POČÍTAČOVÁ PODPORA V ELEKTROTECHNICE

Ing. Lenka Šroubová, Ph.D.
lsroubov@kte.zcu.cz

Ing. Petr Kropík, Ph.D.
pkropik@kte.zcu.cz

Katedra teoretické elektrotechniky
Fakulta elektrotechnická
Západočeská univerzita v Plzni

Místnost: EK602

FACULTY OF ELECTRICAL ENGINEERING
UNIVERSITY OF WEST BOHEMIA

Funkce pro vstup a výstup

Formátovaný textový výstup

`fprintf ('formatovací sekvence', argumenty)`

– vypíše argumenty v požadovaném formátu v pořadí v jakém jsou zapsány. Formátovací sekvence – začínají znakem `%` :

`%d` nebo `%i` – desítkové číslo (znaménkové)
`%o` – číslo v osmičkové soustavě
`%u` – desítkové číslo (neznaménkové)
`%x` nebo `%X` – číslo v šestnáctkové soustavě (`a` – `f` nebo `A` – `F`)
`%f` – desetinné číslo
`%e` nebo `%E` – desetinné číslo v exponenciálním tvaru (`e` nebo `E`)
`%g` nebo `%G` – použito `%f` nebo `%e` resp. `%E` – exponenciální tvar se použije, je-li třeba, tj. je-li číslo moc velké nebo malé. Navíc vypouští nevýznamné nuly
`%c` – tisk znaku (z proměnné)
`%s` – tisk textového řetězce (z proměnné)

Pozn.
`%%` - tiskne znak `%`

Funkce pro vstup a výstup

Formátovaný textový výstup

`fprintf`

Důležité řídící znaky:

`\n` – nová řádka
`\t` – tabelátor
`\r` – návrat na začátek téhož řádku (záleží na operačním systému)
`\a` – v některých operačních systémech – pípnutí
`\b` – Backspace – vymazání předchozího znaku
`\\` - tiskne znak \ (zpětné lomítko)

Příklad:

```
a = 56; b = 134;
```

– tisk hodnot uložených v proměnných `a`, `b` na obrazovku

```
fprintf('%d %d\n', a, b);
56 134
```

Funkce pro vstup a výstup

Formátovaný textový výstup

fprintf

Příklad:

```
a = 56; b = 134;
```

- tisk hodnot na 8 znaků, zleva doplněny **mezery**

```
fprintf('%8d %8d\n',a,b);
      56      134
```

- tisk hodnot na 8 znaků, zleva doplní poprvé **nuly**, podruhé **mezery**

```
fprintf('%08d %8d\n',a,b);
00000056      134
```

Funkce pro vstup a výstup

Formátovaný textový výstup

fprintf

Příklad:

```
a = 56; b = 134;
```

- tisk hodnot na 8 znaků, zleva doplněny **mezery**

```
fprintf('%8d %8d\n',a,b);
      56      134
```

- tisk hodnot na 8 znaků, zleva doplní poprvé **nuly**, podruhé **mezery**

```
fprintf('%08d %8d\n',a,b);
00000056      134
```

6+2=8

5+3=8

Funkce pro vstup a výstup

Formátovaný textový výstup

fprintf

– uvedené číslo říká "alespoň 8 znaků", nesmí oříznout delší celé číslo.

Příklad:

```
a = 56; b = 134;
```

- tisk hodnot na 8 znaků, zleva doplněny **mezery**

```
fprintf('%8d %8d\n',a,b);
      56      134
```

- tisk hodnot na 8 znaků, zleva doplní poprvé **nuly**, podruhé **mezery**

```
fprintf('%08d %8d\n',a,b);
00000056      134
```

6+2=8

5+3=8

```
fprintf('%6d\n',1000);
  1000
fprintf('%4d\n',1000);
1000
fprintf('%2d\n',1000);
1000
```

Funkce pro vstup a výstup

Formátovaný textový výstup

fprintf

– uvedené číslo říká "alespoň 8 znaků", nesmí oříznout delší celé číslo.

