

Octave alapok

1. BEVEZETŐ

Az Octave egy numerikus számításokra és grafikus megjelenítésre kihegyezett nyílt forráskódú (ingyenes) programozási környezet.

Az Octave elsősorban nem szimbolikus számításokra alkalmazandó (ellentétben pl. a Mathematica-val, vagy a Maple-vel), hanem matematikai problémák numerikus megoldására, ami miatt nem mindig ad egzakt megoldást (a mérnöki problémák döntő többségében ilyen nem is létezik)

Az Octave szintaxisában nagyon hasonlít a Matlab-hoz, a különbség, hogy a Matlab kicsit kötöttebb az Octave-hoz képest, így a Matlab-ban írt kódot az Octave lefuttatja, az Octave-ban írt kódot viszont nem biztos hogy lefuttatja a Matlab (csak ha szigorúan a Matlab szintakszisa szerint lett írva a program)

Octave letöltése: <https://www.gnu.org/software/octave/download.html>

Grafikus felület

Az Octave-nak a grafikus felülete is nagyon hasonlít a Matlab-éhoz. A grafikus felület fontosabb részei: az éppen aktuális könyvtár, ahová mindent ment a program (current folder/directory), a parancssor (command window), a munkakörnyezet (workspace) az éppen használt változókkal, az eddig futtatott parancsok (command history) és a szerkesztő (editor). Az adott panel nevére kattintva, lenyomva tartott bal egér gombbal áthúzhatóak a panelek, és a mozgatás során a kékre színezett területre helyezhetők.

Octave mint egyszerű számológép

A parancsablakba különböző műveleteket beírva az Octave egyszerű számológépként is használható. Minden szokásos aritmetikai művelet használható: + - * / ^ (a legutolsó a hatványozás szimbóluma)

```
Command Window
>> 2+5
ans = 7
>> 8-3
ans = 5
>> 5*6
ans = 30
>> 4/7
ans = 0.57143
>> 2^3
ans = 8
```

2. BEÉPÍTETT FÜGGVÉNYEK

Az aritmetikai műveletekhez hasonlóan számos függvény is be van építve az Octave-ba. A leggyakrabban használt függvények:

cos	szög koszinusza radiánban
sin	szög szinusza radiánban
tan	szög tangense radiánban
exp	exponenciális függvény (e^x)
log	természetes alapú logaritmus
log10	10-es alapú logaritmus
sinh	szimónusz hiperbolikus
cosh	koszinusz hiperbolikus
tanh	tangens hiperbolikus
acos	arkusz koszinusz
acosh	arkusz koszinusz hiperbolikus
asin	arkusz szinus
asinh	arkusz szinus hiperbolikus
atan	arkusz tangens
atan2	általánosított arkusztangens (x tengellyel bezárt szög)
atanh	arkusz tangens hiperbolikus
abs	szám abszolút értéke
sign	szignum
round	kerekítés legközelebbi egészre
floor	lekerekítés
ceil	felkerekítés
fix	0 felé kerekítés
rem	osztás maradéka

függvények használata: a függvény után zárójelbe írjuk be az argumentumot.

Command Window

```
>> exp(1)
ans = 2.7183
>> sin(30)
ans = -0.98803
>> sin(30*pi/180)
ans = 0.50000
>> 1.2*sin(40*pi/180+log(2.4^2))
ans = 0.76618
>> abs(-5)
ans = 5
>> round(3.8)
ans = 4
>> floor(3.8)
ans = 3
>> fix(3.8)
ans = 3
>> rem(5,6)
ans = 5
>> rem(15,4)
ans = 3
```

Néhány hasznos parancs:

clear	törli a memóriát
clc	törli a parancsablakot
ctrl+C	félbeszakítja az adott parancsot
page_screen_output(0)	A műveletek eredményét folyamatosan jelenítse meg, ne oldalanként törölve

3. A SZERKESZŐ (EDITOR)

Ha egy programot szeretnénk írni, akkor azt az editor (szerkesztő) ablakban tudjuk megtenni. Az ide beírt programot az F5 billentyűvel tudjuk lefuttatni. Az Octave alpból minden művelet eredményét megjeleníti a parancsablakban. Ha valaminek nem szeretnénk látni az eredményét, a sor végére rakjunk egy pontos vesszőt. A szerkesztőbe írhatunk kommenteket is, melyeket a program nem vesz figyelembe futtatás során, ezeket %-kal #-tel kezdjük. Több sor együttes kikommentelése a Ctrl+R, visszakommentelése a Ctrl+Shift+R billentyűkombinációt használhatjuk

```
tutorial.m x
1  #Összeadás:
2  2+2
3  5+8;
4  4+1

Command Window
>> tutorial
ans = 4
ans = 5
>> |
```

4. VÁLTOZÓK

A számításokhoz definiálhatunk változókat, amiket bárminek elnevezhetünk, de:

- kerüljük az ékezeteket
- pi-t ne definiáljunk, mert az már be van építve

Minden változónak van egy hatásköre, hogy hol látszódik. Pl. ha egy változót egy for ciklusban, vagy function-ban definiálunk, az csak ott lesz látható a program számára. Ha azt akarjuk, hogy egy változó mindenhol látható legyen, írjuk eléje, hogy *global*.

```
* tutorial.m x
1  a=1;
2  b=2;
3  c=a+b

Command Window
>> tutorial
c = 3
>> |
```

5. MÁTRIXOK

5.1 Mátrixok definiálása

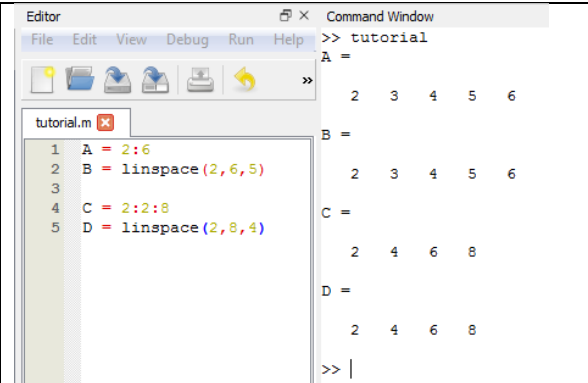
a) direkt megadás

A mátrixokat szögletes zárójellel definiáljuk, az egymás melletti elemeket vesszővel vagy szóközzel, míg az egyes sorokat pontosvesszővel vagy enterrel választjuk el egymástól.

