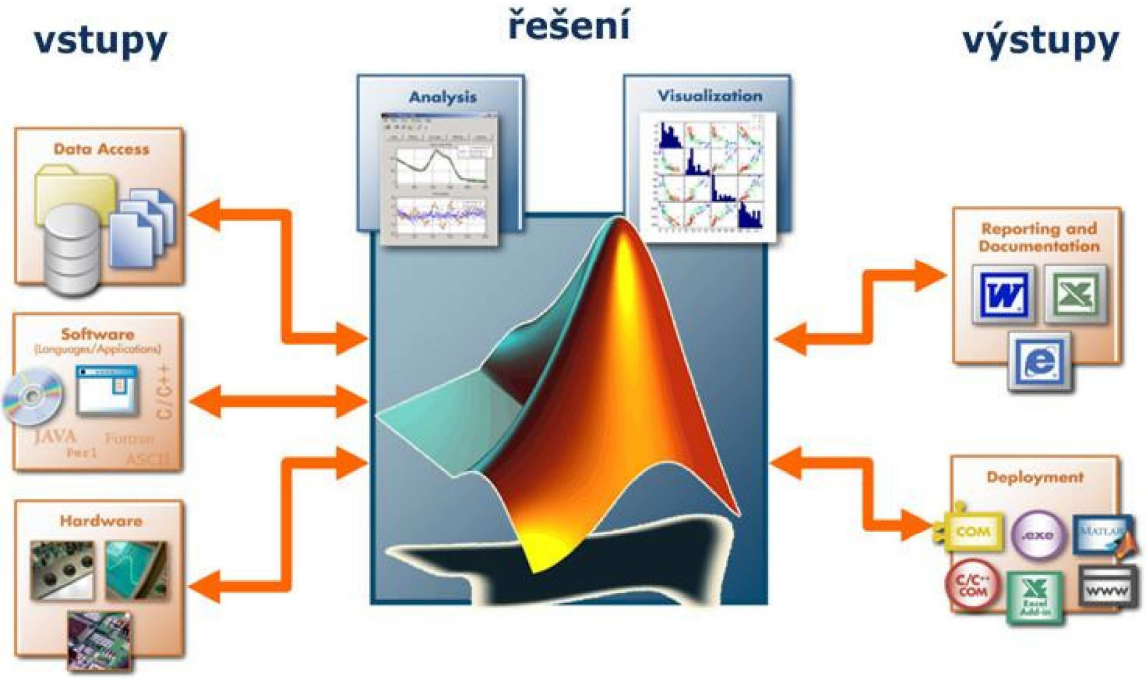
1 Modelovanie, simulácia a programové prostredie MATLAB/Simulink

- **Simulačný jazyk MATLAB**

⇨ **MATLAB** (MATrix LABoratory) je simulačný jazyk vyvinutý pre vedecko-technické výpočty, modelovanie, návrhy algoritmov, simuláciu, analýzu a prezentáciu dát, meranie a spracovanie signálov, návrhy riadiacich a komunikačných systémov.
⇨ Základ tvorí výpočtové jadro, ktoré je zamerané na operácie s maticami a preto je považované za najsilnejšiu stránku simulačného jazyka MATLAB s jeho optimálnymi algoritmami.
⇨ Jadro je rozšírené o množstvo nadstavieb (Toolboxov = aplikačné knižnice), ktoré sú určené na riešenie úloh z takmer všetkých oblastí ľudskej činnosti.



Obrázok 1-1 Výpočtový systém MATLAB/Simulink

⇨ Základné operácie simulačného jazyka MATLAB sú operácie s maticami
⇨ Umožňuje 2D a 3D vizualizáciu v grafoch s množstvom voliteľných a nastaviteľných parametrov
⇨ Umožňuje vytváranie vlastných funkcií a knižníc
⇨ Distribúcia užívateľských aplikácií napríklad do jazyka C
⇨ Obsahuje rozsiahlu dokumentáciu v PDF alebo help v on-line hypertextovej forme

- **Vývoj simulačného jazyka MATLAB**

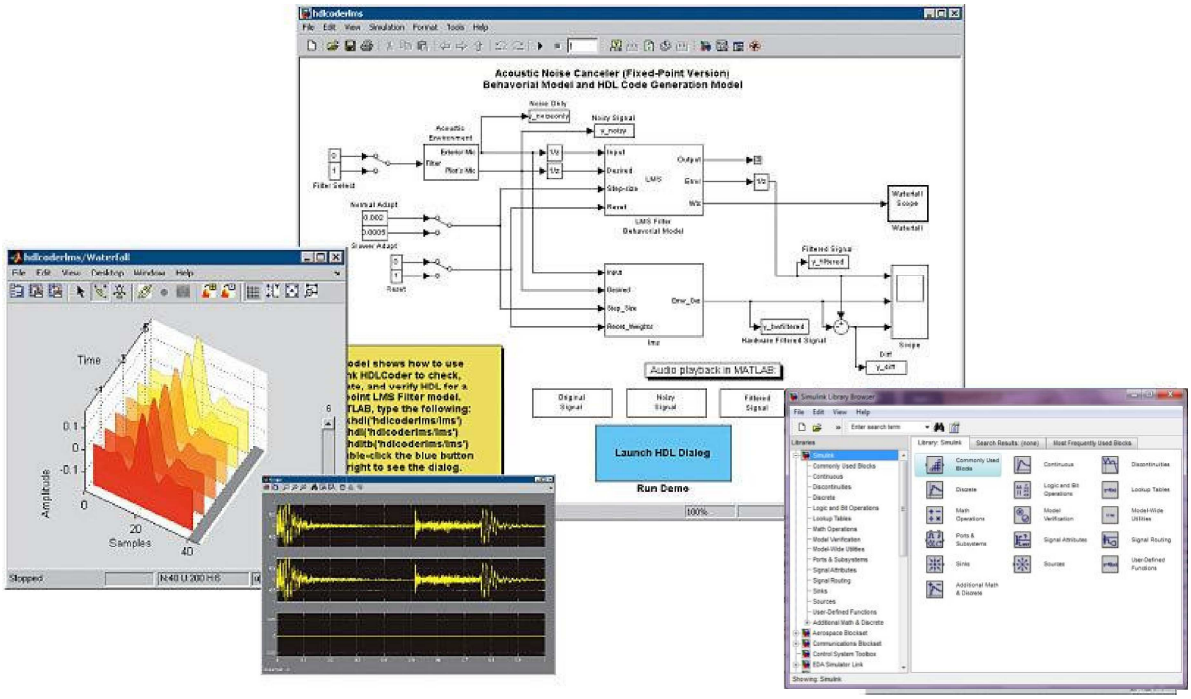
Korene MATLABu siahajú do 70-tych rokoch 20. storočia, kedy boli v jazyku Fortran naprogramované knižnice LINPACK (zastrešovala oblasť lineárnych rovníc) a EISPACK (vyvinutá pre riešenie vlastných čísel). V osemdesiatych rokoch sa prof. Cleve Moler rozhodol pre svojich študentov z univerzity v Novom Mexiku vytvoriť program, ktorý by zastrešoval subrutiny LINPACK a EISPACK s interaktívnym rozhraním bez nutnosti ovládať Fortran. Tento program nazval MATLAB (MATrix LABoratory), neskôr ho s Stevem Bangertom a Johnom Littleom preprogramovali do jazyka C a doplnili o grafiku a založili spoločnosť The MathWorks, ktorá MATLAB ďalej vyvíja a inovuje.

- **Typické úlohy riešené simulačným jazykom MATLAB**

⇨ matematické výpočty, maticové operácie, lineárna algebra, numerické výpočty
⇨ tvorba vlastných funkcií a skriptov
⇨ riešenie lineárnych a nelineárnych diferenciálnych rovníc
⇨ riešenie úloh z analýzy a syntézy lineárnych a nelineárnych DS
⇨ simulácia modelov fyzikálnych systémov

- **Programové prostredie Simulink**

Simulink je grafická nadstavba, zameraná na modelovanie a simuláciu dynamických systémov, ktorý využíva algoritmy jazyka Matlab, ako napríklad algoritmy pre numerické riešenie nelineárnych diferenciálnych rovníc.



Obrázok 1-2 Typy okien v programovom prostredí MATLAB/Simulink

⇨ nezávislé používateľské rozhranie
⇨ pomocou aplikácie Simulink a jeho grafického editora je možné vytvárať modely lineárnych, nelineárnych, v čase diskrétnych alebo spojitých systémov jednoduchým posúvaním funkčných blokov pomocou myši
⇨ obsahuje bloky, ktoré reprezentujú jednotlivé elementárne prvky systémov. Zo skupiny blokov je možné jednoducho vytvárať subsystémy
⇨ využitie napríklad v aplikovanej matematike, automatickom riadení a regulácii, modelovaní fyzikálnych systémov, spracovanie obrazu, zjednodušenie zložitých blokových schém a iné...

- **Aplikačné knižnice (Toolboxy)**

⇨ toolboxy sú tematický zamerané knižnice príkazov určitej vednej oblasti
⇨ časť je distribuovaná voľne prostredníctvom internetu, iné sú vyvíjané spoločnosťou The MathWorks sú predávané prostredníctvom regionálnych predajcov.
⇨ sú vzájomne previazané a používanie jedného neobmedzuje použitie iného.
⇨ všetky toolboxy pritom používajú základné výpočtové jadro simulačného jazyku MATLAB.

- **Control System Toolbox**

Control System Toolbox je aplikačná knižnica, ktorá rozširuje programové prostredie MATLAB o nástroje pre teóriu systémov a kybernetiku (riadiacu techniku).

