SZÉCHENYI EGYETEM
UNIVERSITY OF GYŐR

ALKALMAZOTT MECHANIKA TANSZÉK

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 helyezhetőek.

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 hiperbolikusz
cosh	koszinusz hiperbolikusz
tanh	tangens hiperbolikusz
acos	arkusz koszinusz
acosh	arkusz koszinusz hiperbolikusz
asin	arkusz szinusz
asinh	arkusz szinusz hiperbolikusz
atan	arkusz tangens
atan2	általánosított arkusztangens (x tengellyel bezárt szög)
atanh	arkusz tangens hiperbolikusz
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.

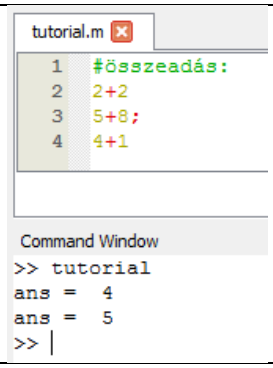
```
Command Window
>> exp(1)
ans =  2.7183
>> sin(30)
ans = -0.98803
>> sin(30*pi/180)
ans =  0.50000
>> 1.2*sin(40*pi/180+log(2.4^2))
ans =  0.76618
>> abs(-5)
ans =  5
>> round(3.8)
ans =  4
>> floor(3.8)
ans =  3
>> fix(3.8)
ans =  3
>> rem(5,6)
ans =  5
>> rem(15,4)
ans =  3
```

Néhány hasznos parancs:

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

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 alapból 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

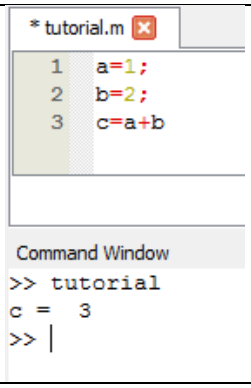


4. VÁLTOZÓK

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*.

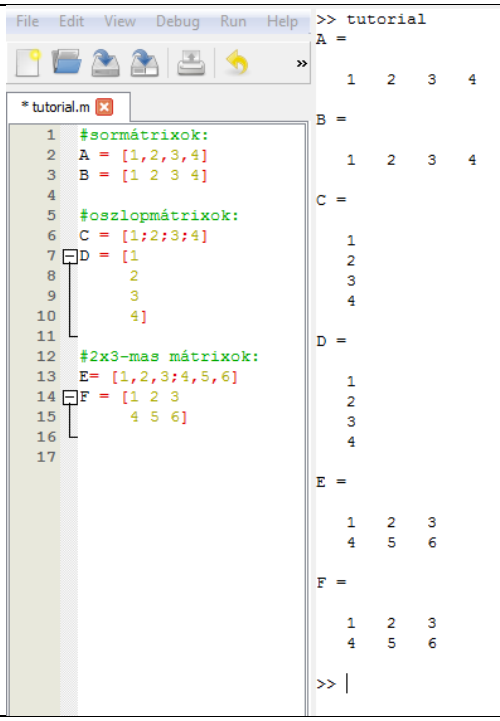


5. MÁTRIXOK

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.

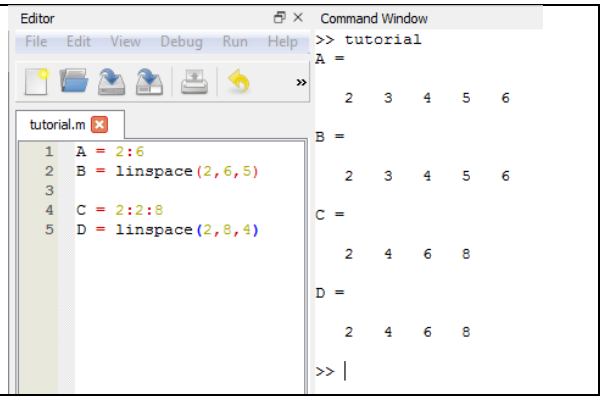


b) megadás tartománnyal

A megadás módja:
`kezdőérték:növekmény(opcionális, ha nem adjuk meg akkor 1):utolsó érték`

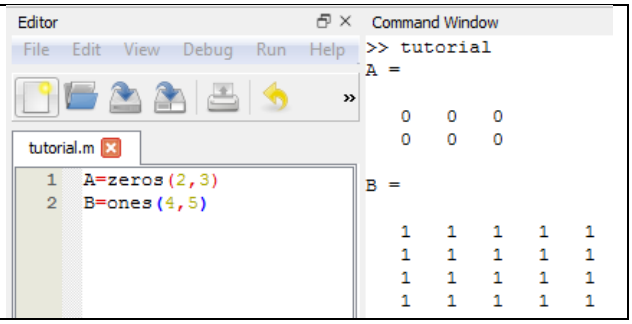
vagy

`linspace(kezdőérték, utolsó érték, elemek száma)`



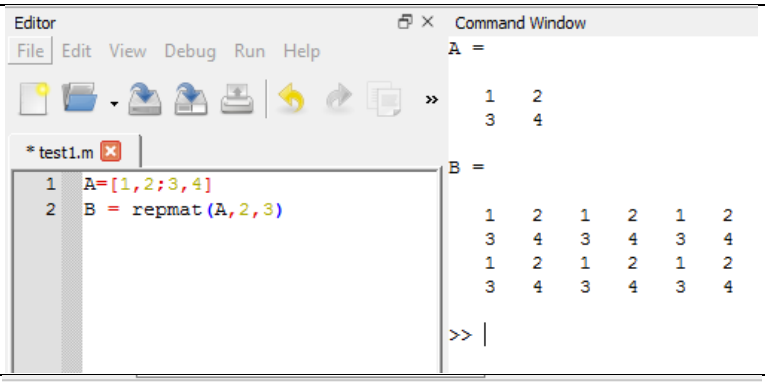
c) speciális mátrixok

nullmátrix:
`zeros(sorok száma, oszlopok száma)`
egyeseket tartlmazó mátrix:
`ones(sorok száma, oszlopok száma)`
egységmátrix:
`eye(mátrix mérete)`