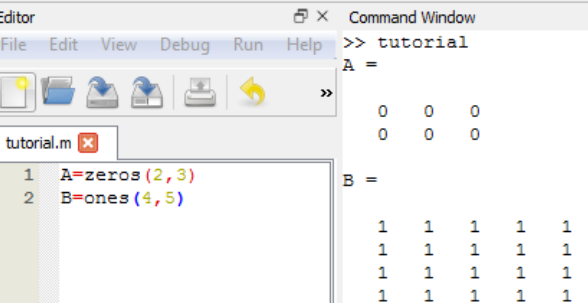
```
File Edit View Debug Run Help
>> tutorial
A =
     1     2     3     4
B =
     1     2     3     4
C =
     1
     2
     3
     4
D =
     1
     2
     3
     4
E =
     1     2     3
     4     5     6
F =
     1     2     3
     4     5     6
>> |

* tutorial.m x
1  #sormátrixok:
2  A = [1,2,3,4]
3  B = [1 2 3 4]
4
5  #oszlopmátrixok:
6  C = [1;2;3;4]
7  D = [1
8       2
9       3
10      4]
11
12 #2x3-mas mátrixok:
13 E = [1,2,3;4,5,6]
14 F = [1 2 3
15      4 5 6]
16
17
```

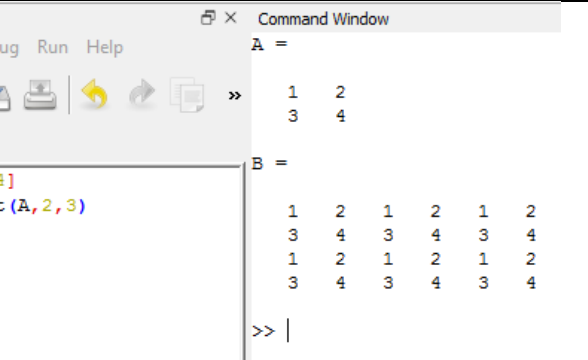
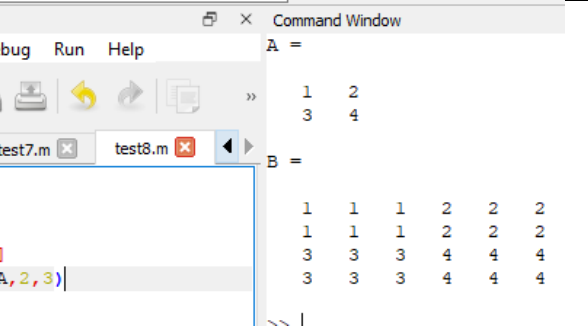
b) megadás tartománnyal

A megadás módja: kezdőérték:növekmény (opcionális, ha nem adjuk meg akkor 1):utolsó érték vagy linspace(kezdőérték, utolsó érték, elemek száma)	 The screenshot shows the MATLAB Editor with a script named 'tutorial.m' containing the following code: <pre>1 A = 2:6 2 B = linspace(2,6,5) 3 4 C = 2:2:8 5 D = linspace(2,8,4)</pre> The Command Window shows the output of these commands: <pre>>> tutorial A = 2 3 4 5 6 B = 2 3 4 5 6 C = 2 4 6 8 D = 2 4 6 8 >> </pre>
---	---

c) speciális mátrixok

nullmátrix: zeros(sorok száma, oszlopok száma) egyeseket tartalmazó mátrix: ones(sorok száma, oszlopok száma) egységmátrix: eye(mátrix mérete)	 The screenshot shows the MATLAB Editor with a script named 'tutorial.m' containing the following code: <pre>1 A=zeros(2,3) 2 B=ones(4,5)</pre> The Command Window shows the output of these commands: <pre>>> tutorial A = 0 0 0 0 0 0 B = 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 >> </pre>
---	---

d) mátrixmegadás ismétlő parancsokkal

A B= repmat(A,m,n) parancs: az A mátrixot rakja m-szer egymás alá, majd a kapott blokk-oszlopot n-szer egymás mellé.	 The screenshot shows the MATLAB Editor with a script named 'test1.m' containing the following code: <pre>1 A=[1,2;3,4] 2 B = repmat(A,2,3)</pre> The Command Window shows the output of these commands: <pre>>> A = 1 2 3 4 B = 1 2 1 2 1 2 3 4 3 4 3 4 1 2 1 2 1 2 3 4 3 4 3 4 >> </pre>
A B= repelem(A,m,n) parancs: az A mátrix elemeit rakja m-szer egymás alá, majd a kapott blokk-oszlopot n-szer egymás mellé.	 The screenshot shows the MATLAB Editor with a script named 'test6.m' containing the following code: <pre>1 clear all 2 clc 3 4 A=[1,2;3,4] 5 B=repelem(A,2,3)</pre> The Command Window shows the output of these commands: <pre>>> A = 1 2 3 4 B = 1 1 1 2 2 2 1 1 1 2 2 2 3 3 3 4 4 4 3 3 3 4 4 4 >> </pre>

e) diagonál mátrix definiálása:

diag(diagonálemeket tartalmazó oszlopmátrix, eltolás a főátlótól)

The screenshot shows the MATLAB Editor with a script named 'tutorial.m' and the Command Window. The script defines matrices A, B, C, D, and E. The Command Window displays the results of the operations.

```

Editor
File Edit View Debug Run Help
tutorial.m
1 A = ones(4,1)
2 B=ones(3,1)*2
3
4 C=diag(A,0)
5 D=diag(B,1)
6
7 E=C+D
8
9
10
11

Command Window
>> tutorial
A =
1
1
1
1
B =
2
2
2
C =
Diagonal Matrix
1 0 0 0
0 1 0 0
0 0 1 0
0 0 0 1
D =
0 2 0 0
0 0 2 0
0 0 0 2
0 0 0 0
E =
1 2 0 0
0 1 2 0
0 0 1 2
0 0 0 1
>> |
  
```

5.2 Mátrix mérete

size(mátrix jele) (hányadik dimenzió)

vagy

size(mátrix jele, hányadik dimenzió)

The screenshot shows the MATLAB Editor with a script named 'tutorial.m' and the Command Window. The script defines matrix A and uses the 'size' function to get its dimensions. The Command Window displays the results.

```

Editor
File Edit View Debug Run Help
tutorial.m
1 A=[1,2,3;4,5,6]
2
3 a=size(A)
4 b=size(A)(1)
5 c=size(A)(2)
6
7
8

Command Window
>> tutorial
A =
1 2 3
4 5 6
a =
2 3
b = 2
c = 3
>> |
  
```

5.3 Részmatrixok

mátrix jele(hányadik
sorról: mekkora
lépésközzel: hányadik
sorig, hányadik
oszloptól: mekkora
lépésközzel: hányadik
oszlopig)
Ha a lépésközt kihagyjuk, az
alapértelmezésként 1