⇨ obsahuje funkcie z oblasti analýzy a návrhu riadiacich systémov
⇨ okrem klasických prechodových charakteristík využíva aj popis systémov v stavovom priestore
⇨ obsahuje časovo invariantné objekty (LTI), čo sú štruktúry popisujúce jednorozmerné i viacrozmerné lineárne systémy
⇨ poskytuje nástroj na analýzu odozvy systému, ako napríklad prechodovú a frekvenčnú charakteristiku v logaritmických súradniciach a komplexnej rovine.

- **Symbolic Math Toolbox**

Symbolický Toolbox ako už aj jeho názov napovedá poskytuje nástroje pre riešenie a prácu so symbolickými výrazmi.

⇨ do simulačného jazyka Matlab prináša stovky symbolických funkcií, ako napríklad derivovanie, integrovanie, transformácia a riešenie algebraických a diferencialnych rovníc, zjednodušovanie výrazov ...
⇨ oblasť jeho využitia je v aplikovanej matematike a v finančnom modelovaní a analýze

- **Nápoveda v programovom prostredí MATLAB**

Súčasťou programového prostredia MATLAB je nápoveda, ktorá obsahuje kompletnú dokumentáciu k programovému systému aj s príkladmi pre konkrétne funkcie.

- **Nápoveda v okne Command Window**

Ľahko ju vyvoláme príkazom ***help*** za ktorý zapíšeme príkaz alebo okruh príkazov. O takejto nápovede sa ľahko dozvieme ak zadáme do príkazového okna ***help help***

⇨ Ak potrebujeme zistiť použitie a zápis nejakej funkcie (napríklad eye) použijeme príkaz ***help eye***

```
>> help eye
 EYE Identity matrix.
   EYE(N) is the N-by-N identity matrix.

   EYE(M,N) or EYE([M,N]) is an M-by-N matrix with 1's on
   the diagonal and zeros elsewhere.

   EYE(SIZE(A)) is the same size as A.

   EYE with no arguments is the scalar 1.

   EYE(M,N,CLASSNAME) or EYE([M,N],CLASSNAME) is an M
   of class CLASSNAME on the diagonal and zeros elsewhere
```

⇨ Ak potrebujeme zistiť príkazy z niektorého okruhu (napr. elementárne matematické funkcie) do príkazového okna zadáme ***help elfun***

```
>> help elfun
  Elementary math functions.

Trigonometric.
  sin       - Sine.
  sind      - Sine of argument in degrees.
  sinh      - Hyperbolic sine.
  asin      - Inverse sine.
  asind     - Inverse sine, result in degrees.
  asinh     - Inverse hyperbolic sine.
  cos       - Cosine.
  cosd      - Cosine of argument in degrees.
  cosh      - Hyperbolic cosine
```

⇨ Ak potrebujeme nájsť niektorú funkciu ale nevieme jej presný názov, môžeme použiť funkciu ***lookfor***, ktorá prehľadáva celú dokumentáciu a hľadá v nej konkrétne slovo alebo reťazec slov.

- **MATLAB Help**

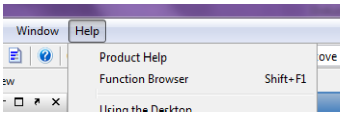
MATLAB Help je ďalšou z možností nápovedy v programovom prostredí MATLAB. Vyvoláme ju

⇨ stlačením tlačidla F1

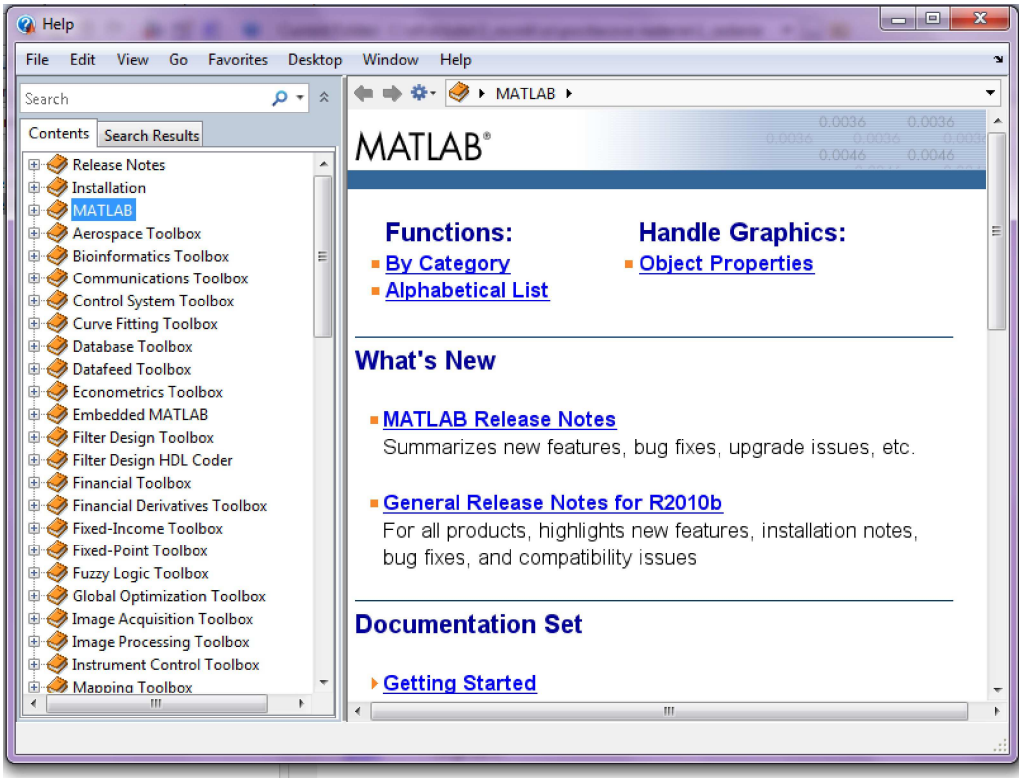
⇨ klepnutím na ikonu 

⇨ výberom možnosti z panelu nástrojov **Help -> Product Help**

⇨ alebo zadaním príkazu ***helpbrowser*** v príkazovom okne



Tieto možnosti otvoria nasledujúce okno:



Obrázok 1-3 Okno pre nápovedu v simulačnom jazyku MATLAB

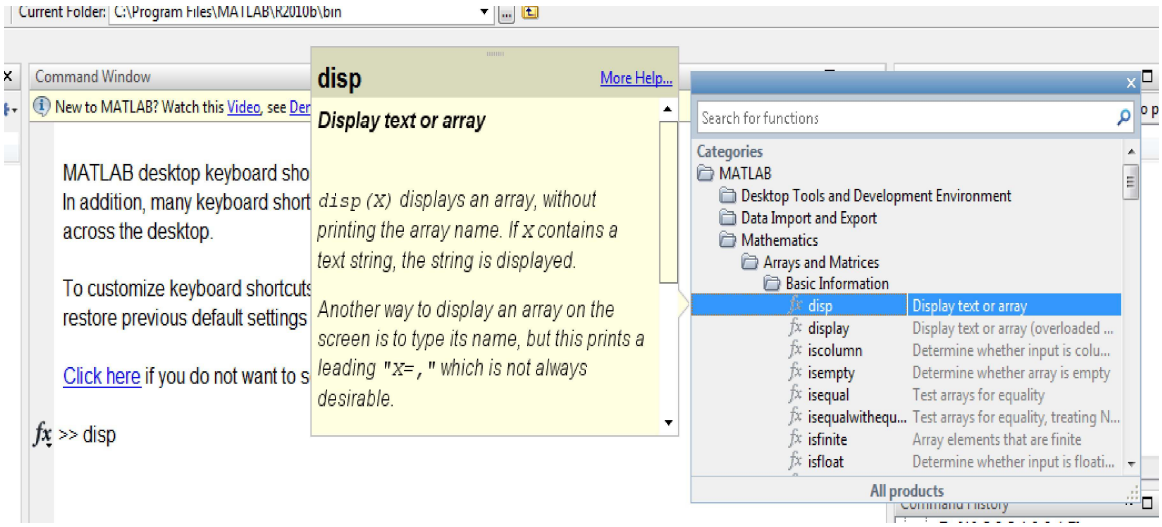
V tomto ***help-e*** sú všetky témy spracované ako www stránky. Súčasťou sú aj ukážkové programy (Demos), ktoré názorne ukazujú použitie jednotlivých funkcií na príkladoch.

- ***Okno nápovedy*** sa skladá z dvoch panelov. Ľavý panel umožňuje vyhľadávať v štruktúre nápovedy, v pravom paneli sa následne objaví aktuálna téma.
- ***Contents*** obsahuje knižnice k jednotlivým Toolboxom, ktorými je možné postupným prechádzaním po zložkách dostať sa až k jednotlivým funkciám.
- V záložke ***Search Results*** je možné pomocou kľúčového slova alebo funkcie možné vyhľadávanie.

- **Prehliadač funkcií**

Ďalšou alternatívou je prehliadač funkcií spúšťaný kombináciou kláves **Shift + F1** alebo výberom z panelu nástrojov z položky ***Help -> Function Browser***.

Prehliadač funkcií umožňuje prezerať jednotlivé záložky – toolboxy a ich podsekcie - v ktorých sú podľa účelu porozdeľované funkcie.
Pri podržaní kurzora nad danou funkciou je otvorená informačná bublina, ktorá nás informuje o použití danej funkcie.
Ak na danú funkciu dvakrát klikneme, dostane sa do *Command Window*, kde s ňou môžeme následne pracovať.



