```
%% 0)
clc; % clears the command window
clear all; % clears all the stored variables
close all; % closes all open figures

%% 3)
v = 1:10; % row wector
w = [10:-1:1]'; %column vector
dot(v,w) % scalar(dot) product; with the use of matrix multiplication: v*w
% cross([1 2 3],[3 2 1]) % vector product

%% 4)
x1 = linspace(0,10,5)
x2 = logspace(1,3,3)

x1a = 0:2.5:10
x2a = 10.^[1:3]

%% 5)
A = [[1 1 1 1 1];
    [1 2 3 4 5];
    [1 3 6 10 15];
    [1 4 10 20 35];
    [1 5 15 35 70]]
A(1:4,3)
A(:,3)
A(1:4,[2,4])
A = A([2,3,3,4,5],:)
A = A(:,[2,3,3,4,5])

%% 6)
%clear all; %clearing all variables
A = [[1 2 3]; [4 5 6]; [7,8,9]]

size(A)
length(A(:))

B = flipud(A) % fliplr

x = B(:,1)
y = B(3,:)

A = A(:,[1,3])
A(:,2) = []

%% 7)
D = diag([2 2 2 2],0) + diag([-1 -1 -1],-1) + diag([-1 -1 -1],1)

%% 8)
A = [[9 -2];[3 1];[-3 7]]
B = [[2 -2];[-1 1];[4 4]]
C = A./B
D = A-B.^2
```

```
%% 9)
% rand() creates matrix with random numbers between 0 an 1
A = (rand(5)-0.5)*4
d = diag(A) % diagonal
ad = diag(flipud(A)) % antidiagonal

%% 10)
A = [[1 2 3];[0.1 0.2 0.3];[10 20 30]];
B = [[4 5 6];[0.4 0.5 0.6];[40 50 60]];

C = A+B
D = A-B
E = A.*B
F = A./B
G = A*B

%% 11)
v1 = 1:5;
v2 = 10*v1;
vs = v1+v2;
vd = v1-v2;
s = v1*v2'

%% 12a)
A = ones(3,2);
B = 2*ones(2,3);
A*B
A(2,3) = 2;
%A*B % command "A(2,3) = 2;" added a new column to the matrix A -> dimensions does↙
not agree

%% 12b)
u = 0:3;
v = (-3:-1:0)'; % incorrect array definition! -> "-1" should be replaced by "1"
W = u.*v

%% 13)
A = [[1 2]; [3 4]];
b = [3;7];
A+A
A-A
A*A
A^2
A*b
A.*A
A.^2
b*A(1,:)
A(:,2).*b

%% 14)
A = sin(1:10000);
sum(A>=1/2)
```

```
%% 15)
A = [
    [3 7 -4 12]
    [-5 9 10 2]
    [6 13 8 11]
    [15 5 4 1]
    ];
min(A')
max(max(A))

%% 16)
n = 5;
v = (1:5)*3

%% 17)
k = 5;
F = [1 1];
for i=2:k
    F(i+1) = F(i)+F(i-1);
end
disp(F)

%% 18)
k = 15;
F = [1 1];
for i=2:k
    F(i+1) = F(i)+F(i-1);
end
r_k = F(k)/F(k-1)
err = r_k-(1+(5^(1/2)))/2

%% 19)
%...

%% 22)
f = sinlog(1,100,6)

% function f = sinlog(a,b,h)
% if (a<=b && h>0)
%
%     x = a:h:b;
%     f = 2*sin(8*x)-log(x.*x +1);
%
% else
%     f = NaN;
% end
%
% end

%% 23)
fun = @(x)2*sin(8*x)-log(x.*x +1);
f = generalFunEval(fun,1,100,6)
```

```
% function f = generalFunEval(fun,a,b,h)
% if (a<=b && h>0)
%
%     x = a:h:b;
%     f = fun(x);
%
% else
%     f = NaN;
% end
%
% end

%% 24)
S1 = area(3,4)
S2 = area(5)

% function S = area(a,b)
%  if ~exist('b','var')
%       b=a;
%  end
%  S=a*b;
% end

%% 25)
figure
x = linspace(-1,1,1000);
y = x.^3;
plot(x,y)
title('x^3')
grid on;

figure
x = linspace(-2,5,1000);
y = exp(x);
plot(x,y)
title('exp(x)')
grid off;

figure
x = linspace(-20,20,1000);
y = sin(x)./x;
plot(x,y)
title('exp(x)')

%% 26)
figure
t = linspace(0,2*pi,1000);
y1 = sin(t);
y2 = sin(t+(2/3)*pi);
y3 = sin(t+(4/3)*pi);
y = max(y1,y2);
y = max(y,y3);
```

```
plot(t,y)
xlabel('x axis')
ylabel('y axis')

%% 27)
figure
x = linspace(0,pi,1000);
plot(x,sin(x),'r--')
hold on
plot(x,cos(x),'k:')
plot(x,sin(x).*cos(x),'b')
title('goniometricke funkce')
legend('sin','cos','sincos')
grid on
%axis([0 pi -1.5 2.5])

%% saving and loading data
a = 3;
b = [4 5 6];
c = 'slovo';
save('naseData','a','b','c')

% clear all
% load('naseData')
```