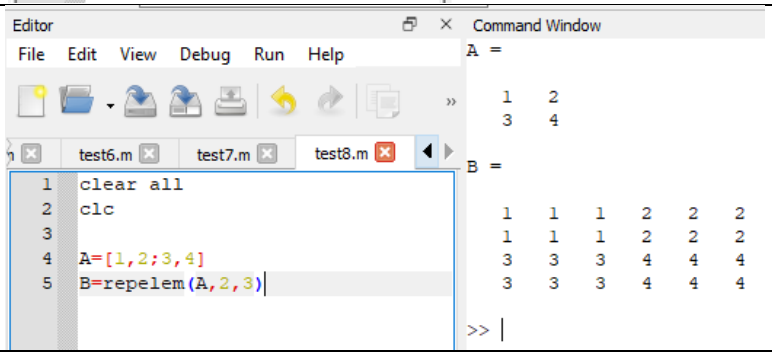


d) mátrixmegadás ismétlő parancsokkal

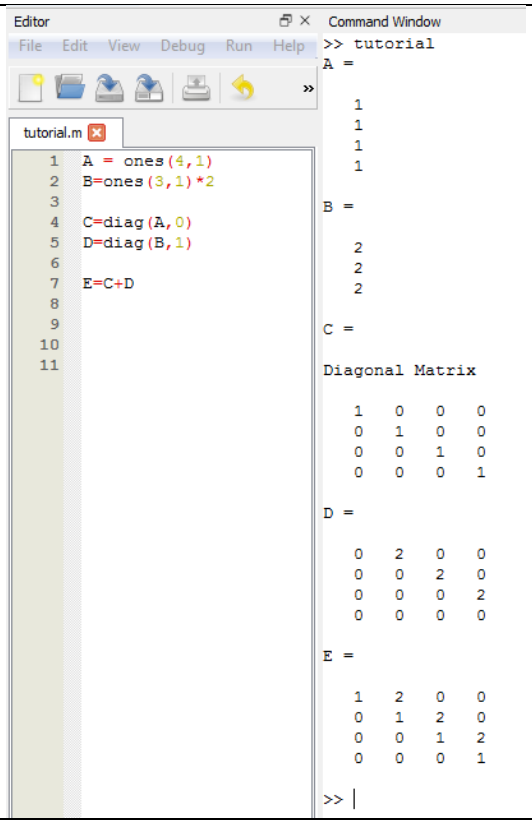
A **`B=repmat(A,m,n)`** parancs: az A mátrixot rakja m-szer egymás alá, majd a kapott blokk-oszlopot n-szer egymás mellé.



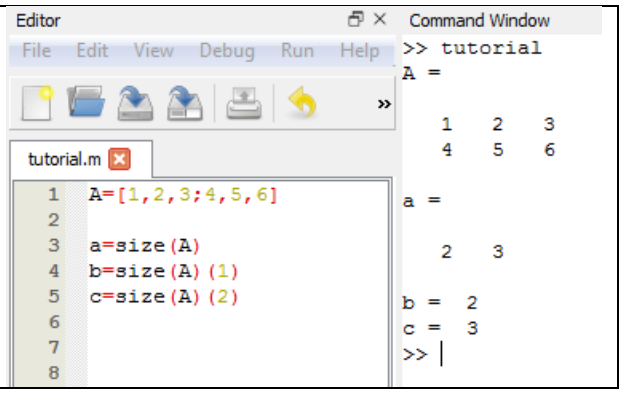
A **`B=repelem(A,m,n)`** parancs: az A mátrix elemeit rakja m-szer egymás alá, majd a kapott blokk-oszlopot n-szer egymás mellé.



e) diagonál mátrix definiálása:

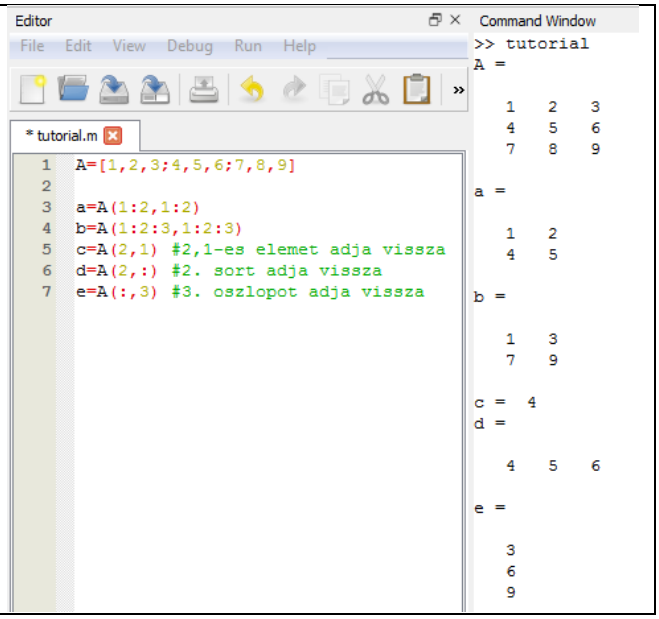
diag(diagonálelemeket tartalmazó oszlopmátrix, eltolás a főátlótól)	