Obrázok 1-4 Okno pre prehliadač funkcií

- **The MathWorks**

The MathWorks je oficiálna stránka programového prostredia MATLAB, kde je taktiež možné nájsť mnoho zaujímavých noviniek o produkte MATLAB, funkcií a tutoriálov k nim, nové toolboxy a mnoho ďalších. http://www.mathworks.com

1.1 Porovnanie simulačného jazyka MATLAB a procedurálneho jazyka C

S programovacím jazykom C sa študenti oboznámili v 1. ročníku na odbore Kybernetika v predmete Programovanie. Na úvod tejto časti si porovnáme programovací jazyk C a programové prostredie Matlab, s ktorým budeme pracovať na predmetoch študijného programu Kybernetika ako napríklad Simulačné systémy , Základy automatického riadenia, Riadenie a vizualizačné systémy, Počitačové riadenie.

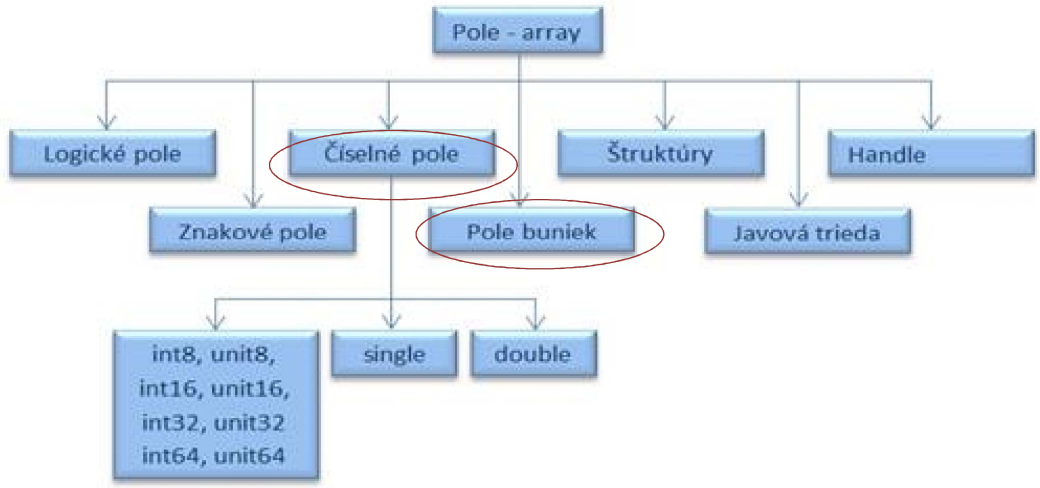
MATLAB (MATrix LABoratory) je vysokoúrovňový simulačný jazyk a interaktívne prostredie pre numerické výpočty, vizualizáciu a programovanie. Pomocou simulačného jazyka Matlab, môžete analyzovať dáta, vyvinúť algoritmy, a vytvárať modely aplikácií. Už samotný názov nám prezrádza, že bol navrhnutý predovšetkým pre prácu s maticami (matrix) a vektormi. Simulačný jazyk ďalej umožňuje prácu s rôznymi grafickými nástrojmi.
Otvorená architektúra prostredia MATLAB viedla k vzniku knižníc funkcií nazvaných toolboxy. Toolboxy predstavujú jednou z výhod jazyka MATLAB, rozširujú použitie programu v príslušných vedných disciplínach.

Jazyk C je problémovo orientovaný programovací jazyk, ktorý bol pôvodne vyvinutý pre operačný systém UNIX a až neskôr bol implementovaný aj do iných operačných systémov a stal sa jedným z najpoužívanejších programovacích jazykov. Jeho hlavnou výhodou je jednoduchosť a nezávislosť na počítači. Umožňuje vytvárať rozsiahle a výkonné programy.

	Matlab	**C**
Druh jazyka	Skriptovací / simulačný	Problémovo orientovaný
Deklarácia dát	Nie je potrebná	Potrebná
Reprezentácia dát	Matica, vektor	Podľa deklarácie
Indexovanie od	1	0

1.2 Dátové typy jazyka MATLAB

Dátový typ je určenie množiny hodnôt, ktoré môže daná premenná nadobúdať. Matlab obsahuje 15 základných dátových typov a každý z nich môže byť zapísaný vo forme matice alebo poľa. Všetky základné typy sú znázornené v nasledujúcom obrázku.



Obrázok 1-5 Dátové typy simulačného jazyka MATLAB

- **Číselné premenné**

Číselné premenné v MATLABe môžu byť zapísané ako znamienkové (signed), alebo neznamienkové (unsigned), ako celé čísla (integer) alebo ako reálne čísla a to buď s jednoduchou (single) alebo dvojitou (double) presnosťou.

Pri celých číslach MATLAB podporuje 8, 16, 32, 64 - bitový spôsob znamienkového/ neznamienkového zobrazenia. Rozsahy jednotlivých typov celých čísel si môžete pozrieť v nasledujúcej tabuľke.

Označenie	**popis**	**rozsah**
int8	znamienkový 8bitový integer	-2^7 – 2^7 -1
int16	znamienkový 16bitový integer	-2^{15} – 2^{15}-1
int32	znamienkový 32bitový integer	-2^{31} – 2^{31}-1
int64	znamienkový 64bitový integer	-2^{63} – 2^{63}-1
uint8	neznamienkový 8bitový integer	0 – 2^8 -1
uint16	neznamienkový 16bitový integer	0 – 2^{16}-1
uint32	neznamienkový 32bitový integer	0 – 2^{32}-1
uint64	neznamienkový 64bitový integer	0 – 2^{64}-1

Pri nezadaní konkrétneho typu číselnej premennej MATLAB ukladá všetky čísla ako reálne čísla s dvojitou presnosťou. Pokiaľ chceme reálne číslo uložiť len s jednoduchou presnosťou musíme to

spraviť pri jeho inicializácií. Na výpise kódu z programovacieho prostredia MATLAB je znázornený rôzny číselný zápis.

Pri použití funkcie „***whos***“ môžeme získať informácie o type premennej a jej veľkosti v bytoch.

```
>> a=uint8(136);
>> b=int32(2165);
>> c=5.214;
>> d=single(6.1247);
>> whos a b c d
  Name      Size            Bytes  Class     Attributes

  a         1x1                 1  uint8
  b         1x1                 4  int32
  c         1x1                 8  double
  d         1x1                 4  single
```

- **Formátovaný vstup a výstup**

Matlab ponúka dve možnosti pre vstup a to :

funkcia ***sscanf*** - čítanie dát z reťazca, po preformátovaní ho uloží do premennej
funckia ***fcanf*** - číta a formátuje dáta z textového súboru.

Pre formátovaný výstup sa používajú funkcie :

funkcia ***sprintf*** - vzťahuje sa na všetky prvky poľa, naformátuje ich a vráti výsledok
funkcia ***fprintf*** - zapisuje dáta do textového súboru alebo ich vypíše na obrazovku.

Pri používaní funkcií formátového vstupu a výstupu je potrebné použiť konverzné znaky. V nasledujúcej tabuľke sú uvedené najpoužívanejšie.

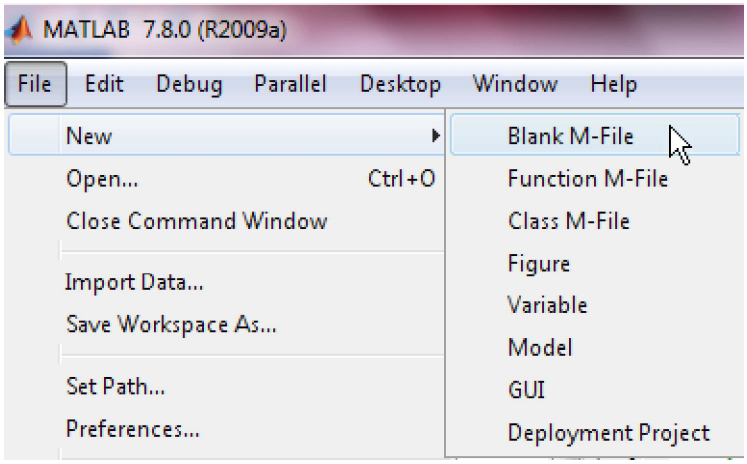
Celé číslo	signed	%d	Desiatkové číslo typu signed
		%ld	Desiatkové číslo typu signed
	unsigned	%u	Desiatkové číslo typu unsigned
		%lu	Desiatkové číslo typu unsigned long
		%o	Osmičkové číslo
		%x	Hexadecimetrálne číslo s malými písmenami
		%X	Hexadecimetrálne číslo s veľkými písmenami
Reálne číslo		%f	
Znaky		%c	Jeden znak
		%s	Reťazec

Príklad:

```
premenna=sscanf(x,'%o');
fprintf('Výsledok je: %f',premenna);
```

1.3 Riadiace štruktúry jazyka MATLAB

Pri tvorbe zložitejších programov, v ktorých sa využívajú riadiace štruktúry, je nevyhnutné vytvárať skripty a kód písať v nich. Nový skript vytvoríme nasledovne:

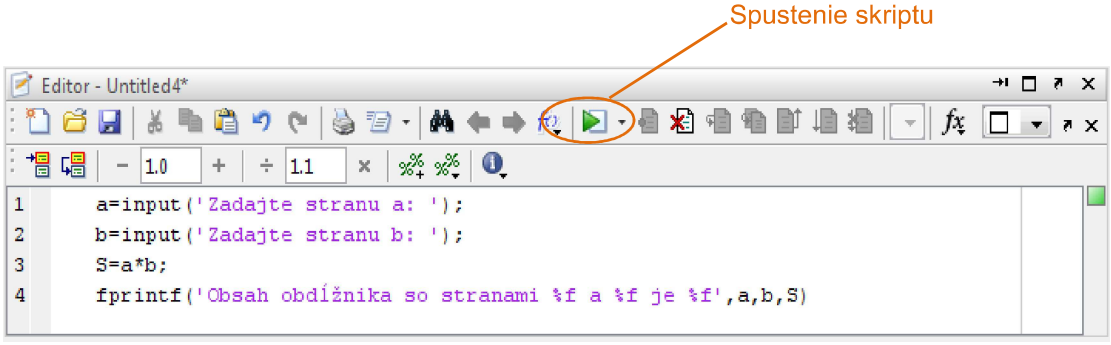


Obrázok 1-6 Vytvorenie nového *m-súboru* pre napísanie SKRIPT-u

- **Skripty** sú postupnosti príkazov uložené do súborov. Sú považované za najjednoduchšie m-fily z toho dôvodu, že nevyžadujú vstupné a výstupné argumenty. Pracujú v spoločnom prostredí programu s globálnymi premennými t.j. pracujú s premennými, ktoré sú definované vo Workspace, ale môžu vytvárať aj vlastné premenné. Hodnoty priradené premenným zostávajú v pamäti aj po vykonaní skriptu.