The screenshot shows the MATLAB Editor with a script named 'tutorial.m' and the Command Window. The script defines a 3x3 matrix A and extracts submatrices a, b, c, d, and e using indexing.

```

1 A=[1,2,3;4,5,6;7,8,9]
2
3 a=A(1:2,1:2)
4 b=A(1:2:3,1:2:3)
5 c=A(2,1) #2,1-es elemet adja vissza
6 d=A(2,:) #2. sort adja vissza
7 e=A(:,3) #3. oszlopot adja vissza

```

The Command Window displays the results of these operations:

```

>> tutorial
A =
     1     2     3
     4     5     6
     7     8     9

a =
     1     2
     4     5

b =
     1     3
     7     9

c = 4
d =
     4     5     6

e =
     3
     6
     9

```

5.4 Mátrixműveletek:

The screenshot shows the MATLAB Editor with a script named 'tutorial.m' and the Command Window. The script performs various matrix operations including transpose, addition, subtraction, multiplication, and inversion.

```

1 A=[1,2,3;4,5,6;7,8,9]
2 B=[7,3,5;2,1,8;3,4,7]
3
4 #Mátrix transzponáltja:
5 C=transpose(A)
6 D=A'
7
8 E=A+B #összeadás
9 F=A-B #kivonás
10
11 G=A*B #mátrixszorzás
12 H=A.*B #szorzás elemenként
13
14 I=A^2 #mátrix hatványozás
15 J=A.^2 #hatványozás elemenként
16
17 #Mátrix invertálás:
18 K=inv(B)
19 L=inv(B)*B
20 M=inv(B)*A
21 N=B\A
22
23 #determináns
24 O=det(B)

```

The Command Window displays the results of these operations:

```

>> tutorial
A =
     1     2     3
     4     5     6
     7     8     9

B =
     7     3     5
     2     1     8
     3     4     7

C =
     1     4     7
     2     5     8
     3     6     9

D =
     1     4     7
     2     5     8
     3     6     9

E =
     8     5     8
     6     6    14
    10    12    16

F =
    -6    -1    -2
     2     4    -2
     4     4     2

G =
    20    17    42
    56    41   102
    92    65   162

H =
     7     6    15
     8     5     8
    21    32    63

I =
    30    36    42
    66    81    96
   102   126   150

J =
     1     4     9
    16    25    36
    49    64    81

K =
    0.2083333    0.0083333   -0.1583333
   -0.0833333   -0.2833333    0.3833333
   -0.0416667    0.1583333   -0.0083333

L =
    1.00000    0.00000    0.00000
    0.00000    1.00000    0.00000
    0.00000    0.00000    1.00000

M =
   -0.86667   -0.80833   -0.75000
    1.46667    1.48333    1.50000
    0.53333    0.64167    0.75000

N =
   -0.86667   -0.80833   -0.75000
    1.46667    1.48333    1.50000
    0.53333    0.64167    0.75000

O = -120

```

6. VEZÉRLÉSI SZERKEZETEK

A vezérlési szerkezetek segítségével az egyes utasítások végrehajtási sorrendjét határozhatjuk meg.

6.1 for ciklus

A for utasítás egy kódrész többszöri ismétlésére használható. A for utasítás jó módszer egy értéktartomány (mátrix) bejárására. A for utasítás általános formája a következőképpen néz ki:

```
for helyi változó neve = első érték : növekmény (opcionális) :  
    utasítások  
endfor
```

A for ciklus helyi változóit általában az *i*, *j*, *k*, *l* ... betűkkel szoktuk jelölni.

példák:

<pre>tutorial.m x 1 a=0 2 for i=1:5 3 a=a+i 4 endfor</pre>	<pre>tutorial.m x 1 for i=1:2:10 2 i 3 endfor</pre>	<pre>tutorial.m x 1 for i=1:5 2 faktorialis=factorial(i) 3 endfor</pre>
<pre>Command Window >> tutorial a = 0 a = 1 a = 3 a = 6 a = 10 a = 15 >> </pre>	<pre>Command Window >> tutorial i = 1 i = 3 i = 5 i = 7 i = 9 >> </pre>	<pre>Command Window >> tutorial faktorialis = 1 faktorialis = 2 faktorialis = 6 faktorialis = 24 faktorialis = 120 >> </pre>
<pre>tutorial.m x 1 for i=1:5 2 faktorialis(i)=factorial(i); 3 endfor 4 faktorialis</pre> line: 4 col: 12 encoding: SYSTEM eol: CRLF <pre>Command Window >> tutorial faktorialis = 1 2 6 24 120</pre>	<pre>tutorial.m x 1 a=1; 2 b=2; 3 for i=1:2 4 A(1,3*i)=a; 5 A(2,3*i-1)=b; 6 endfor 7 A</pre> line: 7 col: 2 encoding: SYSTEM <pre>Command Window >> tutorial A = 0 0 1 0 0 1 0 2 0 0 2 0</pre>	

6.2 feltétel vizsgálat

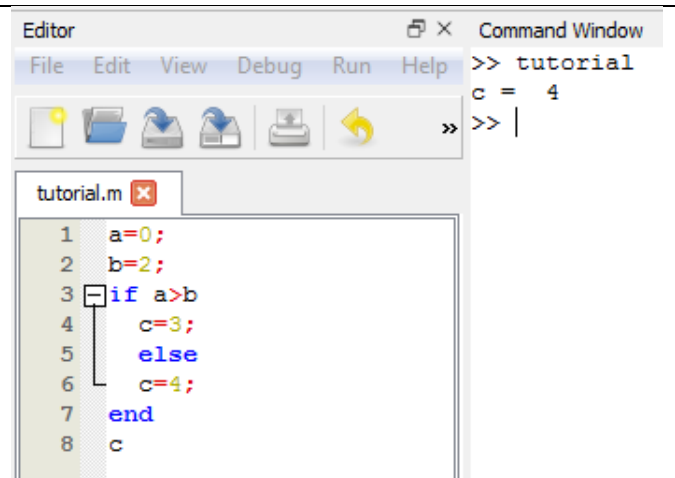
egyik lehetőség:

Általános felépítés:

```
if feltétel
    utasítás
elseif feltétel
    utasítás
else feltétel
    utasítás
endif
```

feltételek megadása:

egyenlő	if a == b
nem egyenlő	if a ~= b
nagyobb	if a > b
nagyobb egyenlő	if a >= b
kisebb	if a < b
kisebb egyenlő	if a <= b
és	if a > b && b > 2
vagy	if a > b b > 2



The screenshot shows the MATLAB Editor with a file named 'tutorial.m'. The code in the editor is:

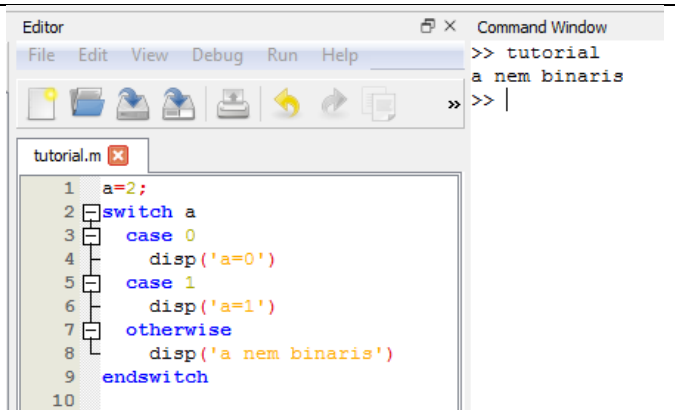
```
1 a=0;
2 b=2;
3 if a>b
4     c=3;
5 else
6     c=4;
7 end
8 c
```

The Command Window on the right shows the execution of the script:

```
>> tutorial
c = 4
>>
```

másik lehetőség (ha 1 változó különböző értékekkel rendelkezhet):

```
switch a
    case 1
        utasítások
    case 2
        utasítások
    otherwise
        utasítások
endswitch
```



The screenshot shows the MATLAB Editor with a file named 'tutorial.m'. The code in the editor is:

```
1 a=2;
2 switch a
3     case 0
4         disp('a=0')
5     case 1
6         disp('a=1')
7     otherwise
8         disp('a nem binaris')
9 endswitch
10
```

The Command Window on the right shows the execution of the script:

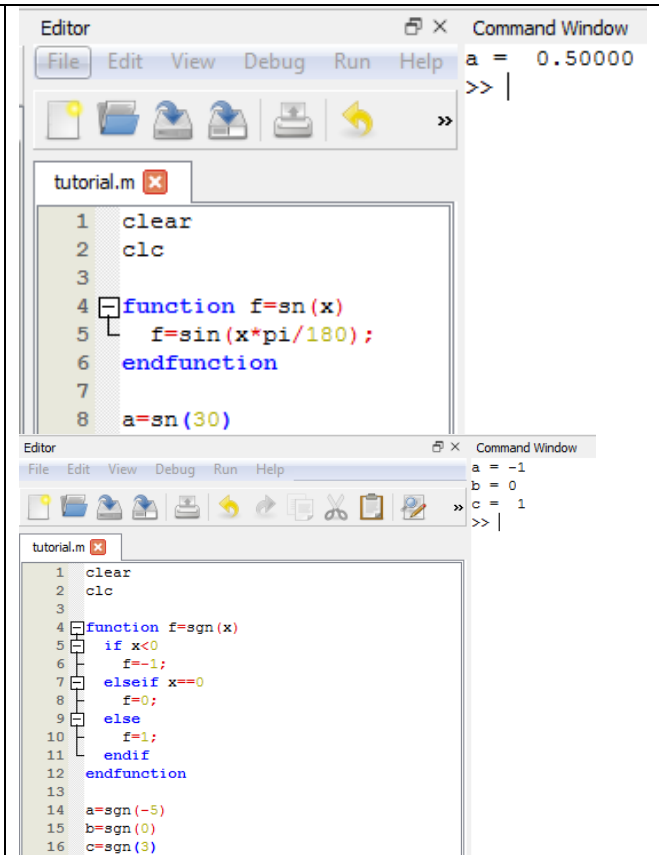
```
>> tutorial
a nem binaris
>>
```


7. FÜGGVÉNYEK

A függvénnyel egy saját Octave parancsot definiálhatunk, amit aztán a parancsablakban vagy a szkriptben bármikor felhasználhatunk.

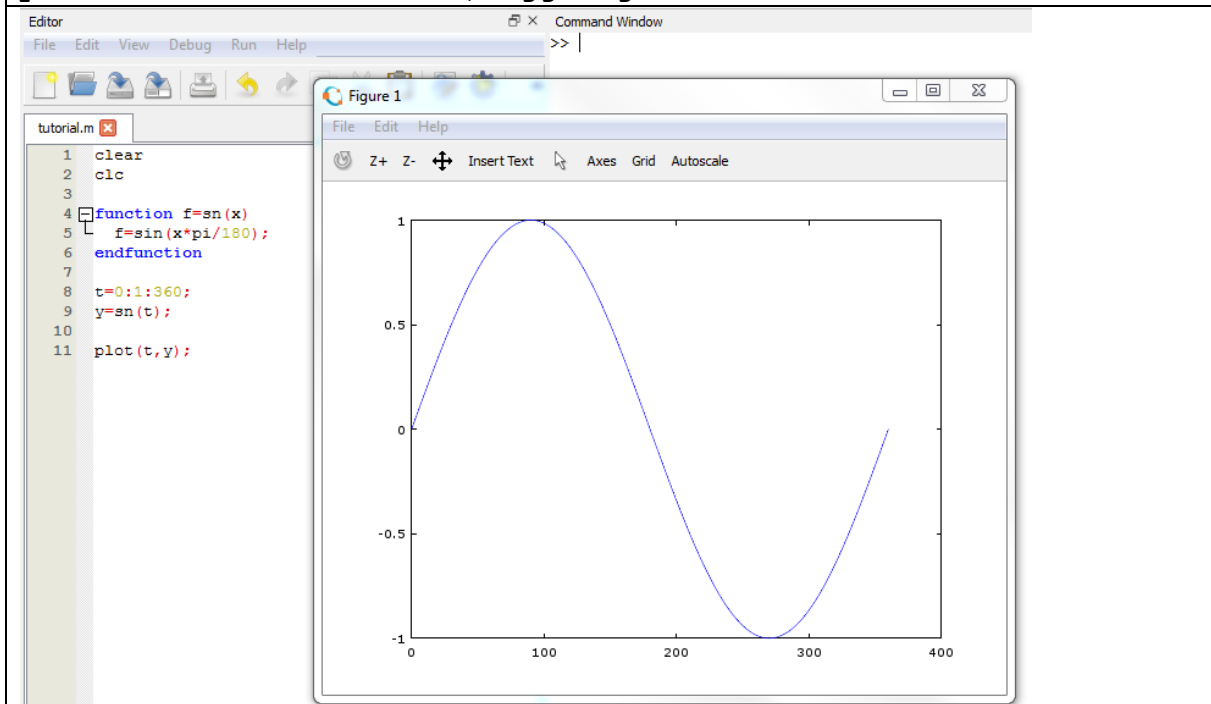
Felépítés:

```
function f = név(változó1,  
    változó2, ...)  
    f=...  
endfunction
```