Příklad:

```
a = 56; b = 134;
```

- tisk hodnot na 8 znaků, zleva doplněny mezery

```
fprintf('%8d %8d\n',a,b);
      56      134
```

číslo 1000 má 4 číslice, 6 znaků => 2 mezery, 2 +4= 6

- tisk hodnot na 8 znaků, zleva doplní poprvé nuly, podruhé mezery

```
fprintf('%08d %8d\n',a,b);
00000056      134
```

```
fprintf('%6d\n',1000);
  1000
fprintf('%4d\n',1000);
1000
fprintf('%2d\n',1000);
1000
```

6+2=8

5+3=8

4 znaky => žádná mezera, 0 +4= 4

2 znaky => žádná mezera, celé číslo 1000

Funkce pro vstup a výstup

Formátovaný textový výstup

`fprintf`

Příklad:

```
x = 123.3456;
```

– výpis **desetinného** čísla

```
fprintf('%f\n', x);
123.345600
```

– výpis **desetinného** čísla na **tři desetinná místa**

```
fprintf('%.3f\n', x);
123.346
```

`fprintf` zaokrouhluje

– výpis **desetinného** čísla na **dvě desetinná místa**

```
fprintf('%.2f\n', x);
123.35
```

– výpis **desetinného** čísla na **osm desetinných míst**

```
fprintf('%.8f\n', x);
123.34560000
```

Pozn. `%f` – zobrazení čísla v pevné řádové čárce

Funkce pro vstup a výstup

Formátovaný textový výstup

`fprintf`

Příklad:

```
x = 123.3456;
```

– výpis **desetinného** čísla na celkem **15** znaků včetně **desetinné tečky** a z toho budou **2 desetinná místa** (tj. na celou část zbývá minimálně **12**, **celá část** má **3** číslice =>doplněno **9 mezerami**)

```
fprintf('%15.2f\n', x);
           123.35
```

Funkce pro vstup a výstup

Formátovaný textový výstup

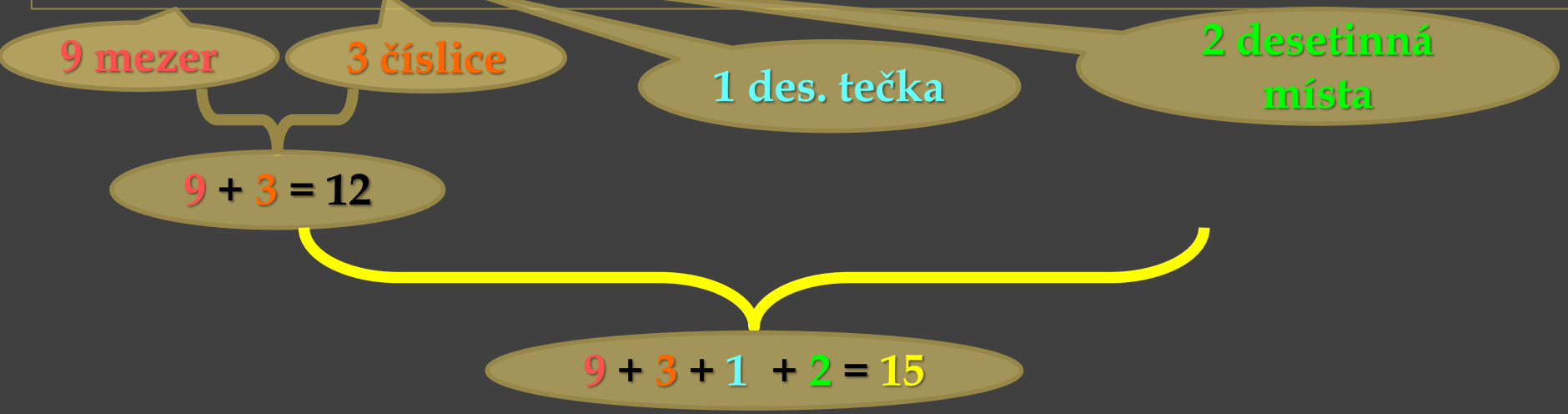
`fprintf`

Příklad:

```
x = 123.3456;
```

– výpis **desetinného** čísla na celkem **15** znaků včetně **desetinné tečky** a z toho budou **2 desetinná místa** (tj. na celou část zbývá minimálně **12**, **celá část** má **3** číslice =>doplněno **9 mezerami**)

```
fprintf('%15.2f\n', x);
          123.35
```