5.2 Mátrix mérete

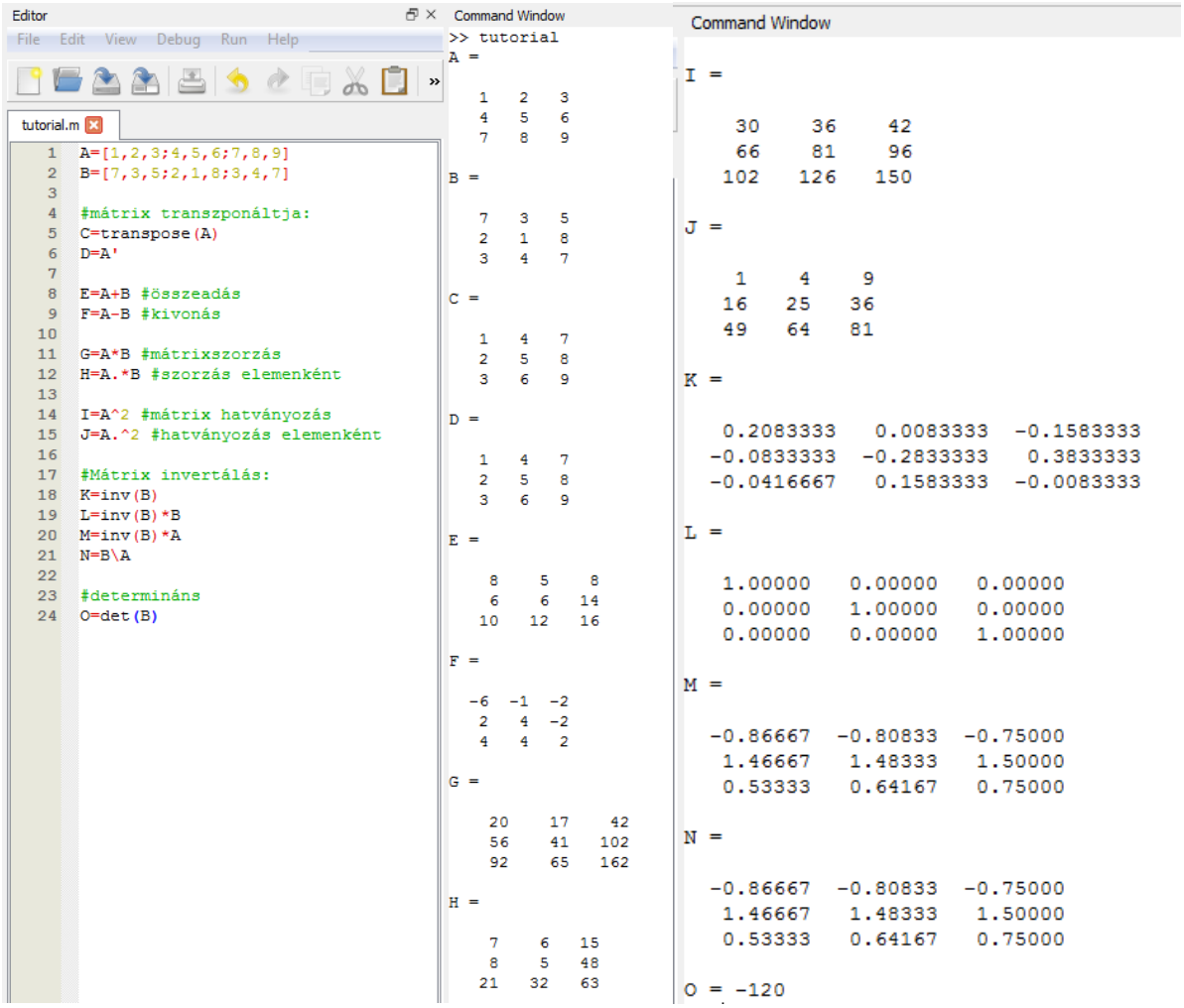
size(mátrix jele)(hányadik dimenzió) vagy **size(mátrix jele,hányadik dimenzió)**	