8. 2D FÜGGVÉNYÁBRÁZOLÁS

plot(vízszintes változó, függőleges változó)



8.1 Vonallajdonságok beállítása

A plot argumentumában, a változók után vesszővel elválasztva soroljuk fel a tulajdonságokat

a) vonal színe

'Color', színdefiníció

A színeket az alábbi módon definiálhatjuk:

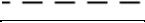
szín neve	rövidített enev	RGB kódja	megjelenés
'red'	'r'	[1 0 0]	
'green'	'g'	[0 1 0]	
'blue'	'b'	[0 0 1]	
'cyan'	'c'	[0 1 1]	
'magenta'	'm'	[1 0 1]	
'yellow'	'y'	[1 1 0]	
'black'	'k'	[0 0 0]	
'white'	'w'	[1 1 1]	
'none'			nincs szín

RGB kóddal természetesen bármilyen tetszőleges szín előállítható.

b) vonal stílusa

'LineStyle', vonalstílusdefiníció

A vonalak stílusát az alábbi módon definiálhatjuk:

definíció	leírás	megjelenés
'_'	folytonos vonal	
'--'	szaggatott vonal	
'::'	pontozott vonal	
'-.'	pontvonal	
'none'	nincs vonal	nincs vonal

c) vonalvastagság

'LineWidth', vastagság pontban (1 pont=1/72 col)

d) markerek

Ezzel az adatpontokhoz különböző szimbólumokat rakhatunk.

'Marker', definíció

A markerek definíciói az alábbiak lehetnek:

definíció	leírás	definíció	leírás
'o'	kör	'diamond' or 'd'	rombusz
'+'	plusz jel	'^'	felfele álló háromszög
'*'	csillag	'v'	lefele álló háromszög
'.'	pont	'>'	jobbra álló háromszög
'x'	kerszt	'<'	balra álló háromszög
'_'	vízszintes vonal	'pentagram' or 'p'	5 ágú csillag
' '	függőleges vonal	'hexagram' or 'h'	6 ágú csillag
'square' or 's'	négyszet	'none'	nincs marker

e) marker méret

'MarkerSize', markerméret pontban (1 pont=1/72 col)

f) marker határolóvonalának színe

'MarkerEdgeColor', színdefiníció (lásd vonal színe)

g) marker kitöltőszíne

'MarkerFaceColor', színdefiníció (lásd vonal színe)

8.2 Tengely tulajdonságok beállítása

Beállítás módja: set(gca,tulajdonságok)

a) felirat tulajdonságok:

'FontName', betűtípus neve
'FontWeight', 'normal' (alapértelmezett) vagy 'bold'
'FontSize', betűméret
'FontAngle', 'normal' (alapértelmezett) vagy 'italic'

b) tengelyhatárok

xlim([kezdőérték,végérték])
ylim([kezdőérték,végérték])

c) segédvonalak

xgrid on vagy off
ygrid on vagy off
grid on vagy off

d) cím

title('név')

g) jelmagyarázat

legend('név1','név2')

h) tengelyfeliratok

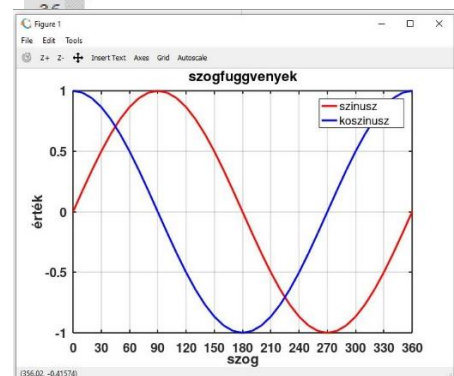
xlabel('név')
ylabel('név')

j) azonos lépték a 2 tengelyen
axis equal

```

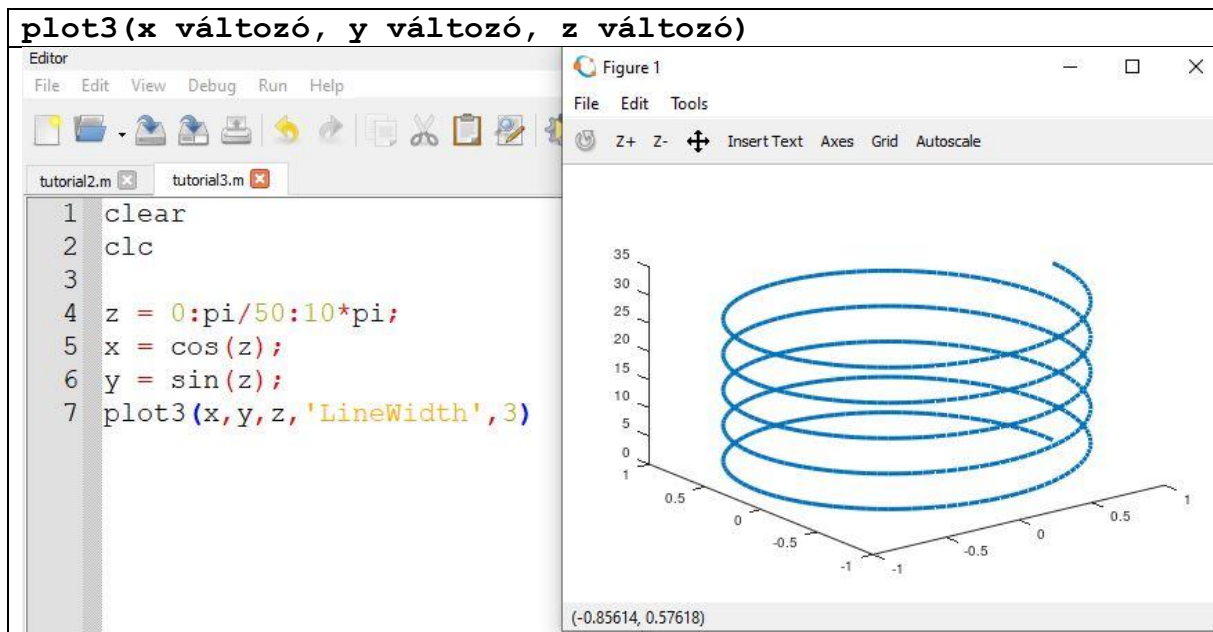
1 clear
2 clc
3
4 function f=sn(x)
5     f=sin(x*pi/180);
6 endfunction
7
8 function f=cs(x)
9     f=cos(x*pi/180);
10 endfunction
11
12 t=0:10:360;
13 y1=sn(t);
14 y2=cs(t);
15
16 plot(t,y1,'Color','red',...
17       'LineWidth',3)
18
19 hold on
20
21 plot(t,y2,'Color','blue',...
22       'LineWidth',3)
23
24 set(gca,'LineWidth',2,...
25         'FontSize',20,...
26         'FontWeight','bold',...
27         'xtick',[0:30:360])
28 title('szogfuggvenyek')
29 xlim([0,360])
30 ylim([-1,1])
31 xlabel('szog')
32 ylabel('érték')
33 grid on
34 legend('szinusz','koszinusz')
35