Funkce pro vstup a výstup

Formátovaný textový výstup

fprintf

Pokračování příkladu:

```
x = 123.3456;
```

– výpis desetinného čísla na celkem 15 znaků včetně desetinné tečky a z toho budou 2 desetinná místa (doplněno 9 mezerami)

```
fprintf('%15.2f\n', x);
          123.35
```

– a teď doplněno 9 nulami

```
fprintf('%015.2f\n',x);
000000000123.35
```

Funkce pro vstup a výstup

Formátovaný textový výstup

fprintf

Pokračování příkladu:

```
x = 123.3456;
```

– výpis desetinného čísla na celkem 15 znaků včetně desetinné tečky a z toho budou 2 desetinná místa (doplněno 9 mezerami)

```
fprintf('%15.2f\n', x);
_________123.35
```

– a teď doplněno 9 nulami

```
fprintf('%015.2f\n',x);
000000000123.35
```

Funkce pro vstup a výstup

Formátovaný textový výstup

fprintf

Pokračování příkladu:

```
x = 123.3456;
```

– výpis **desetinného** čísla na celkem **15** znaků včetně **desetinné tečky** a z toho budou **2 desetinná místa** (doplněno **9 mezerami**)

```
fprintf('%15.2f\n', x);
         123.35
```

– a teď doplněno **9 nulami**

```
fprintf('%015.2f\n',x);
000000000123.35
```

```
fprintf('%06.2f\n',x);
123.35
```

– **nula** není doplněna, už je vypsáno **6** znaků (**2** desetinná místa, **1** tečka a **3** znaky celku)

Funkce pro vstup a výstup

Formátovaný textový výstup

fprintf

Pokračování příkladu:

```
x = 123.3456;
```

– výpis **desetinného** čísla na celkem **15** znaků včetně **desetinné tečky** a z toho budou **2 desetinná místa** (doplněno **9 mezerami**)

```
fprintf('%15.2f\n', x);
          123.35
```

– a teď doplněno **9 nulami**

```
fprintf('%015.2f\n',x);
000000000123.35
```

3 + 1 + 2 = 6

```
fprintf('%06.2f\n',x);
123.35
```

– **nula** není doplněna, už je vypsáno **6** znaků (**2** desetinná místa, **1** tečka a **3** znaky celku)

Funkce pro vstup a výstup

Formátovaný textový výstup

fprintf

Pokračování příkladu:

```
x = 123.3456;
```

– výpis **desetinného** čísla na celkem **15** znaků včetně **desetinné tečky** a z toho budou **2 desetinná místa** (doplněno **9 mezerami**)

```
fprintf('%15.2f\n', x);
          123.35
```

– a teď doplněno **9 nulami**

```
fprintf('%015.2f\n',x);
000000000123.35
```

3 + 1 + 2 = 6

```
fprintf('%06.2f\n',x);
123.35
```

– **nula** není doplněna, už je vypsáno **6** znaků (**2** desetinná místa, **1** tečka a **3** znaky celku)

```
fprintf('%04.2f\n',x);
123.35
```

– vypsáno **6** znaků (**2** desetinná místa, **1** tečka a **3** znaky celku)

Funkce pro vstup a výstup

Formátovaný textový výstup

fprintf

Pokračování příkladu:

```
x = 123.3456;
```

– výpis **desetinného** čísla na celkem **15** znaků včetně **desetinné tečky** a z toho budou **2 desetinná místa** (doplněno 9 **mezerami**)

```
fprintf('%15.2f\n', x);
          123.35
```