5.3 Részmátrixok

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



5.4 Mátrixműveletek:



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) :
utolsó érték
      utasítások
endfor
```

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

példák:

6.2 feltétel vizsgálat

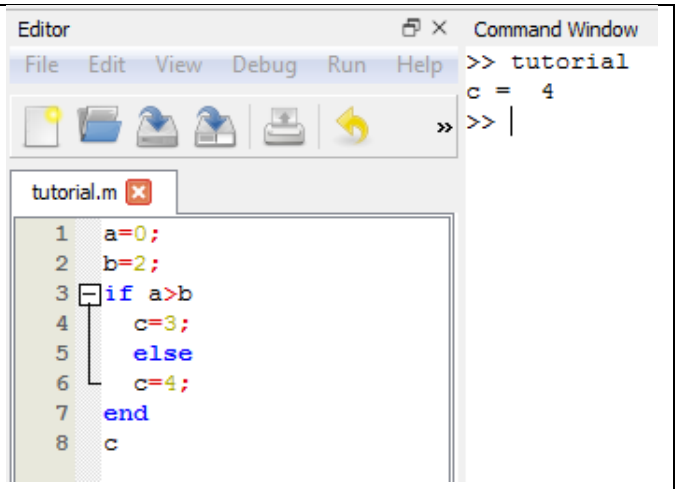
egyik lehetőség:

Általános felépítés:

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

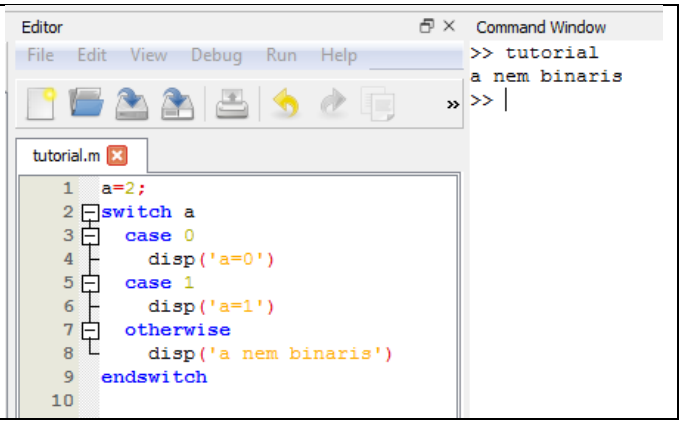
feltételek megadása:

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



másik lehetőség (ha 1 változó különböző értékekkel rendelkezhet):

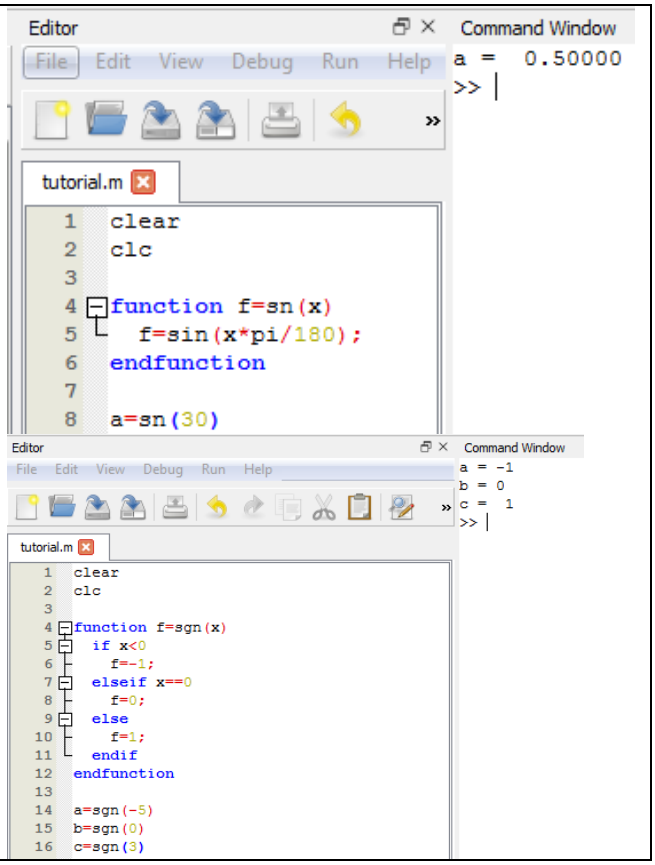
```
switch a
      case 1
            utasítások
      case 2
            utasítások
      othervise
            utasítások
endswitch
```



7. FÜGGVÉNYEK

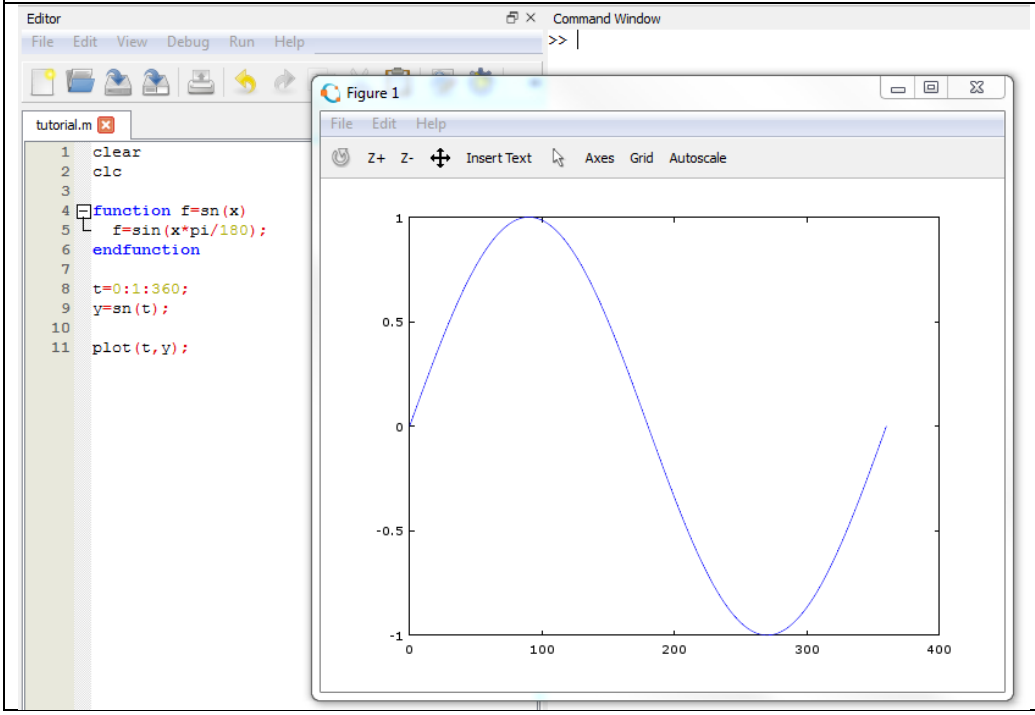
A függvénnyel egy saját Octave parancsot definiálhatunk, amit aztán a parancsablakban vagy a szkriptben bármikor felhasználhatunk.
Felépítés:

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



8. 2D FÜGGVÉNYÁBRÁZOLÁS

