

## 1. GYAKORLAT

---

# A MATLAB ALAPJAI

## KÖRNYEZET, SÚGÓ

A D:\ meghajtón hozzuk létre a *munka* könyvtárat, majd indítsuk el a Matlabot!

A program betöltődése után felbukkan a parancsablak a Matlab prompttal: >>

A prompt után írhatjuk be az utasításokat.

Az alábbiakban bemutatott, vastagbetűs és piros parancsokat rögtön próbáljuk ki a gépen, és jegyezzük fel a tapasztalatokat.

Ismerkedjünk meg a súgó rendszerrel (*help*, illetve *helpwin* vagy *doc* parancs)! Bármely utasításról angol nyelvű információt nyerhetünk, ha a *help* szó után beírjuk az adott utasítás nevét. Így még a *help* parancs használatáról is kaphatunk eligazítást:

```
>> help help  
>> doc
```

### Feladat

Nézzük meg a súgóban (*help* és *doc*) a *sin* parancs használatáról olvasható információt!

Próbáljuk ki a kirajzoltató utasítást.

Változtassuk az ablakok megjelenítését (ki- és bekapcsolás) menüből (Desktop), ill. a megfelelő parancsok segítségével (pl. *workspace*)!

A Matlabból kilépni a *quit* vagy az *exit* parancs beírásával lehet.

A parancsablakban beállításokat is végezhetünk:

```
>> cd D:\munka           % munkakönyvtár  
>> diary on              % naplózás be
```

Az utóbbi beállítás hatására az aktuális könyvtárban a parancsablakba bekerülő információk naplózódnak (parancsok, válaszok, hibaüzenetek, stb.).

```
>> diary off
```

A parancs kikapcsolja a naplózást, és a lezárt naplófájl elvihetjük emlékébe.

Ha órai munkánkat meg szeretnénk őrizni, akkor javasolt, hogy a *diary on* beállítást használjuk az óra végéig.

```
>> sqrt(2)
ans =
    1.4142
```

Az *sqrt* a megadott szám négyzetgyökét számítja. Az *ans* (ANSwer) a Matlab egy speciális, beépített változója. Értéke mindig a legutolsó parancs végrehajtásakor keletkezett eredmény. A numerikus (nem egész típusú, lásd lent) kiírás formátumát a

```
>> format short      % ez a default formátum
>> format long
>> format short e    % illetve format long e
```

parancsokkal szabályozhatjuk. Hasznos a *format compact* opció is!

## A MATLAB MINT SZÁMOLÓGÉP, EGYSZERŰ KIFEJEZÉSEK

A Matlab segítségével sok matematikai számítás egyszerűen elvégezhető. Használni fogjuk a nevezetes konstansokat:

```
pi = 3.14159265358979
e = 2.718281828459046    % Vigyázat, így a rendszer nem ismeri!
```

### Feladat

Melyik paranccsal tudjuk előállítani az *e* konstans értékét? Állítsuk elő az  $e^2$  értékét két különböző módon is! (Pontosan ugyanazt az eredményt adja a két előállítás?)  
Próbáljuk ki a különböző formátumokat (*format*) a *pi* és az *e* számokon!

A továbbiakban használjunk mindig a konkrét feladatnak megfelelő *format ...* formátumot.

Szintén beépített változóként használható:

```
j, i % sqrt(-1); a vessző szeparátorjel
```

Az *i* és *j* változók definíciójából látszik, hogy a Matlab komplex számokat is képes kezelni. Például:

```
>> sqrt(-9) + 3
ans =
    3.0000 + 3.0000i
```

Értékadó utasításoknál vigyázzunk, a Matlab különbséget tesz nagy- és kisbetűk között!

```
>> a = 5
a = 5
```

Ha nem szeretnénk, hogy a Matlab minden utasítás végrehajtása után beszámoljon az eredményről, akkor az utasítás után tegyünk pontosvesszőt.  
Egy változó értékét egyszerűen lekérdezhetjük nevének begépelésével (persze üssünk Entert).

### Feladat

Az előadás fóliák esetleges felhasználásával próbáljuk ki, hogy milyen változóneveket használhatunk a Matlabban! (Vizsgáljuk meg, hogy változónév kezdődhet-e számjeggyel, tartalmazhat-e magyar ékezetes betűt stb.)

Mi történik akkor, ha nem írunk változónevet egy kifejezés (vagy egy konstans) elé?

Mint már láttuk, egy sorba több parancsot is írhatunk. Adjunk ki két utasítást úgy, hogy az első végrehajtását ne jelezze ki a rendszer!