V čase písania skriptu sa jeho príkazy nevykonávajú. Na ich vykonanie stačí napísať do príkazového okna Command Window MATLABu názov skriptu bez prípony a odoslať na spracovanie.
Niekedy je vhodné nevykonávať určité časti príkazov v M-súbore. Na túto činnosť slúžia komentáre. Komentár začína znakom percenta a končí na konci aktuálneho riadku. Pre lepšiu identifikáciu sa komentáre zvýrazňujú štandardne zelenou farbou.

Príklad 4
Vytvorte postupnosť príkazov - skript, v ktorý vyzve užívateľa na zadanie strán obdĺžnika a následne vypočíta jeho obsah, skript uložte na disk a definujte meno m-súboru



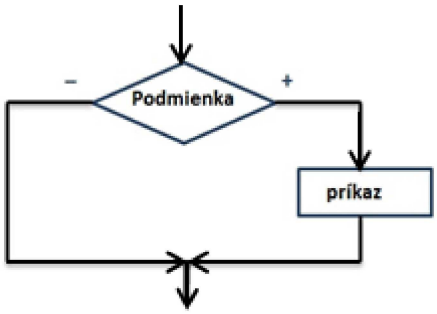
Obrázok 1-7 Vytvorenie skriptu v editore pre výpočet obsahu obdĺžnika

Riadiace štruktúry sú nevyhnutnou súčasťou programovacieho jazyka. Umožňujú riadiť chod programu, jeho vetvenie či opakovanie časti kódu.

- **Jednoduché vetvenie - príkaz if**

Príkaz ***if*** sa používa na jednoduché vetvenie, v prípade kedy je potrebné vykonať dané príkazy len za predpokladu splnenia určitej podmienke. Príkaz vetvenia umožňuje vyhodnotiť podmienku a na základe tohto vyhodnotenia vykonať respektíve nevykonať príkazy v danej vetve. Jeho základná syntax a štruktúra vetvenia znázorňujúca činnosť príkazu sú nasledovné :

```
if podmienka
    prikazy
end
```

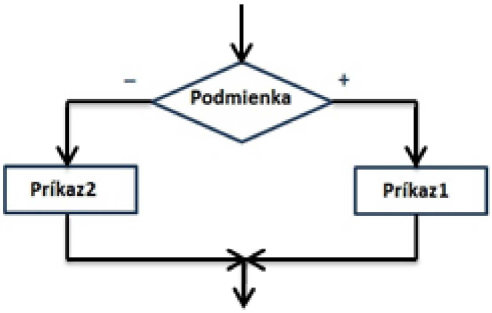


Príklad 5
Ak x < 10, potom vypíšte hlásenie „ číslo x je menšie ako 10“.

```
if x<10
    disp('číslo x je menšie ako 10')
end
```

V prípade potreby vykonania iných príkazov pri nesplnení podmienky sa používa príkaz ***if – else***.
Syntax a štruktúra takto definovaného vetvenia znázorňuje činnosť :

```
if podmienka
    prikaz1
else prikaz2
end
```

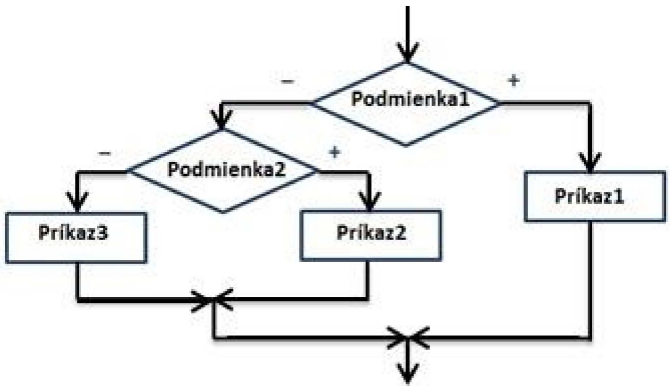


Príklad 6
Ak x < 10 potom vypíšte hlásenie „ číslo x je menšie ako 10“, v opačnom prípade vypíšte hlásenie „číslo x je väčšie alebo rovné číslu 10“.

```
if x<10
    disp('číslo x je menšie ako 10')
else disp('číslo x je väčšie alebo rovné číslu 10')
end
```

Ďalšou modifikáciou príkazu ***if*** je príkaz ***if – elseif – else***.
Syntax a štruktúra vnoreného vetvenia sú nasledovné:

```
if podmienka1
    prikaz1
elseif podmienka2
    prikaz2
else
    prikaz3
end
```



Pri splnení podmienky `podmienka1` sa vykoná príkaz `príkaz1`, a zvyšok príkazu ostane ignorovaný. Pri nesplnení prvej podmienky nasleduje testovanie podmienky `podmienka2`, pri jej kladnom vyhodnotení sa vykoná príkaz `príkaz2`, v opačnom prípade bude vykonaný príkaz `príkaz3`.

Príklad 7
Porovnajte číslo x s číslom 10 a vypíšte príslušné hlásenia.

```
if x<10
    disp('číslo x je menšie ako 10')
elseif x>10
    disp('číslo x je väčšie ako 10')
else disp('číslo x je rovné číslu 10')
end
```

Príklad 8
Zostavte program na určenie minima z 3 čísel zadaných z klávesnice a určte jeho poradie.

Kód v jazyku MATLAB pre Príklad 8

```
a=input('Zadajte číslo: ');
b=input('Zadajte číslo: ');
c=input('Zadajte číslo: ');
min=a;
pozm=1;
if min>b
    min=b;
    pozm=2;
end
if min>c
    min=c;
    pozm=3;
end
fprintf('Hodnota minima zo zadaných čísel je %f \n',min);
fprintf('Pozícia minima zo zadaných čísel je %d \n',pozm);
```

Príklad 9
Vytvorte program na výpočet minimálneho potrebného počtu jázd výťahom. Výťah má obmedzenú nosnosť a sú známe hmotnosti troch cestujúcich, tieto hodnoty zadá užívateľ z klávesnice.

Kód v jazyku MATLAB pre Príklad 9

```
h=input('Zadajte nosnosť výťahu: ');
m1=input('Zadajte hmotnosť 1.pasažiera: ');
m2=input('Zadajte hmotnosť 2.pasažiera: ');
m3=input('Zadajte hmotnosť 3.pasažiera: ');
n=3; %maximálny potrebný počet jázd
if m1+m2<=h | m1+m3<=h | m2+m3<=h %overenie-nájde sa dvojica, ktorá by mohla ísť spolu
   n=2; % nutné sú dve jazdy
end
if m1+m2+m3<=h  % overenie, či by mohli ísť všetci spolu
   n=1; % stačí jedna jazda
end
if m1>h | m2>h | m3>h   % overenie, či určitá hmotnosť nepresahuje nosnosť výťahu
   n=inf;   % nie je možné aby išli všetci výťahom
end
fprintf('Potrebný počet jázd výťahom je: %d \n',n)
```

- **Príkaz switch - case**

V prípade potreby väčšieho počtu vetiev pri jednej podmienke je príkaz vetvenia ***if***, nahradený príkazom ***switch***, ktorého syntax je nasledovná :

```
switch podmienka
    case {hodnota1}
        prikazy1
    case {hodnota2}
        prikazy2
    otherwise
        prikazy3
end
```

`Podmienka` predstavuje hodnotu, ktorú ideme porovnávať s obsahom premenných `hodnota1` a `hodnota2`. Po vyhodnotení zhodnosti s jednou z týchto hodnôt sa vykoná príslušný `príkaz`.
V prípade, že sa `vyraz` nebude zhodovať ani s jedným obsahom `hodnoty`, vykoná sa príkaz `príkaz3`.

Príklad 10
Nech číslo *a* je vstupom do programu. Zistite či zvyšok po delení číslom 7 je párny, nepárny, alebo číslo a je k-násobkom čísla 7.

Kód v jazyku MATLAB pre Príklad 10

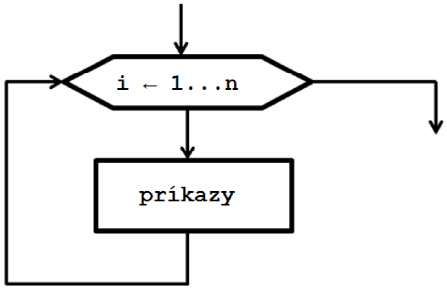
```
a=input('Zadaj celé číslo a : ')
b=mod(a,7)
switch b
    case {1, 3, 5}
        disp('zvyšok po delení číslom 7 je nepárny')
    case {2, 4, 6}
        disp('zvyšok po delení číslom 7 je párny')
    case {0}
        disp('vami zadané číslo je deliteľné číslom 7')
end
```

- **Cykly**

Cykly slúžia na opakované vykonávanie príkazu alebo skupiny príkazov. Cykly môžeme rozdeliť do dvoch skupín a to cykly s pevne daným počtom opakovaní a cykly, pri ktorých nepoznáme počet opakovaní.