```
plot(vízszintes változó,függőleges változó)
```



8.1 Vonal tulajdonságok 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:

definiálás	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'	négyzet	'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')

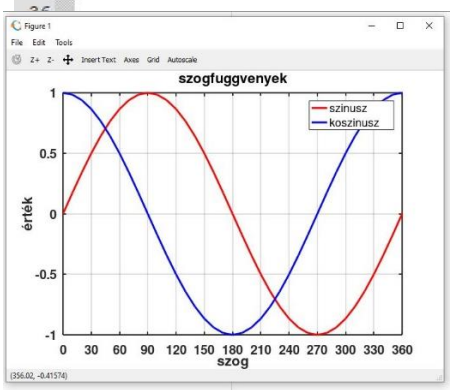
h) tengelyfeliratok

xlabel('név')
ylabel('név')

j) azonos lépték a 2 tengelyen

axis equal

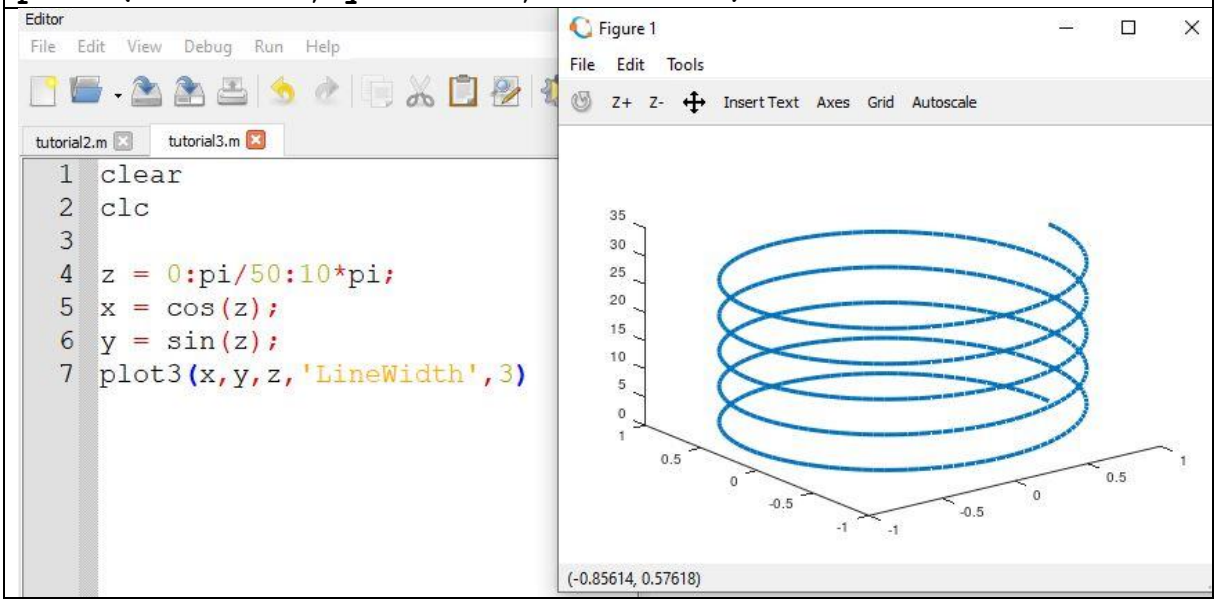
```
clear
clc

function f=sn(x)
  f=sin(x*pi/180);
endfunction

function f=cs(x)
  f=cos(x*pi/180);
endfunction

t=0:10:360;
y1=sn(t);
y2=cs(t);

plot(t,y1,'Color','red',...
          'LineWidth',3)

hold on

plot(t,y2,'Color','blue',...
          'LineWidth',3)