A Matlabban lehetőség van a mínusz és plusz végtelen (*-Inf*, *Inf*) használatára. Ha egy művelet a Matlab számára (számként) értelmezhetetlen, akkor az eredmény *NaN* (Not a Number). Például

```
>> 6/0
```

```
ans =  
    Inf
```

```
>> Inf/Inf
```

```
ans =  
    NaN
```

### Feladat

Mi lesz a következő műveletek eredménye:  $\text{Inf} + \text{Inf}$ ,  $3 * \text{Inf}$ ,  $-4 * \text{Inf}$ ,  $0/100$ ,  $2/0$ ,  $-2/0$ ,  $\text{Inf} - \text{Inf}$ ,  $\text{NaN} + \text{Inf}$ ,  $\text{NaN} * \text{Inf}$ ,  $\text{NaN} - \text{NaN}$ ?

A tényleges kipróbálás előtt írjuk le a füzetünkbe a várható eredményt!

Próbáljuk ki: elfogadja a rendszer az *Inf* és *NaN* „számokat” kisbetűvel is? Helyes a „NaN” alak is? (És a „Nan”?)

Tudjuk, hogy a Matlab rendszerben egyszerű kifejezések összeállíthatók és kiértékelhetők az aritmetikai műveletek és az alapvető függvények felhasználásával (megfelelő zárójelezéssel). Idézzük fel a precedenciáról az előadásokon tanultakat, szükség esetén nézzük meg a súgót.

### Feladatok

Nézzük meg a műveletek precedenciáját a Matlabban! (Súgó)

Mi lesz a következő kifejezések eredménye a Matlabban:  $-2^3$ ,  $3^2 - (5 + 4)/2 + 6*3$ ,  $\cos(0)$ ,  $\cos(\pi/4)$ ,  $2 * \text{ans}$ ,  $a^b$

Írassuk ki a Matlabbal a koszinusz-hiperbolikus függvény értékét a 0,1 helyen!

Írassuk ki a Matlabbal az arkusztangens függvény értékét a 0,5 helyen!

Találunk-e felesleges zárójeleket a következő kifejezésekben:  $2^{(3^4)}$ ,  $2^{(-3)}$

Próbáljuk ki a következő példákat is:  $\log(\exp(3))$ ,  $\sin(2 * \pi/3)$

Utóbbi esetben pontos (szimbolikus) eredményt kapunk, ha a

```
>> sin(sym('2*pi/3'))
```

parancsot adjuk ki.

### Feladat

A 2709/1024, 10583/4000 és 2024/765 törtek mind a gyök(7) közelítései. Melyik a legjobb közülük? (Milyen közelítési pontosság érhető el?)

## EGÉSZ ÉS VALÓS ADATTÍPUSOK

A Matlab alapértelmezésben double adattípust használ. Módosító utasításokkal azonban (típuskényszerítés) további egész és lebegőpontos típusok is használhatók.

Az adattípusok és korlátaik:

típus	intmin('típus')	intmax('típus')	bájt
int8	-128	127	1
uint8	0	255	1
int16	-32768	32767	2
uint16	0	65535	2
int32	-2147483648	2147483647	4
uint32	0	4294967295	4
int64	-9223372036854775808	9223372036854775807	8
uint64	0	18446744073709551615	8
	realmin('típus')	realmax('típus')	
single	1.175494e-038	3.402823e+038	4
double	2.2250738585072e-308	1.79769313486232e+308	8

A táblázatban szereplő parancsok az értékhatárok lekérdezéséhez felhasználhatók.

```
>> a = int8(6)           % int8 típusú változó létrehozása
```

A Matlab természetesen fejben tartja az összes létrehozott változót, de mi erre nem biztos, hogy képesek vagyunk. A *who* illetve a *whos* utasítások segítségével kilistáztathatjuk az összes használatban lévő változót. (A *whos* utasítás részletesebb listát ad.) Adott változó törlésére a *clear változó\_név*, az összes változó törlésére pedig a *clear* utasítás szolgál.

A parancsok dzsóker jelekkel is használhatók.

```
>> whos a*
```

A változók egyes jellemzőit (méret, típus) külön is lekérdezhetjük a *size* (ill. *length*) és a *class* parancsokkal.

Ha egy általunk megadott szám nem tárolható a választott típus értéktartományában, akkor a Matlab automatikus konverziót hajt végre.

```
>> b = int8(-12.34)
>> c = uint8(-12.34)
>> d = int16(b)
```

Műveletek csak akkor végezhetők két számértékű kifejezés között, ha vagy azonos típusúak, vagy az egyikük *double*. Az eredmény típusa a *double*-től különböző típus, vagy *double*.

```
>> b + c           % meglepő!
>> b + pi          % elvégezhető, de az eredmény int8
```

### Feladat

Matlab használata nélkül mondjuk meg, hogy elvégezhetők-e a következő műveletek (a fenti megadásokkal), ill. milyen típusú lesz a következő kifejezések eredménye:

$a + b$ ,  $b + c$ ,  $b + d$ ,  $c + d$ ,  $c + \pi$ ,  $d + \pi$ ,  $c + \exp(1)$

Ellenőrizzük a megadott válaszokat a Matlabbal!

Hajtsuk végre a feladatot úgy is, hogy az eredmény értékét is megmondjuk előre, majd ellenőrizzük a Matlabbal!