– a teď doplněno **9 nulami**

```
fprintf('%015.2f\n',x);
000000000123.35
```

3 + 1 + 2 = 6

```
fprintf('%06.2f\n',x);
123.35
```

– **nula** není doplněna, už je vypsáno **6** znaků (**2** desetinná místa, **1** tečka a **3** znaky celku)

```
fprintf('%04.2f\n',x);
123.35
```

4 - 1 - 2 = 1

– vypsáno **6** znaků (**2** desetinná místa, **1** tečka a **3** znaky celku)

Funkce pro vstup a výstup

Formátovaný textový výstup

fprintf

Pokračování příkladu:

```
x = 123.3456;
```

– výpis **desetinného** čísla na celkem **15** znaků včetně **desetinné tečky** a z toho budou **2 desetinná místa** (doplněno **9 mezerami**)

```
fprintf('%15.2f\n', x);
          123.35
```

– a teď doplněno **9 nulami**

```
fprintf('%015.2f\n',x);
000000000123.35
```

3 + 1 + 2 = 6

```
fprintf('%06.2f\n',x);
123.35
```

– **nula** není doplněna, už je vypsáno **6** znaků (**2** desetinná místa, **1** tečka a **3** znaky celku)

```
fprintf('%04.2f\n',x);
123.35
```

4 - 1 - 2 = 1

– vypsáno **6** znaků (**2** desetinná místa, **1** tečka a **3** znaky celku)

– uvedené číslo říká "alespoň 4 znaky celkem", nesmí být oříznuta delší celá část.

Funkce pro vstup a výstup

Formátovaný textový výstup

`fprintf`

Příklad:

`+` značí **vždy** tisknout **znaménko** **`+`** nebo **`-`**, např.: **`%+d`**, **`%+f`**

```
c = 7; d = -34; x = 123.3456;
fprintf('%+d %+d %+.1f\n', c, d, x);
+7 -34 +123.3
```

`-` znamená **zarovnat doleva**, např.: **`%-8.3f`**

```
fprintf('%-8.3f\n', x);
123.346
fprintf('%-15.3f\n', x);
123.346
```

Funkce pro vstup a výstup

Formátovaný textový výstup

fprintf

Příklad:

+ značí **vždy** tisknout **znaménko** **+** nebo **–**, např.: **%+d**, **%+f**

```
c = 7; d = -34; x = 123.3456;
fprintf('%+d %+d %+.1f\n', c, d, x);
+7 -34 +123.3
```

- znamená **zarovnat doleva**, např.: **%-8.3f**

```
fprintf('%-8.3f\n', x);
123.346␣
fprintf('%-15.3f\n', x);
123.346␣␣␣␣␣␣␣␣
```

před nejsou mezery

Pozn. bez znaménka – doplnění **mezerami** vlevo (před číslem)

```
fprintf('%15.3f\n', x);
␣␣␣␣␣␣␣␣123.346
fprintf('%8.3f\n', x);
␣123.346
```

Funkce pro vstup a výstup

Formátovaný textový výstup

fprintf

Příklad:

```
a = 234; b = 398; c = 25;
fprintf('V desitkove soustave: %d %d %d\n', a, b, c);
V desitkove soustave: 234 398 25

fprintf('V osmickove soustave: %o %o %o\n', a, b, c);
V osmickove soustave: 352 616 31

fprintf('V 16kove soustave: %x %x %x\n', a, b, c);
V 16kove soustave: ea 18e 19

fprintf('V 16kove soustave: %X %X %X\n', a, b, c);
V 16kove soustave: EA 18E 19


fprintf('OCT %o, DEC %d\n', 25, 25);
OCT 31, DEC 25
```