set(gca,'LineWidth',2,...
        'FontSize',20,...
        'FontWeight','bold',...
        'xtick',[0:30:360])
title('szogfuggvenyek')
xlim([0,360])
ylim([-1,1])
xlabel('szog')
ylabel('érték')
grid on
legend('szinusz','koszinusz')
```



9. 3D FÜGGVÉBYÁBRÁZOLÁS

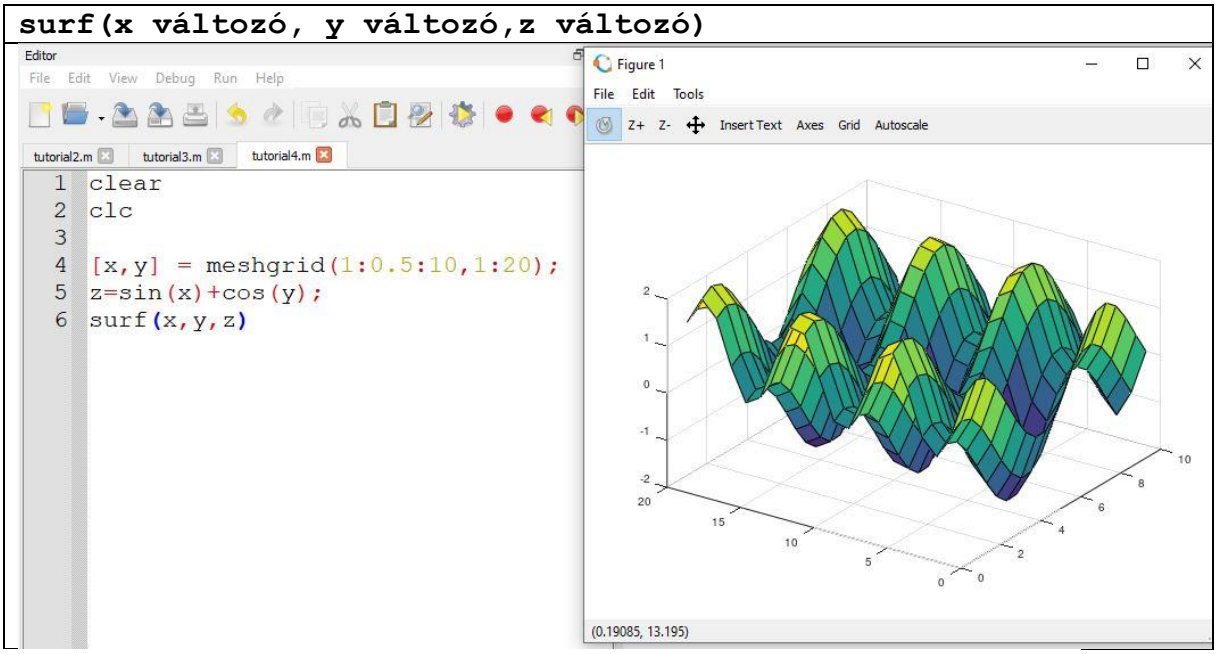
9.1 3D vonal

plot3(x változó, y változó, z változó)



9.2 3D felület

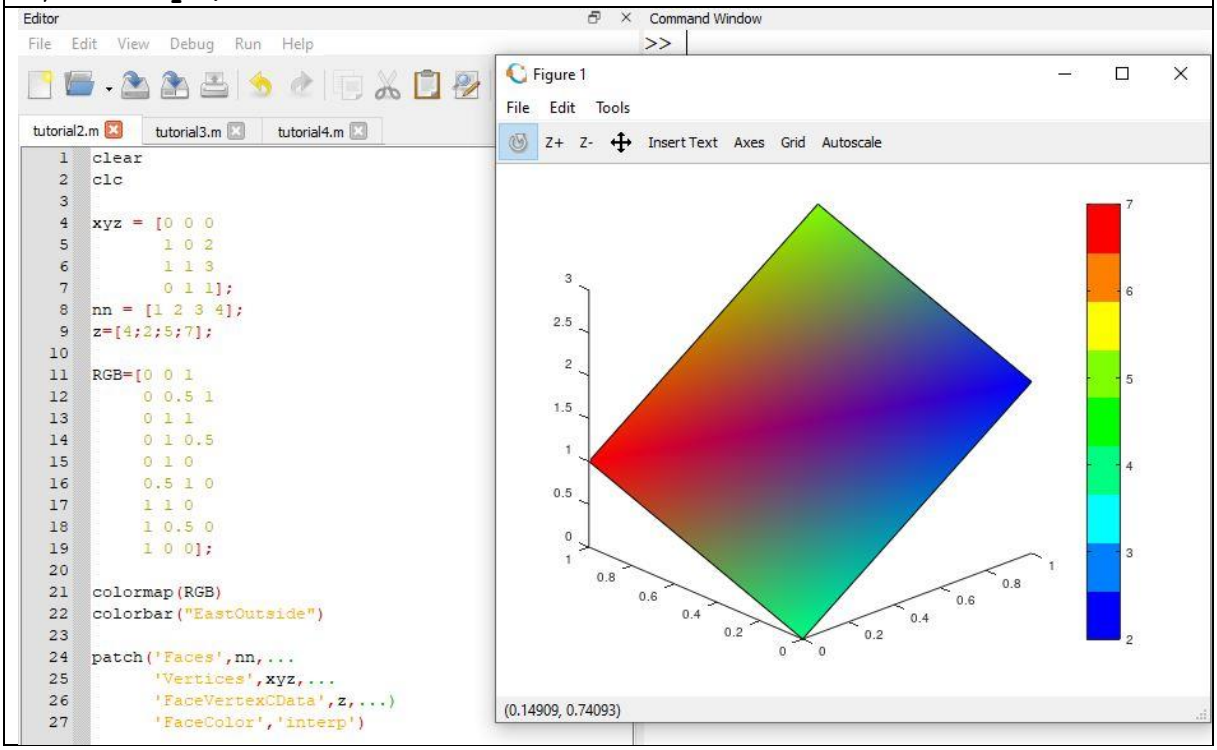
surf(x változó, y változó,z változó)



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 sorszámait tartalmazza, a z mátrix a csúcspontokhoz rendelt jellemzők mátrixa, mely a színezést határozza meg.

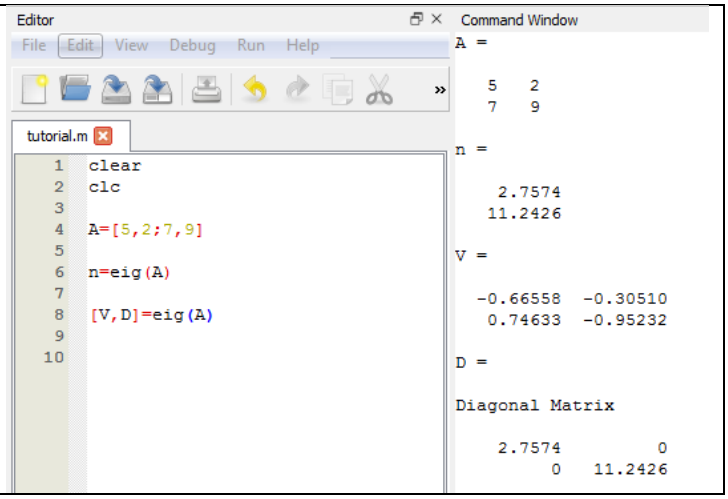
10. SAJÁTÉRTÉK, SAJÁTVEKTOR

$\underline{\underline{A}}\underline{q} = \lambda \underline{q}$, ahol:

- λ : sajátérték
- $\underline{q}$: sajátvektor

alkalmazás:

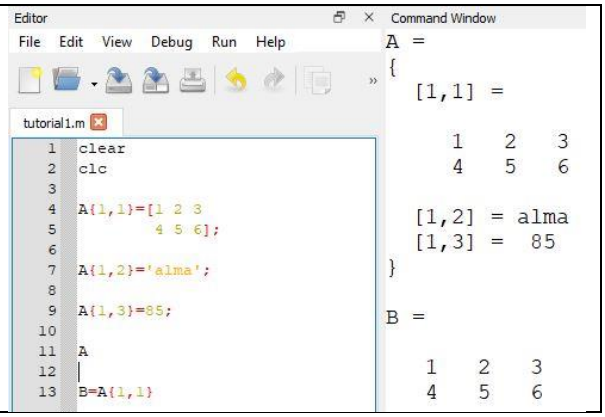
- fő tehetlenségi nyomaték számítás