```

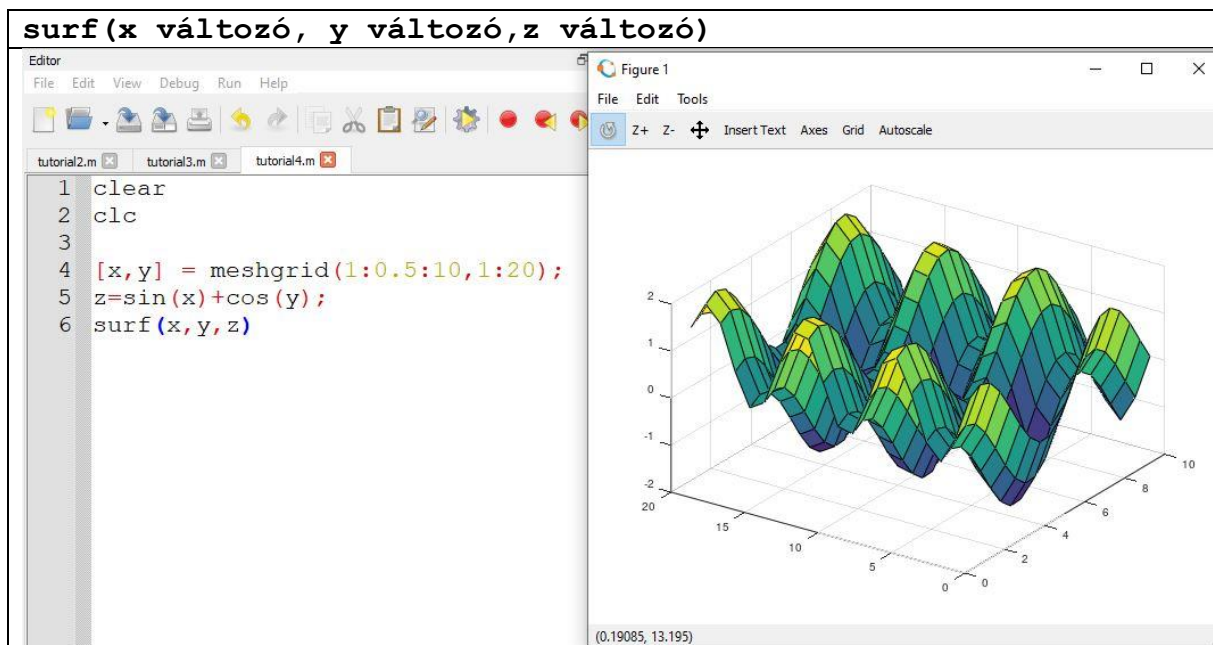


9. 3D FÜGGVÉBYÁBRÁZOLÁS

9.1 3D vonal



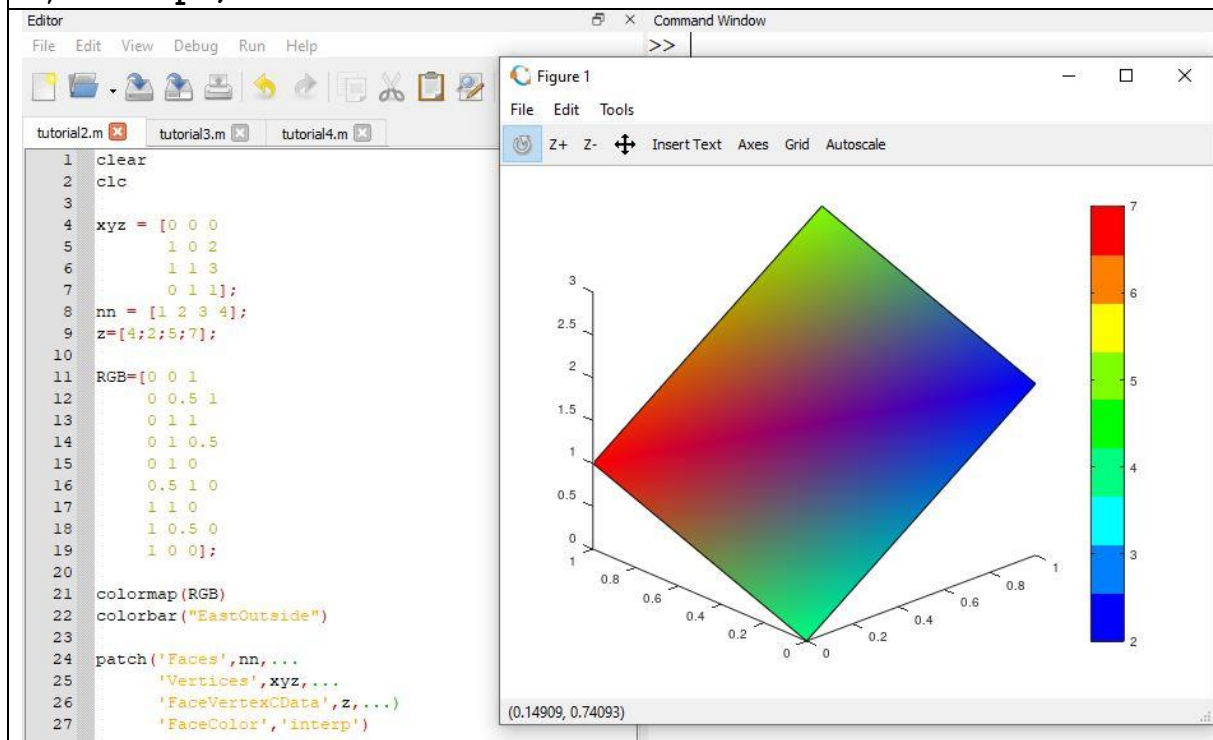
9.2 3D felület



A **meshgrid** függvény bemenetként vár két vektort és kimenete két mátrix, melyek megadják az x és y koordinátákat egy képen.