- **Cyklus so známym počtom opakovaní - for**

Príkaz ***for*** nám slúži na vykonanie určitých príkazov známy definovaný počet krát. Počet opakovaní určujeme v deklarovaní premennej za príkazom ***for***, v našom prípade ***„i“.*** Určíme počiatočnú hodnotu pre premennú ***i*** od ktorej sa cyklus začne vykonávať, ***krok*** ,s ktorým sa bude cyklus vykonávať. V prípade vynechania časti ***„krok“*** sa za krok automaticky považuje číslo 1. Keď premenna ***i*** dosiahne hodnotu n = počet opakovaní cyklu príkazy v tele cyklu prebehnú posledný krát. Vývojový diagram a syntax príkazu sú na obr.:



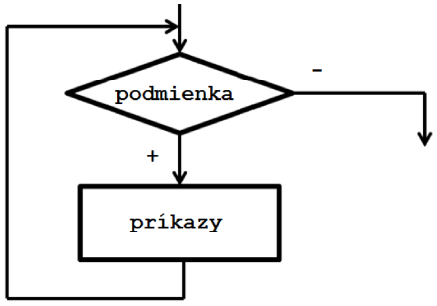

```
for i=zaciatok:krok:koniec
    prikaz
end
```

Príklad 11
Vytvorte vektor ***a*** ktorého prvky budú zodpovedať ich poradiu a dĺžka bude načítaná zo vstupu.

```
n=input('Zadaj dĺžku vektora : ')
for i=1:n
    a(i)=i;
end
disp(a)
```

- **Cyklus while**

Príkaz ***while*** sa využíva väčšinou na vykonanie príkazu, alebo skupiny príkazov s vopred neznámym počtom opakovaní. `Príkaz` sa vykonáva dovtedy, kým je splnená `podmienka`. Syntax príkazu:




```
while podmienka
    prikaz
end
```

Príklad 12
Je zadaná funkcia

$$f(a) = \frac{(a^2-3a-2)}{(a-1)^2(a+2)}$$

Vytvorte program pre výpočet hodnoty definovanej funkcie pre ručne zadaný vstup hodnoty *a*. Použite typ riadiacej štruktúry cyklus, ktorý výpočet vykoná až po zadaní vhodnej hodnoty ***a.***

```
a=input('Zadajte hodnotu a pre ktorú chcete vykonať výpočet výrazu: ');
while (a==1)| (a==-2)
   fprintf('\nZadali ste hodnotu, pre ktorú výraz nemá zmysel.\n')
   a=input('\nZadajte hodnotu a pre ktorú chcete vykonať výpočet výrazu:');
end
x=(a^2-3*a-2)/((a-1)^2*(a+2));
fprintf('Hodnota výrazu:(a^2-3*a-2)/((a-1)^2*(a+2)) pre zadanú hodnotu a=%d je %f',a,x)
```

Príklad 13
Vytvorte maticu s rozmermi 6x8. Použite cyklus, ktorý vytvorí novú maticu, ktorá bude obsahovať prvky pôvodnej matice nachádzajúce sa na pozícií, ktorej poradové číslo stĺpca aj riadku je párne.

Kód v jazyku MATLAB pre Príklad 13

```
M=rand(6,8)
for i=1:6   % cyklus pre prechádzanie riadkov
    for j=1:8   % cyklus pre prechádzanie stĺpcov
        if (rem(i,2)==0 & rem(j,2)==0)
% podmienka: zvyšok po delení čísla i číslom 2 je nula a zároveň zvyšok po
% delení čísla j číslom 2 je nudla
            a=i/2;  % zavedenia nového počítadla riadkov
            b=j/2;  % zavedenie nového počítadla stĺpcov
            A(a,b)=M(i,j);   % vkladanie prvkov z pôvodnej matice do novej
        end
    end
end
A
```

Príklad 14
Predpokladajme, že do „ideálnej banky“, ktorej ročná úroková miera je 5,5% sme vložili 30 000€. Banka úrok pripočítava na konci každého roka zo sumy, ktorá tam je s tou podmienkou, že počas roka sa peniaze nevyberajú. Vypočítajte nárast úroku v nasledujúcich piatich rokoch jednotlivo.

Kód v jazyku MATLAB pre Príklad 14

```
BH(1)=30000*(1+0.055);  %výpočet budúcej hodnoty po prvom roku
for i=2:5
    BH(i)=BH(i-1)*(1+0.055);  %výpočet budúcej hodnoty v ďalších rokoch
end
U(1)=BH(1)-30000; %výpočet úroku, kt. pribudol po prvom roku
for i=1:4
    U(i+1)=BH(i+1)-BH(i);   %výpočet úroku, kt. pribudol v ďalších rokoch
end
U
```

Príklad 15
Vytvorte skript, ktorý vypočíta aritmetický priemer troch najväčších zadaných čísel. Čísla bude zadávať užívateľ z klávesnice a zápis ukončí vložením čísla 0. Ošetrite kód tak, že v prípade, že užívateľ zadal menší počet čísel ako 3 program do aritmetického priemeru vloží „0“.

Kód v jazyku MATLAB pre Príklad 15

```
i=1;
a(1)=input('Zadajte číslo: ');
% kým zadané číslo nie je 0 pokračujte v zadávaní
while a(i)~= 0
    i=i+1;
    a(i)=input('Zadajte číslo: ');
end
% zníženie počítadla o 1  aby hodnota premennej i predstavovaa počet
% zadaných čísel bez čísla 0
i=i-1;
% overenie, či užívateI zadal menší počet čísel ako 3
% v opačno prípade sa vykoná vetva "else"
if i<3
    APmax=0;
else
    % načítanie prvých troch hodnôt do maxím
    for j=1:3
        max(j)=a(j);
    end
    % usporiadaie prvých "maxím" zostupne
    if max(1)<max(2)
        p=max(1);
        max(1)=max(2);
        max(2)=p;
    end
    if max(1)<max(3)
        p=max(1);
        max(1)=max(3);
        max(3)=p;
    end
    if max(2)<max(3)
        p=max(2);
        max(2)=max(3);
        max(3)=p;
    end
    % overenie či zvyšné čísla sú väčšie ako čísla v uložené v premenných
    % max(i)a následné nahradenie hodnoty
    for k=4:i
        if max(3)<a(k)
            max(3)=a(k);
            if max(3)>max(2)
                p=max(2);
                max(2)=max(3);
                max(3)=p;
                if max(2)>max(1)
                    p=max(1);
                    max(1)=max(2);
                    max(2)=p;
                end
            end
        end
    end
    APmax=(max(1)+max(2)+max(3))/3;
end
fprintf('Aritmetický priemer troch najväčších zadaných čísel je: %f',APmax)
```

Príklady na precvičenie

1. Vytvorte program na spravovanie bankového účtu. Užívateľ bude zadávať vklady a výbery prostredníctvom kladných a záporných čísel, ukončenie bude vložením hodnoty 0. Zisti počet výberov, počet príjmov, sumu výberov a sumu príjmov.
2. Naplňte maticu rozmerov 4x4 hodnotami zadanými z klávesnice. Využite pritom cykly.
3. Pomocou cyklu vypíšte sumu všetkých riadkov matice a sumu celej matice spolu.

1.4 Typy matíc a ich reprezentácia v programovom prostredí MATLAB

Matica je určitá množina čísel alebo iných matematických objektov (tzv. prvkov matice) usporiadaných do pravidelných riadkov a stĺpcov.
MATLAB vždy počíta s maticami (t.j. aj skalár je len matica typu 1x1)

- **Typy matíc**

Skalár

- ⇨ matica typu **1** x **1**
- ⇨ pre vytvorenie premennej použijeme priradenie pomocou " =" . Ak nezadáme názov premennej automaticky sa vytvorí premenná ***ans*** (od slovička answer), kde sa uloží hodnota.

```
>> a = 5

a =

     5
```

alebo

```
>> 5

ans =

     5
```

Treba si uvedomiť, že pri každom novom priradení je premenná (vrátane premennej ans) prepisovaná novou hodnotou. Pri novom priradení prídeme o hodnoty uložené v premennej!

Matica m x n

- ⇨ Maticu väčšiu ako 1x1 vždy zapisujeme do hranatých zátvoriek [ ]
- ⇨ Zapisuje sa po riadkoch, pričom jednotlivé elementy (prvky) matice sú oddelené čiarkou alebo medzerou
- ⇨ Nový riadok vytvoríme zadaním bodkočiarky alebo stlačením klávesy ***ENTER*** pričom v tomto prípade ak nie sú zátvorky ukončené, ***ENTER*** neodošle príkaz, iba nás posunie na ďalší riadok

```
>> A=[1,0
2,5
4,3]

A =

     1     0
     2     5
     4     3
```

alebo

```
>> B=[1 0; 2 5; 4 3]

B =

     1     0
     2     5
     4     3
```

alebo

```
>> C=[1 0
2 5
4 3]

C =

     1     0
     2     5
     4     3
```

alebo

```
>> D=[1, 0; 2, 5; 4, 3]

D =

     1     0
     2     5
     4     3
```

Riadkový vektor

- ⇨ matica typu **1** x ***n***
- ⇨ Vytvára sa podobne ako by sme vytvorili maticu ***n x m*** , ale použijeme len medzery alebo čiarky medzi jednotlivými prvkami vektora.
- ⇨ Vytvorenie riadkového vektora je možné aj „***dvojbodkovou konvenciou"*** a to tak, že do hranatých zátvoriek zapíšeme 3 hodnoty :
  počiatočná hodnota,
  hodnota kroku ,
  konečná hodnota.