- főfeszültség számítás
- modálanalízis



11. SPECIÁLIS ADATTÍPUSOK

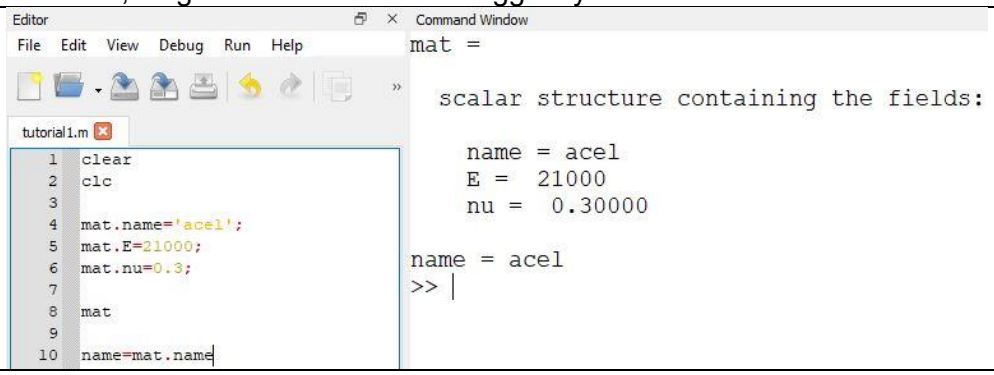
11.1 cella tömb

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.



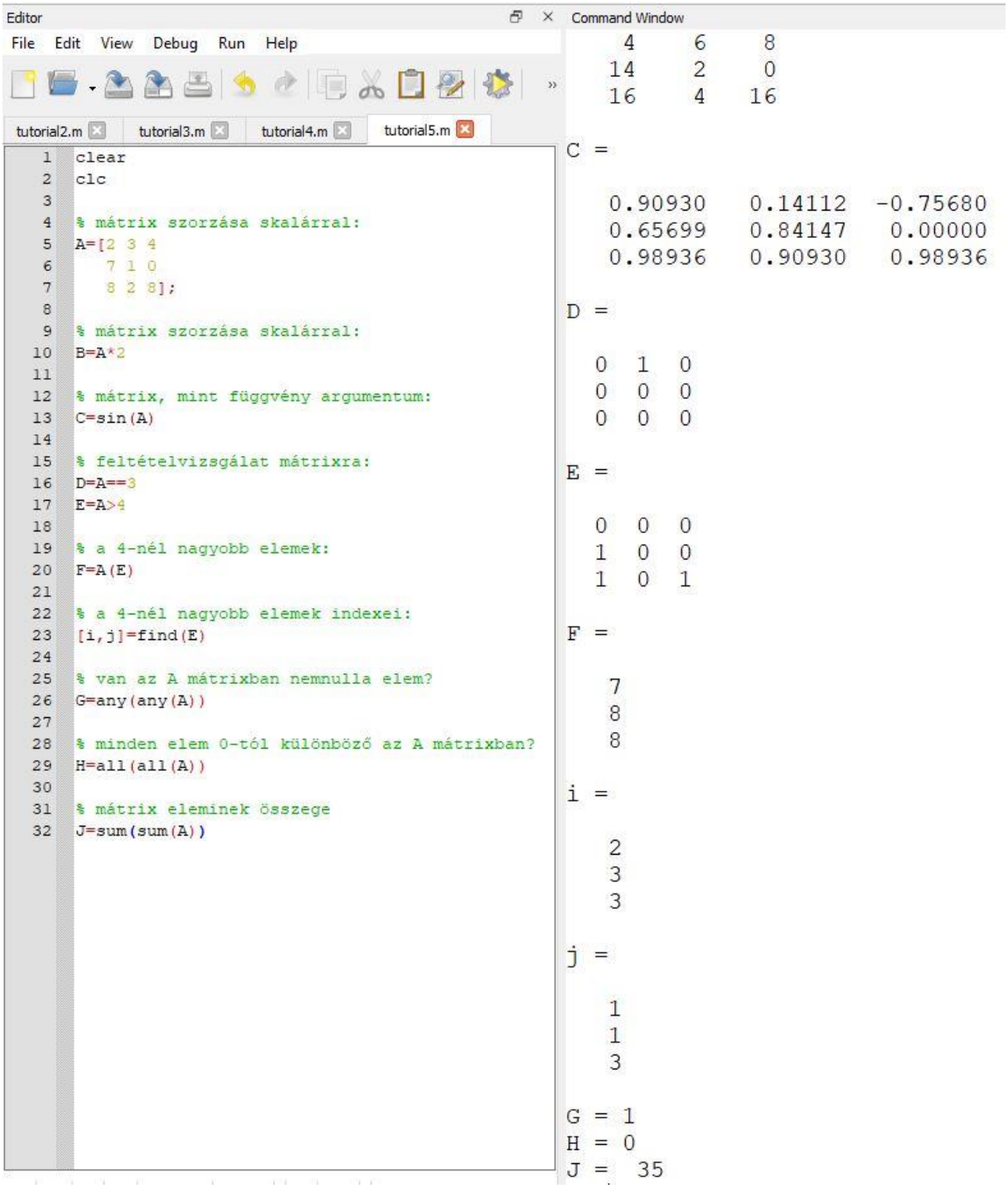
11.2 struktúra

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 föggvény argumentumába, elég a struktúrát átadni a függvénynek.



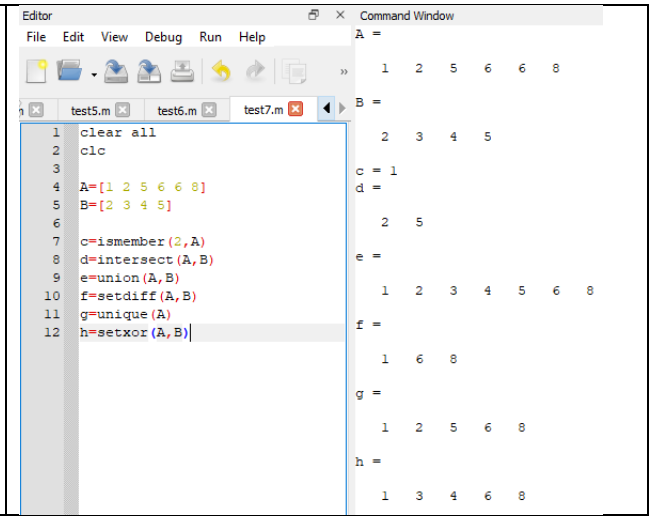
12. VEKTORIZÁLÁS

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



halmazműveletek:

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



13. EGYSZERŰ FELADATOK

13.1 Lineáris algebrai egyenletrendszer megoldás

Pl: oldjuk meg az alábbi egyenletrendszert:

$$\left.\begin{aligned} 5x_1 + 2x_2 &= 3 \\ 7x_1 + 9x_2 &= 6 \end{aligned}\right\}$$

A megoldáshoz az egyenletet mátrixos alakban kell megadnunk:

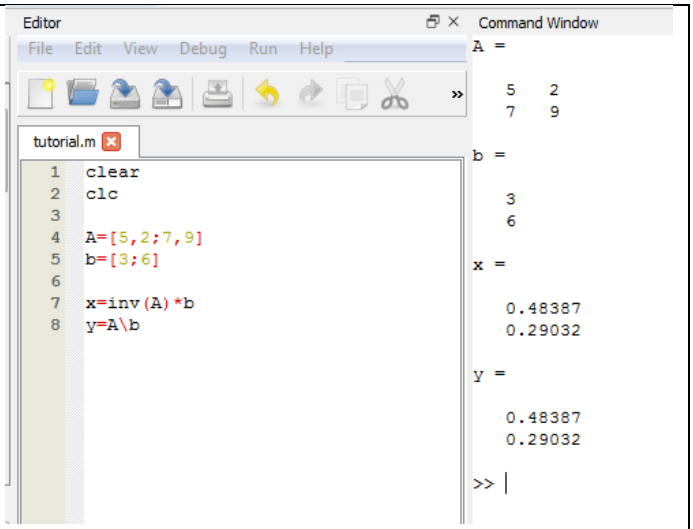