9.3 3D színezett sokszög

```
patch('Faces',nn,'Vertices',xyz,'FaceVertexCData',z,'FaceColor', 'interp')
```



Az xyz mátrix a sokszög csúcspontjainak koordinátáit tartalmazza, az nn mátrix a csúcspontok sorszámainak tartalmazza, a z mátrix a csúcspontokhoz rendelt jellemzők mátrixa, mely a színezést határozza meg.

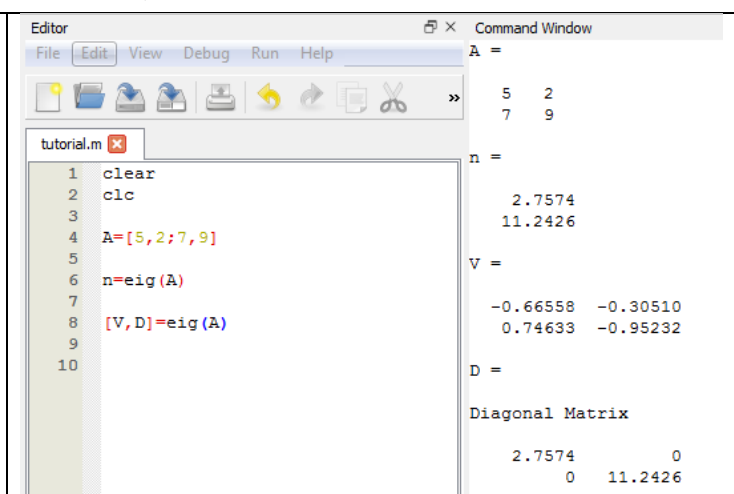
10. SAJÁTÉRTÉK, SAJÁTVEKTOR

$Aq = \lambda q$, ahol:

- λ : sajátérték
- q : sajátvektor

alkalmazás:

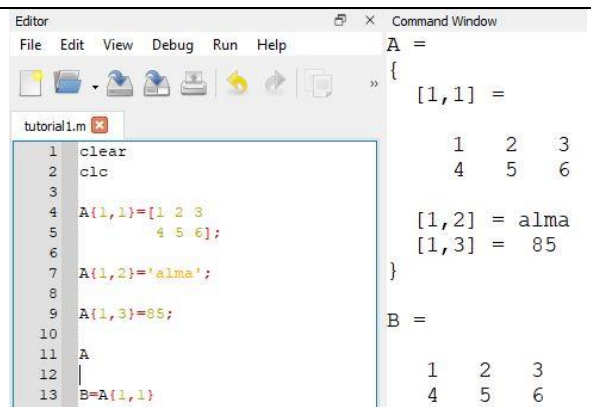
- fő tehetlenségi nyomaték számítás
- főfeszültség számítás
- modálanalízis



11. SPECIÁLIS ADATTÍPUSOK

11.1 cella tömb

A cella tömb egy olyan tömb amelynek celláihoz különböző típusú és méretű elemeket rendelhetünk, pl. egy egészet, vagy egy mátrixot vagy éppen egy másik cella tömböt. Kezelésük hasonlóan történik a mátrixokéhoz annyi különbséggel, hogy hivatkozásoknál a kerek zárójelek, helyet a kapcsosat kell használni.



The image shows a MATLAB Editor window with a script named 'tutorial1.m' and a Command Window. The script contains the following code:

```
1 clear
2 clc
3
4 A{1,1}=[1 2 3
5         4 5 6];
6
7 A{1,2}='alma';
8
9 A{1,3}=85;
10
11 A
12
13 B=A{1,1}
```

The Command Window displays the output of the script:

```
A =
{
    [1,1] =
         1     2     3
         4     5     6

    [1,2] = alma
    [1,3] = 85
}

B =
     1     2     3
     4     5     6
```

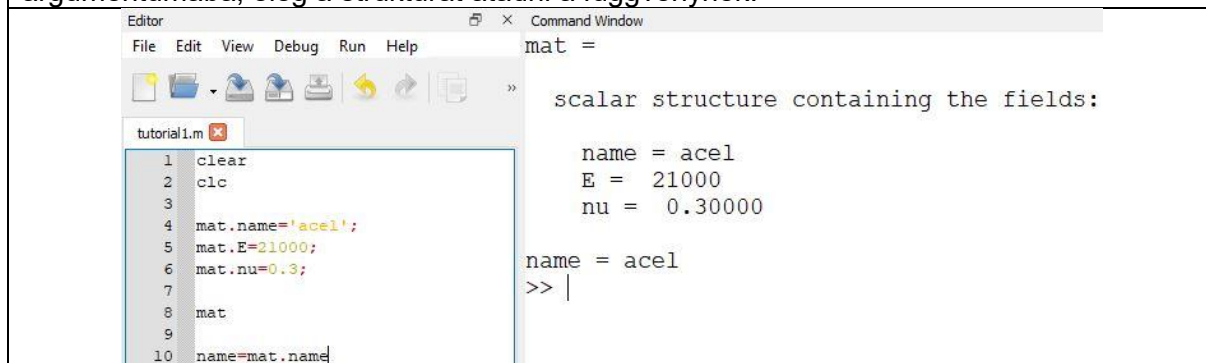
11.2 struktúra

A struktúra tömb egy olyan tömb amelynek elemeire (mezőire) névvel hivatkozhatunk és eltérő típusú adatokat tárolhatunk ezekben a mezőkben. Létrehozása:

változónév.mezőnév=érték vagy:

struct('mező1',érték1,'mező2',érték2...)

Struktúra alkalmazásával nem kell az összes mezőt egyesével bevinni egy függvény argumentumába, elég a struktúrát átadni a függvénynek.



The image shows a MATLAB Editor window with a script named 'tutorial1.m' and a Command Window. The script contains the following code:

```
1 clear
2 clc
3
4 mat.name='acel';
5 mat.E=21000;
6 mat.nu=0.3;
7
8 mat
9
10 name=mat.name
```

The Command Window displays the output of the script:

```
mat =
scalar structure containing the fields:

    name = acel
         E = 21000
         nu = 0.30000

name = acel
>> |
```


12. VEKTORIZÁLÁS

Habár ciklusok és if feltételek rendkívül hasznosak, és sok esetben nélkülözhetetlenek, az Octave-ban, ha van rá mód használatukat célszerű elkerülni, mivel jelentősen csökkennek a program futási sebességét. Kiváltásuk az ún. vektorizáció segítségével lehetséges, melynek lényege, hogy a ciklusok/if feltételek helyett az Octave beépített mátrix operációit alkalmazzuk.