Funkce pro vstup a výstup

Formátovaný textový výstup

fprintf

Příklad:

dekadická, decimální

```
a = 234; b = 398; c = 25;
fprintf('V desitkove soustave: %d %d %d\n', a, b, c);
V desitkove soustave: 234 398 25
```

oktalová

```
fprintf('V osmickove soustave: %o %o %o\n', a, b, c);
V osmickove soustave: 352 616 31

fprintf('V 16kove soustave: %x %x %x\n', a, b, c);
V 16kove soustave: ea 18e 19

fprintf('V 16kove soustave: %X %X %X\n', a, b, c);
V 16kove soustave: EA 18E 19
```

hexadecimální

```
fprintf('OCT %o, DEC %d\n', 25, 25);
OCT 31, DEC 25
```

Funkce pro vstup a výstup

Formátovaný textový výstup

fprintf

Příklad:

```
m = 123.3456;
fprintf('%f\n', m);
123.345600
fprintf('%e\n', m);
1.233456e+002
fprintf('%E\n', m);
1.23456E+002
fprintf('%g\n', m);
123.346
fprintf('%G\n', m);
123.346
```

```
n = 1.23440000e34;
fprintf('%f\n', n);
1234400000000001000000000000000000.000000
fprintf('%e\n', n);
1.234400e+034
fprintf('%E\n', n);
1.234400E+034
fprintf('%g\n', n);
1.2344e+034
fprintf('%G\n', n);
1.2344E+034
```

Funkce pro vstup a výstup

Pozn.: porovnání s **format**,který nastavuje výstupní formát zobrazení čísla na obrazovku, např.

```
y = 123.4567890123456789;
format long e
y
y = 1.2345678901234567e+002
format short e
y
y = 1.2346e+002
format long g
y
y = 123.4567890123456
format short g
y
y = 123.46
```

```
format
y
y = 123.4568
fprintf('%f\n',y)
123.456789
fprintf('%e\n',y)
1.234568e+002
fprintf('%.4e\n',y)
1.2346e+002
fprintf('%.12e\n',y)
1.234567890123e+002
fprintf('%g\n',y)
123.457
```

Příklad: funkce pro výpis podílu pomocí `disp` a `fprintf` různými způsoby, dělenec a dělitel jsou vstupními parametry funkce

```
function vystupyTest1(coDelit,cimDelit)
if(cimDelit==0)
    disp('Nulou nedelim!!!')
    return
end
x = coDelit / cimDelit;
disp(['Vysledek je: ',num2str(x)]);
hlaska = 'Vysledek je: ';
format short
disp(hlaska);
disp(x);
format long g;
disp(hlaska);
disp(x);
format rat;
disp(hlaska);
disp(x);
fprintf('\nVysledek je %.3f\n',x);
end
```

Volání funkce:

```
vystupyTest1(17,3)

Vysledek je: 5.6667
Vysledek je:
    5.6667

Vysledek je:
    5.66666666666667

Vysledek je:
      17/3

Vysledek je 5.667
```

Nebo:

```
vystupyTest1(-59,0)

Nulou nedelim!!!
```

Pokračování příkladu:

```
function vystupyTest2(coDelit,cimDelit)
if(cimDelit==0)
    disp('Nulou nedelim!!!')
    return
end
x = coDelit / cimDelit;
fprintf('Vysledek je x = %-8.3f\nA to je spravne...\n',x);
fprintf('\nNebo presneji x = %-8.6f\n', x);
fprintf('\nNyni prehledne v tabulce:\n');
fprintf('|%15s|%15s|%15s|\n','Delenec','Delitel','Podil');
fprintf('|%15.3f|%15.3f|%15.3f|\n\n', coDelit,cimDelit,x);
end
```

```
vystupyTest2(17,3)   – volání funkce
Vysledek je x = 5.667
A to je spravne...

Nebo presneji x = 5.666667

Nyni prehledne v tabulce:
|        Delenec|        Delitel|          Podil|
|         17.000|          3.000|          5.667|
```

Pokračování příkladu:

```
function vystupyTest3(coDelit,cimDelit)
if(cimDelit==0)
    disp('Nulou nedelim!!!')
    return
end

x = coDelit / cimDelit;

fprintf('\nJeste jednou, tentokrat zarovnano doleva:\n');
fprintf('|%-15s|%-15s|%-15s|\n', 'Delenec','Delitel','Podil');
fprintf('|%-15.3f|%-15.3f|%-15.3f|\n\n', coDelit,cimDelit,x);
end
```

zarovnat doleva

Volání funkce:

```
vystupyTest3(17,3)

Jeste jednou, tentokrat zarovnano doleva:
|Delenec         |Delitel         |Podil           |
|17.000          |3.000           |5.667           |
```