$$\underbrace{\begin{bmatrix} 5 & 2 \\ 7 & 9 \end{bmatrix}}_{\underline{\underline{A}}} \underbrace{\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}}_{\underline{x}} = \underbrace{\begin{bmatrix} 3 \\ 6 \end{bmatrix}}_{\underline{b}}$$

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

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

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

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

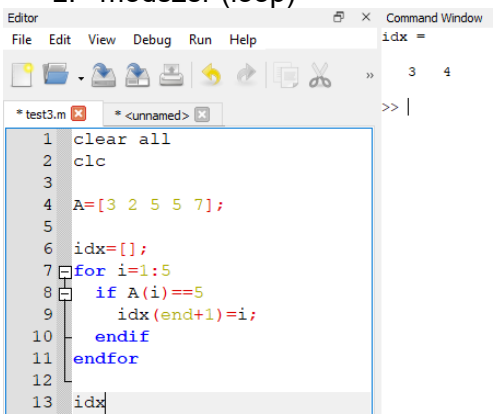


13.2 Egy adott érték indexének megkeresése egy mátrixban

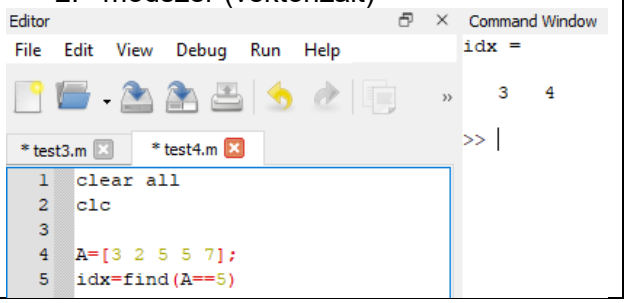
Pl: keressük, meg, hol szerepel az 5 az alábbi mátrixban:

$A = [3 \quad 2 \quad 5 \quad 5 \quad 7]$

1. módszer (loop)



2. módszer (vektorizált)



13.3 Egy adott elem hányszor fordul elő egy mátrixban?

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

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