The screenshot shows the Octave IDE with the following script in the Editor:

```

1 clear
2 clc
3
4 % mátrix szorzása skalárral:
5 A=[2 3 4
6     7 1 0
7     8 2 8];
8
9 % mátrix szorzása skalárral:
10 B=A*2
11
12 % mátrix, mint függvény argumentum:
13 C=sin(A)
14
15 % feltételvizsgálat mátrixra:
16 D=A==3
17 E=A>4
18
19 % a 4-nél nagyobb elemek:
20 F=A(E)
21
22 % a 4-nél nagyobb elemek indexei:
23 [i,j]=find(E)
24
25 % van az A mátrixban nemnulla elem?
26 G=any(any(A))
27
28 % minden elem 0-tól különböző az A mátrixban?
29 H=all(all(A))
30
31 % mátrix eleminek összege
32 J=sum(sum(A))
  
```

The Command Window displays the following results:

```

      4      6      8
     14      2      0
     16      4     16

C =

    0.90930    0.14112   -0.75680
    0.65699    0.84147    0.00000
    0.98936    0.90930    0.98936

D =

     0     1     0
     0     0     0
     0     0     0

E =

     0     0     0
     1     0     0
     1     0     1

F =

     7
     8
     8

i =

     2
     3
     3

j =

     1
     1
     3

G = 1
H = 0
J = 35
  
```

halmazműveletek:

beletartozás vizsgálat: **ismember**
 metszet képzés: **intersect**
 unió: **union**
 különbség: **setdiff**
 egyedi elemek megtartása: **unique**
 azon elemek, melyek csak pontosan
 egyik halmazban találhatók meg:
setxor

```

1 clear all
2 clc
3
4 A=[1 2 5 6 6 8]
5 B=[2 3 4 5]
6
7 c=ismember(2,A)
8 d=intersect(A,B)
9 e=union(A,B)
10 f=setdiff(A,B)
11 g=unique(A)
12 h=setxor(A,B)
    
```

Command Window output:

```

A =
     1     2     5     6     6     8

B =
     2     3     4     5

c = 1
d =
     2     5

e =
     1     2     3     4     5     6     8

f =
     1     6     8

g =
     1     2     5     6     8

h =
     1     3     4     6     8
    
```

13. EGYSZERŰ FELADATOK

13.1 Lineáris algebrai egyenletrendszer megoldás

Pl: oldjuk meg az alábbi
 egyenletrendszert:

$$\begin{cases} 5x_1 + 2x_2 = 3 \\ 7x_1 + 9x_2 = 6 \end{cases}$$

A megoldáshoz az egyenletet
 mátrixos alakban kell megadnunk:

$$\underbrace{\begin{bmatrix} 5 & 2 \\ 7 & 9 \end{bmatrix}}_{\underline{\underline{A}}} \underbrace{\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}_{\underline{\underline{x}}} = \underbrace{\begin{bmatrix} 3 \\ 6 \end{bmatrix}}_{\underline{\underline{b}}}$$

$$\underline{\underline{A}}\underline{\underline{x}} = \underline{\underline{b}}$$

$$\underline{\underline{A}}^{-1}\underline{\underline{A}}\underline{\underline{x}} = \underline{\underline{A}}^{-1}\underline{\underline{b}}$$

$$\underline{\underline{E}}$$

$$\underline{\underline{x}} = \underline{\underline{A}}^{-1}\underline{\underline{b}}$$

```

1 clear
2 clc
3
4 A=[5,2;7,9]
5 b=[3;6]
6
7 x=inv(A)*b
8 y=A\b
    
```

Command Window output:

```

A =
     5     2
     7     9

b =
     3
     6

x =
    0.48387
    0.29032

y =
    0.48387
    0.29032

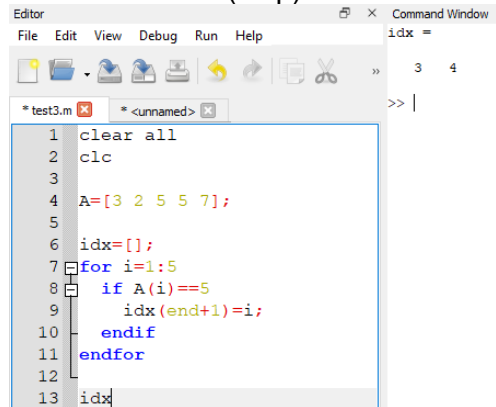
>> |
    
```


13.2 Egy adott érték indexének megkeresése egy mátrixban

Pl: keressük meg, hol szerepel az 5 az alábbi mátrixban:

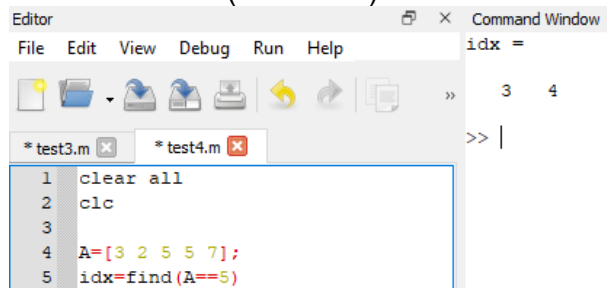
$A = \begin{bmatrix} 3 & 2 & 5 & 5 & 7 \end{bmatrix}$

1. módszer (loop)



```
1 clear all
2 clc
3
4 A=[3 2 5 5 7];
5
6 idx=[];
7 for i=1:5
8     if A(i)==5
9         idx(end+1)=i;
10    endif
11 endfor
12
13 idx
```

2. módszer (vektorizált)

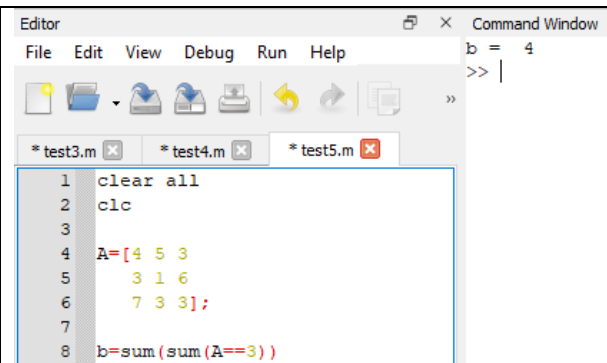


```
1 clear all
2 clc
3
4 A=[3 2 5 5 7];
5 idx=find(A==5)
```

13.3 Egy adott elem hányszor fordul elő egy mátrixban?

Pl: A 3-mas szám hányszor fordul elő az alábbi mátrixban?

$A = \begin{bmatrix} 4 & 5 & 3 \\ 3 & 1 & 6 \\ 7 & 3 & 3 \end{bmatrix}$



```
1 clear all
2 clc
3
4 A=[4 5 3
5     3 1 6
6     7 3 3];
7
8 b=sum(sum(A==3))
```