Túto možnosť je výhodné využívať pri väčších vektoroch, ktoré majú byťinicializované s konštantným krokom.

```
>> E = [1 2 3]

E =

   1   2   3
```

alebo

```
>> E = [1, 2, 3]

E =

   1   2   3
```

alebo

```
>> E=[0:1.2:5]

E =

         0    1.2000    2.4000    3.6000    4.8000
```

⇨ V prípade ak neurčíme hodnotu kroku, bude automaticky rovná 1.

```
>> E=[0:5]

E =

   0   1   2   3   4   5
```

Stĺpcový vektor

⇨ matica typu ***m*** x **1**
⇨ stĺpcový vektor vytvoríme použitím bodkočiarky ";" alebo klávesy ENTER za každou číslicou

```
>> F = [1;2;3]

F =

   1
   2
   3
```

alebo

```
>> F = [1
2
3]

F =

   1
   2
   3
```

Jednotková matica

⇨ štvorcová matica, ktorá obsahuje na hlavnej diagonále samé ***jednotky*** a zvyšné prvky matice tvoria nuly
⇨ takúto maticu vieme vytvoriť dvoma spôsobmi a to tak ako by sme vytvárali maticu ***n x n***, a po jednom vypisovali každý prvok matice, čo je podstatne prácnejšie a náchylnejšie na pomýlenie sa ako druhá možnosť a to je vytvorenie pomocou príkazu :
G = **eye** *(rozmer_matice)*

```
>> G = eye(3)

G =

   1   0   0
   0   1   0
   0   0   1
```

Nulová matica

⇨ matica, ktorej všetky prvky sú **nulové**

⇨ takúto maticu je tiež možné vytvoriť dvoma spôsobmi a to buď prácnym vypisovaním pre každý prvok nulu, alebo použitím príkazu pre štvorcovú maticu *H* = ***zeros(****n)*, alebo príkazu *H* =***zeros(****m,n)* pre maticu typu ***m x n***

```
>> H = zeros(3)

H =

     0     0     0
     0     0     0
     0     0     0
```

alebo

```
>> H = zeros(3,2)

H =

     0     0
     0     0
     0     0
```

Matica jednotiek

⇨ všetky prvky matice sú ***jednotky***
⇨ pre štvorcovú maticu: *I* =***ones(****n)* pre maticu typu ***m x n*** *I* = ***ones(****m,n)*

```
>> I = ones (3)

I =

     1     1     1
     1     1     1
     1     1     1
```

alebo

```
>> I = ones (3,2)

I =

     1     1
     1     1
     1     1
```

Generovanie vektora s ekvidistantným krokom

⇨ pre vytvorenie vektora môžeme použiť príkaz *J* = ***linspace(****poč_hodnota, koneč_hodnota)* pričom tento príkaz vytvorí vektor o 100 hodnotách v intervale <poč_hodnota, koneč_hodnota>, alebo ak chceme vytvoriť ***n-*** hodnotový vektor za konečnú hodnotu napíšeme počet prvkov

J =***linspace(****poč_hodnota, koneč_hodnota, počet_prvkov)*

```
>> J = linspace(1,15)

J =

  Columns 1 through 6

    1.0000    1.1414    1.2828    1.4242    1.5657    1.7071

  Columns 7 through 12

    1.8485    1.9899    2.1313    2.2727    2.4141    2.5556

  Columns 13 through 18

    2.6970    2.8384    2.9798    3.1212    3.2626    3.4040
```

```
>> J = linspace(1,15,10)

J =

  Columns 1 through 6

    1.0000    2.5556    4.1111    5.6667    7.2222    8.7778

  Columns 7 through 10

   10.3333   11.8889   13.4444   15.0000
```

Matica náhodných čísel

⇨ V programovom prostredí MATLAB existuje príkaz ***rand(****veľkosť_matice)*. Tento príkaz vygeneruje maticu náhodných čísel z intervalu <0,1>

```
>> K = rand(2,3)

K =

    0.9572    0.8003    0.4218
    0.4854    0.1419    0.9157
```

⇨ Druhou alternatívou je použiť príkaz ***randn(****veľkosť_matice****)***, ktorý použije čísla z normálneho (Gaussovského) rozdelenia

```
>> K = randn(3,2)

K =

   -0.4336    2.7694
    0.3426   -1.3499
    3.5784    3.0349
```

1.5 Základné operácie s údajovým typom matica

Dáta v prostredí MATLAB, ako sme už spomínali, sú reprezentované maticami. Matica je dvojrozmerné / obdĺžnikové pole reálnych alebo komplexných čísel. Matica, ktorá má *m* riadkov a *n* stĺpcov je nazývaná „matica *m* x *n*“. Každý prvok matici je indexovaný dvojitým indexom. napr a_{ij}, kde *i* predstavuje poradie riadku a *j* poradie stĺpca, v ktorom sa prvok nachádza. Môžeme použiť aj skalárnu veličinu, avšak v Matlabe je tiež chápaná ako matica 1x1.

Maticu môžeme zadať niekoľkými spôsobmi, a to :
- ⇨ zadaním po prvkoch (prvky v riadku sú oddelené medzerou/čiarkou, v stĺpcoch + bodkočiarkou)
- ⇨ pomocou funkcie – špeciálne matice (***eye, ones, rand***...)
- ⇨ nahraním zo súboru

- **Operátor „ : “**

Často používaný operátor, dvojbodkovej konvencie sa používa pri tvorbe postupností s konštantným krokom.

Jeho syntax je :

```
v = začiatok : krok : koniec
```

Kde
- „krok“ je možné vynechať a vtedy sa bude chápať krok = 1
- „krok“ môže byť aj záporné číslo, vtedy sa vytvára zostupná postupnosť

Príklad 1
V príkazovom režime vytvorte vektor v1 s prvkami od 1 po 9 s krokom 2 a vektor v2 ktorého prvky budú zostupnou postupnosťou čísel od 15 po 3 s krokom 3 pomocou dvojbodkovej konvencie.

```
>> v1=1:2:9

v1 =

     1     3     5     7     9
```

```
>> v2=15:-3:3

v2 =

    15    12     9     6     3
```

- **Funkcia *linspace* na generovanie vektorov**

Funkcia ***linspace*** má podobnú funkciu ako predchádzajúci operátor „ : “ s tým rozdielom, že si presne vypočíta krok. Je výhodné použiť túto funkciu v prípadoch, kedy potrebujeme vytvoriť vektor s vopred známym počtom prvkov, ale neznámym krokom. Syntax funkcie je:

```
v = linspace(začiatok, koniec, počet_prvkov)
```

Príklad 2
V príkazovom režime vytvorte vektor v3, s počtom prvkov 5, ktoré budú z intervalu <0;π>. Príklad riešte s využitím funkcie ***linspace***.

```
>> v3=linspace(0,pi,5)

v3 =

         0    0.7854    1.5708    2.3562    3.1416
```

- **Výber prvkov matice a submatica**

Ako sme už spomenuli, každý prvok matice je indexovaný v tvare a_{ij}. Pri výbere konkrétneho prvku matice stačí zadať príkaz :

```
Matica(číslo_riadku, číslo_stĺpca)
```

Pri výbere časti matice – submatice nám pomáha operácia dvojbodkovej konvencie „ : “. Je to podobne ako pri výbere prvku s tým rozdielom, že namiesto „číslo_riadku“ sa dá rozmedzie riadkov, ktoré chceme vybrať. A rovnako aj namiesto „číslo_stĺpca“ dáme rozmedzie stĺpcov, ktoré má submatica obsahovať.
Príkaz:

```
A=M(x:y,x:y)
```

kde

x – poradie riadka/stĺpca, od ktorého chceme vytvoriť submaticu
y - poradie riadka/stĺpca, pri ktorom chceme ukončiť výber

Príklad 3
Vytvorte maticu M(4,4) pričom jej prvky a_{ij} sú celé čísla

$$M = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 9 & 10 & 11 & 12 \\ 13 & 14 & 15 & 16 \end{pmatrix}$$

a) hodnotu prvku 3.riadku a 2. stĺpca vložte do premennej *a*
b) vytvorte submaticu *B* z matice *M*, ktorá bude tvorená prvkami 1.-2. riadku, 2.-4. stĺpca.

```
>> M=[1 2 3 4; 5 6 7 8; 9 10 11 12; 13 14 15 16]

M =

     1     2     3     4
     5     6     7     8
     9    10    11    12
    13    14    15    16

>> a=M(3,2)

a =

    10

>> B=M(1:2,2:end)

B =

     2     3     4
     6     7     8
```

1.6 Príklady na riešenie

Riadiace štruktúry

1. Vytvorte vektor v, ktorý bude obsahovať 10 náhodných čísel z intervalu <-10 ; 10>.

2. Zistite koľko prvkov z vektora v je kladných, koľko záporných a koľko rovných nule.

3. Vytvorte vektory k a z. Do vektora k vložte všetky prvky z vektora v, ktoré sú kladné. Do vektora z vložte všetky prvky z vektora v, ktoré sú záporné. Vypíšte na obrazovku vektory v, k, z.

4. Vytvorte pomocou cyklu maticu rozmerov 4x4, ktorá bude obsahovať len párne čísla. Čísla do matice sa budú vkladať z klávesnice. Zadajte podmienku, aby sa do matice dali vložiť len párne čísla.

5. Vytvorte cyklus, ktorý vypíše minimálne hodnoty stĺpcov z ľubovoľnej matice, ktorú zadáte. Následne porovnajte výsledky, ktoré vyhodnotil vás cyklus s výsledkami, ktoré vygeneroval príkaz min().

6. Vytvorte maticu ľubovoľných hodnôt s rozmermi 5x5. Napíšte cyklus na vytvorenie vektora, ktorý bude obsahovať prvky hlavnej diagonálu zadanej matice. Vypočítajte sumu tohto vektora, výsledok vypíšte.

