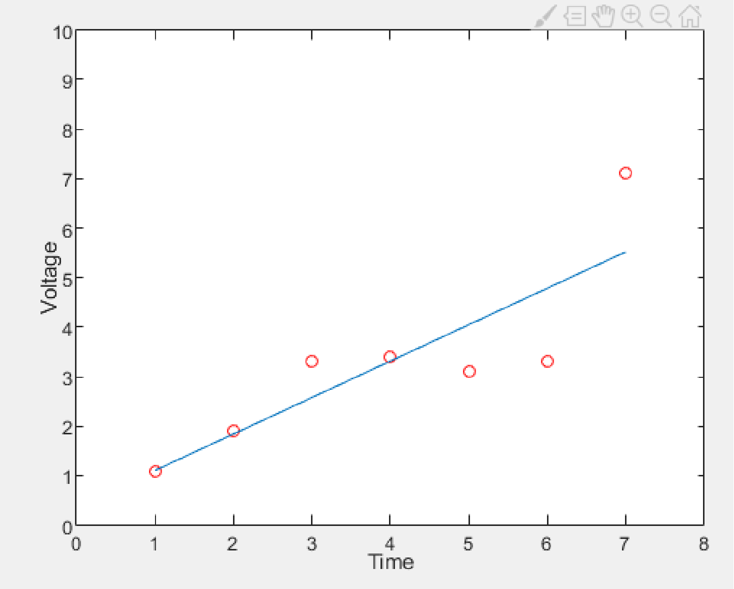
```
subplot(2,1,2)
image(Inew);

% Menggunakan imshow
subplot(1,2,1), imshow(I1)
title('Original Image')
subplot(1,2,2), imshow(Inew)
title('Reversed Image')
```



Program ini melakukan proses ‘flip horizontal’ dengan cara mengkonstruksi kembali matriks tersebut dengan urutan tiap vektor kolomnya dibalik.

Bab 14 No. 7

```
x= 1:7
y= [1.1, 1.9, 3.3, 3.4, 3.1, 3.3, 7.1]
coefs = polyfit(x,y,1);
curve = polyval(coefs,x);
plot(x,y,'ro',x,curve)
xlabel('Time')
ylabel('Voltage')
axis([0 8 0 10])
```



Bab 14 No.12

```
x= 1:4
y= [2, 5, 6, 10]
coefs2 = polyfit(x,y,1);
coefs2 = polyfit(x,y,2);
curve1 = polyval(coefs1,x);
curve2 = polyval(coefs2,x);
plot(x,y,'ro',x,curve1)
plot(x,y,'ro',x,curve2)
```