Pokračování příkladu:

```
function vystupyTest4(coDelit,cimDelit)
if(cimDelit==0)
    disp('Nulou nedelim!!!')
    return
end

x = coDelit / cimDelit;

fprintf('\nA oddeleno jen tabelatorem:\n');
fprintf('%15s\t%15s\t%15s\n','Delenec', 'Delitel', 'Podil');
fprintf('%15.3f\t%15.3f\t%15.3f\n\n', coDelit,cimDelit,x);
end
```

tabelátor

Volání funkce:

```
vystupyTest4(17,3)

A oddeleno jen tabelatorem:
        Delenec         Delitel           Podil
         17.000           3.000           5.667
```

Funkce pro vstup a výstup

Vstup a tisk znaků (příp. textového řetězce)

```
zadany_znak = input('Zadejte pismeno: ', 's');
Zadejte pismeno: n

fprintf('Bylo zadano: %c\n', zadany_znak);
Bylo zadano: n

zadany_text = input('Zadejte text: ', 's');
Zadejte text: Ahoj, jak je?
fprintf('Bylo zadano: %s\n', zadany_text);
Bylo zadano: Ahoj, jak je?
```

Pozn.

```
zadany_text = input('Zadejte text: ', 's');
Zadejte text: Ano!
fprintf('Bylo zadano: %c\n', zadany_text);
Bylo zadano: A
Bylo zadano: n
Bylo zadano: o
Bylo zadano: !
```

Funkce pro vstup a výstup

Vstup a tisk znaků (příp. textového řetězce)

```
zadany_znak = input('Zadejte pismeno: ', 's');
Zadejte pismeno: n

fprintf('Bylo zadano: %c\n', zadany_znak);
Bylo zadano: n

zadany_text = input('Zadejte text: ', 's');
Zadejte text: Ahoj, jak je?
fprintf('Bylo zadano: %s\n', zadany_text);
Bylo zadano: Ahoj, jak je?
```

řetězec

Pozn.

```
zadany_text = input('Zadejte text: ', 's');
Zadejte text: Ano!
fprintf('Bylo zadano: %c\n', zadany_text);
Bylo zadano: A
Bylo zadano: n
Bylo zadano: o
Bylo zadano: !
```

znak

Funkce pro vstup a výstup

Pozn.: k zadávání znaků
Pokud uživatel zadá více znaků, lze se přebytečných zbavit např. takto:

```
zadany_znak = input('Zadejte pismeno: ', 's');
Zadejte pismeno: Ahoj, jak je?

if (length(zadany_znak) > 1)
zadany_znak = zadany_znak(1,1);
end;

fprintf('Bylo zadano: %c\n', zadany_znak);
Bylo zadano: A
```

Funkce pro vstup a výstup

Pozn.: k zadávání znaků
Pokud uživatel zadá více znaků, lze se přebytečných zbavit např. takto:

```
zadany_znak = input('Zadejte pismeno: ', 's');
Zadejte pismeno: Ahoj, jak je?

if (length(zadany_znak) > 1)
zadany_znak = zadany_znak(1,1);
end;

fprintf('Bylo zadano: %c\n', zadany_znak);
Bylo zadano: A
```

Funkce pro vstup a výstup

Pozn.: k zadávání znaků
Pokud uživatel zadá více znaků, lze se přebytečných zbavit např. takto:

```
zadany_znak = input('Zadejte pismeno: ', 's');
Zadejte pismeno: Ahoj, jak je?

if (length(zadany_znak) > 1)
zadany_znak = zadany_znak(1,1);
end;

fprintf('Bylo zadano: %c\n', zadany_znak);
Bylo zadano: A

fprintf('Bylo zadano: %5c\n', zadany_znak);
Bylo zadano:     A
```