Típuskényszerítéssel érjük el, hogy néhány fenti *el nem végezhető művelet* most már elvégezhető legyen.

A Matlabban a túlcsordulás kezelése eltér a megszokottól.

```
>> f = uint8(255)
>> f = intmax('uint8') % ekvivalens művelet
>> f + 1               % !!!

>> g = intmin('uint16') % ugyanígy
>> g - 1
```

### Feladat

Matlab használata nélkül mondjuk meg, hogy a következő műveleteknél történik-e túlcsordulás (ill. mennyi lesz a műveletek eredménye):

```
x1 = int8(100), x1 + x1, -x1 - x1
x2 = uint8(150), x2 + x2, -x2 - x2
x3 = int16(17000), x3+x3, -x3 - x3
x4 = int8(64), x4 + x4, -x4 - x4
```

Ellenőrizzük a megadott válaszokat a Matlabbal!

**Feladat**

Írassuk ki a megfelelő parancs segítségével

- a legnagyobb ábrázolható 8 bites előjeles egész
- a legnagyobb ábrázolható 16 bites előjel nélküli egész
- a legkisebb ábrázolható 16 bites előjeles egész
- a legnagyobb/legkisebb ábrázolható double típusú számot

(Hogyan tudjuk ellenőrizni, hogy az utolsó két válaszunk valóban megfelelő?)

A fentiekhez hasonló módon történik más műveletek végrehajtása is. Az eredmény itt is eltérhet attól, amit várnánk.

```
>> h = uint8(2)/3
>> h = uint8(2)/uint8(3)      % ekvivalens megadás
>> k = uint8(3)/2
>> k = uint8(3)/uint8(2)      % ekvivalens megadás
```

**Feladat**

Matlab használata nélkül mondjuk meg, hogy mennyi lesz a műveletek eredménye:

$k = \text{uint8}(5)/\text{uint8}(3)$ ,  $k = \text{uint8}(5)/\text{uint8}(4)$

Ellenőrizzük a megadott válaszokat a Matlabbal!

## PONTOSSÁG, HASONLÍTÁSOK

Numerikus rendszerekben fontos kérdés az, hogy milyen pontosságot várhatunk el egy utasítássorozat (pl. script) végrehajtása során.

A következő példában egy *single pi* változót hozunk létre, és összehasonlítjuk a *pi* beépített (double) konstanssal.

```
>> s_pi = single(pi)
>> s_pi == pi      % sima = az értékadás
>> kulonbs = pi - s_pi      % eltérés
>> pi > s_pi      % kerekítés miatt a single a nagyobb
>> class(kulonbs)
```

**Feladatok**

Melyik a nagyobb,  $e^{14}$  vagy  $382801\pi$ ? Írassuk ki a választ a Matlab segítségével!

Ellenőrizzük az előadáson bemutatott példát a gép epsilon-ról!

(Eml.: `eps`, `1 + eps`, `1 + eps/2`, `1 + eps/2 == 1`, `eps/2 == 0` parancsok.)

Értelmezzük a lépéseket!

## ÖNÁLLÓ GYAKORLÁS

Futtassuk le az 1. előadás parancsait darabonként, a `parancsok_1.m` felhasználásával! (Ízelítő, adattípusok, hasonlítások.)

Írassunk ki minél több különböző Matlab hibaüzenetet! („Megfelelően hibás” parancsokkal.)  
Pl.:

```
??? ... Error: The input character is not valid in MATLAB statements or  
expressions.  
??? Undefined function or variable '...'.  
??? ... Error: Expression or statement is incomplete or incorrect.  
??? Undefined function or method '...' for input arguments of type 'double'.
```

## OTTHONI MUNKA

1. A demo parancs segítségével indíthatjuk el a Matlab beépített oktató, illetve demonstrációs célokat szolgáló programjait. Ismerkedjünk meg ezekkel, ill. közülük néhányval!

2. Gépeljük be egymás után az alábbi utasításokat. Mit ír ki a Matlab?

```
>> u=[-4,6,8,10]  
>> v=[-2,3,4,5]  
>> min(u)  
>> u+v  
>> diff=u-v, max(diff)  
>> u./v    % osztás elemenként  
>> a=[1 2 3; 4 5 6; 7 8 9]  
>> b=max(a)  
>> c=a'  
>> sin(b), sin(a)  
>> diag(a)  
>> eig(a)  
>> inv(c)
```

3. Ismerkedjünk meg az *isinf*, az *isfinite* és az *isnan* függvények használatával! (Súgó.)

4. Az általunk megismert *eps* valójában az *eps(1)* érték. Hogyan tudjuk értelmezni az *eps(2)* számot?

\*5. Állítsuk elő az 1/10 mellett néhány további szám normalizált lebegőpontos alakját is (lásd előadás). Végezzünk a számokkal egyszerű műveleteket, és ellenőrizzük az eredményeket.