7. Predpokladajme, že do banky, ktorá má ročnú úrokovú mieru 7% vložíme 10 000€.
   a. zistime za koľko rokov budeme mať aspoň 20 000€
   b. ako sa nám bude meniť suma v jednotlivých rokoch

Matice a vektory

1. Do premennej *x* vložte 10 čísel v rozmedzí čísel od 1 do 5 pomocou funkcie linspace.
2. Do premennej *y* vložte logaritmy hodnôt nachádzajúcich sa vo vektore *x*.
3. Do premennej *z* vložte hodnoty premennej *x* umocnené na druhú.
4. Vytvorte vektor *s,* ktorého počet prvkov bude 8, prvky budú usporiadané zostupne a ich hodnoty budú v rozmedzí od 23 po 9 pomocou dvojbodkovej konvencie.
5. Vytvorte maticu M, ktorá bude mať v prvom riadku čísla od 3 po7, v druhom riadku bude tak isto päť prvkov ktoré budú v rozmedzí 10 – 23 a v treťom riadku nech sú čísla v zostupnom poradí od 19 do 5 s rovnomerným rozdelením.
   a. Vytvorte submaticu A matice *M*, ktorá bude obsahovať prvky 2-3riadku a 3-4stĺpca
   b. Vytvorte submaticu B matice *M*, ktorá bude obsahovať prvky 3-koncového riadku a 1-2 stĺpca, tak aby prvky boli v riadku usporiadané v opačnom poradí oproti matici *M*
6. Vytvorte ľubovoľnú maticu K s rozmermi 8x8. Vytvorte ľubovoľnú submaticu *C* matice *M*. Z matice *C* vyberte prvok nachádzajúci sa na pozícií [1,3] a vložte ho na pozíciu [1,4]. Zobrazte maticu *C* pred aj po presune prvku [1,3]
7. Vytvorte premennú *t ,* ktorá bude obsahovať 20 hodnôt funkcie *cos(x)* pričom x bude v rozmedzí od <-10:10>. Hodnoty ktoré získate vypíšte na obrazovku .

1.7 Operácie s maticami

- **Základné operátory**

Znak	Opis	Znak	Opis
+	Plus	**.**	Desatinná bodka
-	Mínus	**%**	Komentár
*	Maticové násobenie	**=**	Priradenie
.*	Násobenie po prvkoch	**==**	Zhodnosť
^	Umocnenie	**&**	Logický AND
. ^	Umocnenie po prvkoch	**|**	Logický OR
\	Ľavé delenie	**~**	Logická negácia
/	Pravé delenie		
./	Pravé delenie prvkov		

Sčítanie a odčítanie matíc

Uvažujme matice ***A*** *(m, n)* a ***B*** *(m, n)*. Ich súčet je definovaný nasledovne:

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix} + \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ b_{m1} & b_{m2} & \cdots & b_{mn} \end{pmatrix} =$$

$$= \begin{pmatrix} a_{11}+b_{11} & a_{12}+b_{12} & \cdots & a_{1n}+b_{1n} \\ a_{21}+b_{11} & a_{22}+b_{22} & \cdots & a_{2n}+b_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1}+b_{m1} & a_{m2}+b_{m2} & \cdots & a_{mn}+b_{mn} \end{pmatrix}$$

Riešenie v programovom prostredí MATLAB:

```
>> A = [1,2,3;4,5,6;7,8,9];
>> B=[7,8,9;4,5,6;1,2,3];
>> C=A+B

C =

    8   10   12
    8   10   12
    8   10   12
```

Pozn. Odčítanie dvoch matíc je definované ako pripočítanie opačnej matice, t.j. platia rovnaké podmienky pre rozmery matíc ako pri sčítaní.
Operácie sčítania a odčítania sú definované len pre matice rovnakých rozmerov, inak MATLAB vypíše chybu kvôli nezhode rozmerov matíc.

```
>> A=[1 0;2 5;4 3];
>> B=[2 3;5 8];
>> A-B
??? Error using ==> minus
Matrix dimensions must agree.

>> A+B
??? Error using ==> plus
Matrix dimensions must agree.
```

⇨ výnimkou je sčítanie a odčítanie skaláru od matice

```
>> A=[2 3;5 8];
>> B= A+2

B =

   4   5
   7   10

>> C=A-2

C =

   0   1
   3   6
```

Násobenie matíc

⇨ dve matice rozmerov ***m x n*** a ***p x r*** môžeme násobiť iba za predpokladu, že ***n = p***; pričom ako výsledok dostaneme maticu veľkosti ***m x r***

⇨ V prípade že neplatí podmienka ***n = p***, vypíše MATLAB chybu. Výnimku opäť tvorí skalár, ktorým vieme násobiť matice.

```
>> A=[1 0;2 5;4 3];
>> B=[2 3;5 8];
>> C=B*A
??? Error using ==> mtimes
Inner matrix dimensions must agree.
```

Príklad 1

Vypočítajte súčin daných matíc A(m,n) , B(p,r) a overte v programovom prostredí MATLAB

$$A=\begin{pmatrix}1 & 0\\ 2 & 5\\ 4 & 3\end{pmatrix},\quad B=\begin{pmatrix}2 & 3\\ 5 & 8\end{pmatrix}.$$

Ručný výpočet súčinu matíc (podmienky pre násobenie matíc sú evidentne splnené):

$$\begin{pmatrix}1 & 0\\ 2 & 5\\ 4 & 3\end{pmatrix}*\begin{pmatrix}2 & 3\\ 5 & 8\end{pmatrix}=\begin{pmatrix}2+0 & 3+0\\ 4+25 & 6+40\\ 8+15 & 12+24\end{pmatrix}=\begin{pmatrix}2 & 3\\ 29 & 46\\ 23 & 36\end{pmatrix}$$

Súčin matíc v programovom prostredí MATLAB:

```
>> A=[1 0;2 5;4 3]

A =

   1   0
   2   5
   4   3

>> B=[2 3;5 8]

B =

   2   3
   5   8
```

...

```
>> C=A*B

C =

   2   3
   29  46
   23  36
```

Transponovaná matica

⇨ matica, ktorá vznikne výmenou jednotlivých riadkov za stĺpce a naopak, t.j. ak $\boldsymbol{B} = \boldsymbol{A}^T$, tak platí $b_{ij} = a_{ji}$

Príklad 2

Vytvorte transponovanú maticu k matici $\boldsymbol{A}(m,n)$, ktorá je zadaná v tvare :

$$\boldsymbol{A} = \begin{pmatrix} 1 & 0 \\ 2 & 5 \\ 4 & 3 \end{pmatrix}$$

a následne overte svoje riešenie v programovom prostredí MATLAB.

Transponovaná matica A:

$$\boldsymbol{A}^T = \begin{pmatrix} 1 & 2 & 4 \\ 0 & 5 & 3 \end{pmatrix}$$

Riešenie v programovom prostredí MATLAB:

```
>> A=[1 0;2 5;4 3]

A =

   1   0
   2   5
   4   3

>> D=A'

D =

   1   2   4
   0   5   3
```

1.8 Operácie po prvkoch matice (element-by-element)

Pri použití operácií element-by-element sa príslušný operátor aplikuje postupne na každú dvojicu prvkov uvažovaných matíc nezávisle od seba. Pre zápis takýchto operácií používame bodku ". " pred operátorom. Matice vystupujúce ako operandy musia mať rovnaké rozmery, inak MATLAB vypíše chybu.

```
>> A=[1 2 3; 4 5 6];
>> B=[9 8 7 ; 6 5 4 ;3 2 1];
>> C=A.*B
??? Error using ==> times
Matrix dimensions must agree.
```

Násobenie po prvkoch – vynásobenie každého prvku matice *A* prvkom matice *B* na príslušnej pozícii v riadku a stĺpci

$$\begin{pmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{1n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{pmatrix} .* \begin{pmatrix} b_{11} & b_{12} & \dots & b_{1n} \\ b_{21} & b_{22} & \dots & b_{1n} \\ \vdots & \vdots & \vdots & \vdots \\ b_{m1} & b_{m2} & \dots & b_{mn} \end{pmatrix} =$$

$$= \begin{pmatrix} a_{11} * b_{11} & a_{12} * b_{12} & \dots & a_{1n} * b_{1n} \\ a_{21} * b_{21} & a_{22} * b_{22} & \dots & a_{2n} * b_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} * b_{m1} & a_{m2} * b_{m2} & \dots & a_{mn} * b_{mn} \end{pmatrix}$$

Výsledok operácie násobenia po prvkoch v programovom prostredí MATLAB**:**

```
>> A=[1 2 3 ; 4 5 6; 7 8 9];
>> B=[9 8 7 ; 6 5 4 ;3 2 1];
>> C=A.*B

C =

    9   16   21
   24   25   24
   21   16    9
```

Z definície násobenia skalárom vyplýva, že rovnaký výsledok dostávame pri štandardnom násobení a násobení po prvkoch:

```
>> A=[1 2 3 ; 4 5 6; 7 8 9];
>> C= A.*5

C =

    5   10   15
   20   25   30
   35   40   45
```

```
>> C = A*5

C =

    5   10   15
   20   25   30
   35   40   45
```

Delenie po prvkoch – vydelenie každého prvku matice *A* prvkom matice *B* na príslušnej pozícii v riadku a stĺpci

Delenie tak ako násobenie po prvkoch musí byť vykonávané s maticami rovnakých rozmerov, inak programový system MATLAB zahlási chybu.

```
>> A=[5,10,15;20,25,30;35,40,45];
>> B=[1,2,3;4,5,6];
>> C=A./B
??? Error using ==> rdivide
Matrix dimensions must agree.
```

```
>> A=[5,10,15;20,25,30;35,40,45];
>> B=[1,2,3;4,5,6;7,8,9];
>> C=A./B

C =

    5    5    5
    5    5    5
    5    5    5
```