Funkce pro vstup a výstup

Pozn.: k zadávání znaků
Pokud uživatel zadá více znaků, lze se přebytečných zbavit např. takto:

```
zadany_znak = input('Zadejte pismeno: ', 's');
Zadejte pismeno: Ahoj, jak je?

if (length(zadany_znak) > 1)
zadany_znak = zadany_znak(1,1);
end;

fprintf('Bylo zadano: %c\n', zadany_znak);
Bylo zadano: A
```

Doplněno 4 mezerami,
4 +1= 5

```
fprintf('Bylo zadano: %5c\n', zadany_znak);
Bylo zadano:     A
```

Funkce pro vstup a výstup

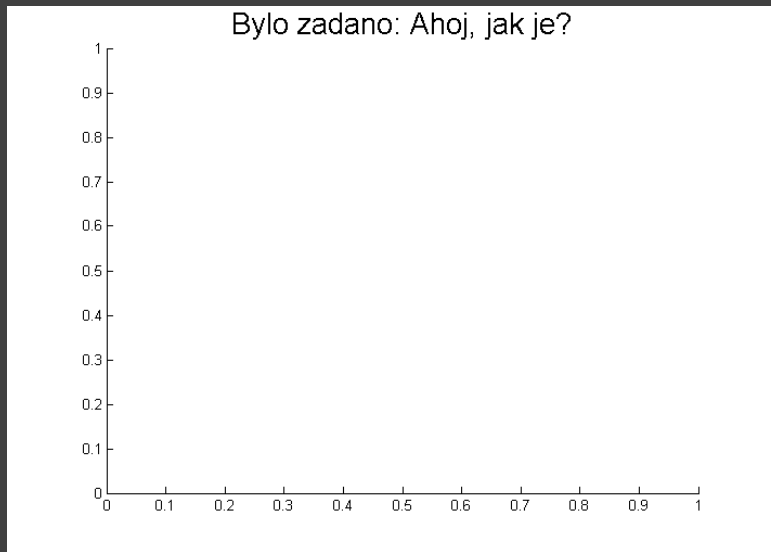
Tisk do řetězce

```
r = sprintf('formatovací sekvence', argumenty)
```

– vrací řetězec, do řetězce **r** jsou vypsány argumenty v požadovaném formátu v uvedeném pořadí, stejně jako u **fprintf**, formátovací sekvence jsou také stejné jako u **fprintf**.

Příklady:

```
zadany_text = input('Zadejte text: ', 's');
Zadejte text: Ahoj, jak je?
hlaska = sprintf('Bylo zadano: %s\n', zadany_text);
figure
% prázdné grafické okno
title(hlaska)
% titulek grafu z řetězce hlaska
```



Funkce pro vstup a výstup

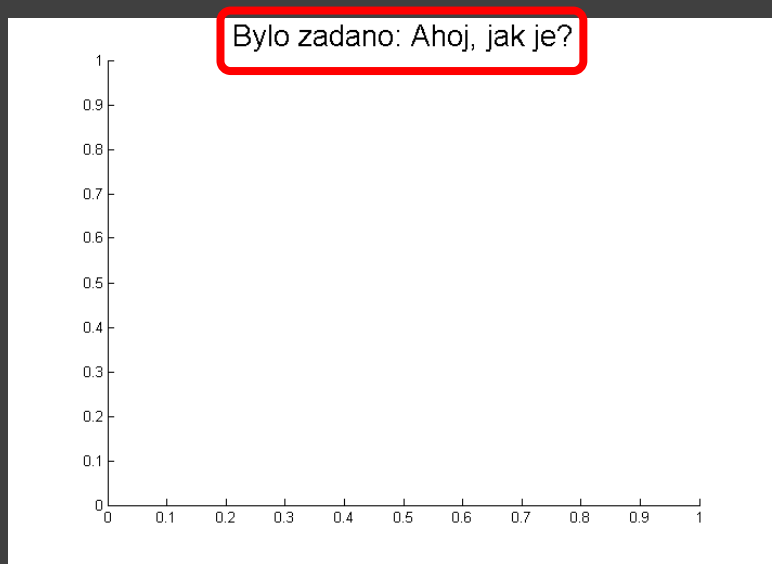
Tisk do řetězce

`r = sprintf('formatovací sekvence', argumenty)`

– vrací řetězec, do řetězce **`r`** jsou vypsány argumenty v požadovaném formátu v uvedeném pořadí, stejně jako u **`fprintf`**, formátovací sekvence jsou také stejné jako u **`fprintf`**.