Umocňovanie po prvkoch – umocnenie každého prvku matice *A* prvkom matice *B* na príslušnej pozícii v riadku a stĺpci

$$\begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{1n} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix} . \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1n} \\ b_{21} & b_{22} & \cdots & b_{1n} \\ \vdots & \vdots & \vdots & \vdots \\ b_{m1} & b_{m2} & \cdots & b_{mn} \end{pmatrix} = \begin{pmatrix} a_{11}{}^{b_{11}} & a_{12}{}^{b_{12}} & \cdots & a_{1n}{}^{b_{1n}} \\ a_{21}{}^{b_{21}} & a_{22}{}^{b_{22}} & \cdots & a_{2n}{}^{b_{2n}} \\ \vdots & \vdots & \vdots & \vdots \\ a_{m1}{}^{b_{m1}} & a_{m2}{}^{b_{m2}} & \cdots & a_{mn}{}^{b_{mn}} \end{pmatrix}$$

Výsledok umocňovania po prvkoch v programovom prostredí MATLAB:

```
>> A=[2,3,5;4,5,2;1,3,2];
>> B=[2,2,1;2,1,2;3,1,3];
>> C=A.^B

C =

    4    9    5
   16    5    4
    1    3    8
```

1.9 Riešenie systémov algebraických rovnic v jazyku MATLAB

Nech je daný **systém lineárnych algebraických rovníc** *n*-tého rádu vo všeobecnom tvare:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{n1}x_1 + a_{n2}x_2 + \ldots + a_{nn}x_n &= b_n \end{aligned} \tag{1}$$

kde $a_{11}, a_{12}, \ldots, a_{nn}, b_1, \ldots, b_n$ sú reálne čísla a usporiadaná *n*-tica $\{x_1, x_2, ..., x_n\}$ predstavuje riešenie systému.
Vzhľadom na pravidlá násobenia matíc môžeme systém (1) zapísať v maticovom tvare

$$\boxed{\boldsymbol{Ax} = \boldsymbol{b}} \tag{2}$$

ktorého matice koeficientov majú tvar $\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \ldots & a_{nn} \end{bmatrix}$ (matica rozmeru *nxn*) a $\boldsymbol{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$ (stĺpcový vektor rozmeru *nx1*). Riešením systému je stĺpcový vektor $\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$ rozmeru *nx1*.

Pokiaľ je matica $\boldsymbol{A}$ **regulárna** (jej determinant je rôzny od 0, resp. matica $\boldsymbol{A}$ má plnú hodnosť), potom k nej **existuje inverzná matica** $\boldsymbol{A}^{-1}$, pre ktorú platí

$$\boldsymbol{A}\boldsymbol{A}^{-1} = \boldsymbol{A}^{-1}\boldsymbol{A} = \boldsymbol{I} \tag{3}$$

kde $\boldsymbol{I} = \begin{bmatrix} 1 & 0 & \ldots & 0 \\ 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & 1 \end{bmatrix}$ je jednotková matica *n*-tého rádu.

Ak teda systém (2) vynásobíme **zľava** maticou $\boldsymbol{A}^{-1}$, naľavo sa osamostatní vektor $\boldsymbol{x}$ (pretože $\boldsymbol{Ix} = \boldsymbol{x}$):

$$\begin{aligned} \boldsymbol{A}^{-1}\boldsymbol{Ax} &= \boldsymbol{A}^{-1}\boldsymbol{b} \\ \boldsymbol{Ix} &= \boldsymbol{A}^{-1}\boldsymbol{b} \end{aligned} \tag{4}$$

a riešenie systému možno vyjadriť v tvare:

$$\boxed{\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{b}} \tag{5}$$

***Pomôcka**:* Princíp (5) je analogický princípu, podľa ktorého sa riešia lineárne rovnice. Ak je daná lineárna rovnica (t.j. 1 rovnica 1. rádu s 1 riešením), napr.

$$5x = 8\,, \tag{6}$$

tak riešenie spočíva v tom, že vynásobíme ľavú aj pravú stranu prevrátenou hodnotou ku koeficientu, ktorý sa nachádza pri *x*, v tomto prípade $\frac{1}{5} = 5^{-1}$. Keďže platí, že súčin čísla a jeho prevrátenej hodnoty je 1 (analógia s (3)), tak riešenie rovnice dostávame v tvare

$$\begin{gathered} \frac{1}{5}5x = \frac{1}{5}8 \\ \boxed{x = \frac{8}{5}} \end{gathered} \tag{7}$$

Inak povedané, riešenie sme dostali **vydelením pravej strany koeficientom pri *x***. Je to síce triviálne tvrdenie, ale ak si ho porovnáme so vzťahom (5), môžeme povedať, že inverzia je istou analógiou delenia. Tento záver využijeme nižšie pri popise operácií ľavého a pravého maticového delenia.

Príklad 1. *Použitím programového prostredia MATLAB riešte nasledujúci systém lineárnych algebraických rovníc 3. rádu metódou a) násobenia inverznou maticou zľava b) ľavého maticového delenia:*

$$\begin{aligned} 2x_1 + 5x_2 + 3x_3 &= 1 \\ 4x_1 + 6x_2 + 2x_3 &= 5 \\ x_1 - 5x_2 + 3x_3 &= 3 \end{aligned} \tag{8}$$

Riešenie: *a) V MATLABe si vytvoríme matice* ***A***, ***b*** *a využitím príkazu* **`inv`** *na výpočet inverznej matice určíme riešenie systému* ***x*** *zo vzťahu (5):*

```
>> A=[2 5 3; 4 6 2; 1 -5 3];
>> b=[1; 5; 3];
>> x=inv(A)*b

x =

    2.0278
   -0.4028
   -0.3472
```

b) Operácia vynásobenia stĺpcového vektora inverznou maticou zľava je ekvivalentná s ***operáciou ľavého maticového delenia*** *v MATLABe. (Pomôcka: Uvažujte s inverziou ako s analógiou delenia a vzťah (5) si predstavujte ako „****b*** *delené* ***A****“.)*

```
>> x=A\b

x =

    2.0278
   -0.4028
   -0.3472
```

Systém lineárnych algebraických rovníc (1) možno okrem štandardného maticového vyjadrenia (2) zapísať aj v nasledovnom ekvivalentnom maticovom tvare:

$$\boxed{\boldsymbol{x}_E \boldsymbol{A}_E = \boldsymbol{b}_E} \tag{9}$$

pričom matice koeficientov majú tvar $\boldsymbol{A}_E = \boldsymbol{A}^T = \begin{bmatrix} a_{11} & a_{21} & \dots & a_{n1} \\ a_{12} & a_{22} & \dots & a_{n2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1n} & a_{2n} & \dots & a_{nn} \end{bmatrix}$ (matica rozmeru *nxn*),

$\boldsymbol{b}_E = \boldsymbol{b}^T = \begin{bmatrix} b_1 & b_2 & \cdots & b_n \end{bmatrix}$ (riadkový vektor rozmeru *nx1*). Riešením systému je riadkový vektor $\boldsymbol{x}_E = \boldsymbol{x}^T = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}$ rozmeru *nx1*. (*Odporúčanie:* Roznásobením po prvkoch overte, že zápis (9) je ekvivalentný zápisu (1) aj (2).)

Riešenie $\boldsymbol{x}_E$ systému (9) dostaneme, ak obidve strany rovnice vynásobíme **sprava** maticou $\boldsymbol{A}_E^{-1}$, čím sa podobne ako v predchádzajúcom prípade naľavo osamostatní vektor $\boldsymbol{x}_E$:

$$\begin{aligned} \boldsymbol{x}_E \boldsymbol{A}_E \boldsymbol{A}_E^{-1} &= \boldsymbol{b}_E \boldsymbol{A}_E^{-1} \\ \boldsymbol{x}_E \boldsymbol{I} &= \boldsymbol{b}_E \boldsymbol{A}_E^{-1} \end{aligned} \tag{10}$$

a riešenie systému nadobudne tvar:

$$\boxed{\boldsymbol{x}_E = \boldsymbol{b}_E \boldsymbol{A}_E^{-1} = \boldsymbol{b}^T \left(\boldsymbol{A}^T \right)^{-1}} \tag{11}$$

resp. (pokiaľ chceme riešenie vyjadriť v tvare stĺpcového, nie riadkového vektora):

$$\boxed{\boldsymbol{x} = \boldsymbol{x}_E^T = \left(\boldsymbol{b}^T \left(\boldsymbol{A}^T \right)^{-1} \right)^T} \tag{12}$$

Príklad 2. Použitím programového prostredia MATLAB a maticového zápisu (9) riešte systém lineárnych algebraických rovníc 3. rádu z príkladu 1 metódou
a) násobenia inverznou maticou sprava,
b) pravého maticového delenia.

Riešenie:
a) Podobne ako v príklade 1 si v MATLABe vytvoríme matice ***A***, ***b*** systému a využitím príkazu `inv` na výpočet inverznej matice určíme riešenie systému buď v tvare riadkového vektora $\boldsymbol{x}_E$ zo vzťahu (11), alebo v tvare stĺpcového vektora $\boldsymbol{x}$ zo vzťahu (12):

```
>> A=[2 5 3; 4 6 2; 1 -5 3];
>> b=[1; 5; 3];
>> xe=b'*inv(A')

xe =

    2.0278   -0.4028   -0.3472

>> x=(b'*inv(A'))'

x =

    2.0278
   -0.4028
   -0.3472
```

b) Operácia vynásobenia riadkového vektora inverznou maticou sprava je ekvivalentná s **operáciou pravého maticového delenia** v MATLABe. (Pomôcka: Uvažujte s inverziou ako s analógiou delenia a vzťah (5) si predstavujte ako „$\boldsymbol{b}^T$ delené $\boldsymbol{A}^T$“.)

```
>> xe=b'/A'

xe =

    2.0278   -0.4028   -0.3472

>> x=(b'/A')'

x =

    2.0278
   -0.4028
   -0.3472
```