Příklady:

```
zadany_text = input('Zadejte text: ', 's');
Zadejte text: Ahoj, jak je?
hlaska = sprintf('Bylo zadano: %s\n', zadany_text);
figure
% prázdné grafické okno
title(hlaska)
% titulek grafu z řetězce hlaska
```



Funkce pro vstup a výstup

Tisk do řetězce

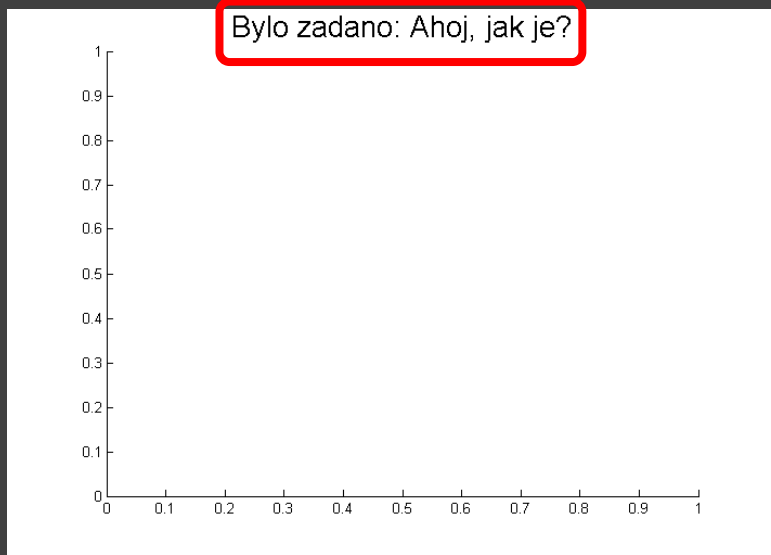
`r = sprintf('formatovací sekvence', argumenty)`

– vrací řetězec, do řetězce **`r`** jsou vypsány argumenty v požadovaném formátu v uvedeném pořadí, stejně jako u **`fprintf`**, formátovací sekvence jsou také stejné jako u **`fprintf`**.

Příklady:

```
zadany_text = input('Zadejte text: ', 's');
Zadejte text: Ahoj, jak je?
hlaska = sprintf('Bylo zadano: %s\n', zadany_text);
figure
% prázdné grafické okno
title(hlaska)
% titulek grafu z řetězce hlaska

disp(hlaska)
Bylo zadano: Ahoj, jak je?
fprintf('%s\n',hlaska)
Bylo zadano: Ahoj, jak je?
```



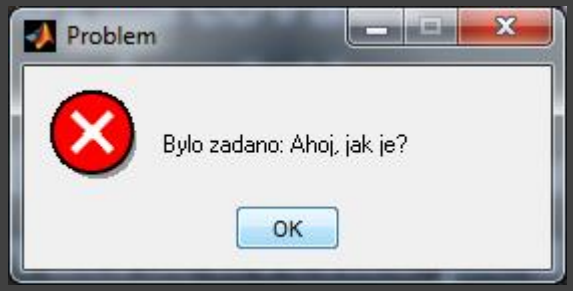
Funkce pro vstup a výstup

Tisk do řetězce

sprintf

Příklady:

```
zadany_text = input('Zadejte text: ', 's');
Zadejte text: Ahoj, jak je?
hlaska = sprintf('Bylo zadano: %s\n', zadany_text);
msgbox(hlaska,'Problem','error');
```



msgbox(zpráva,titulek,ikona) - vytvoří okno se **zprávou**, **titulek** se objeví v liště, **ikona** určuje, jaký obrázek (**'error'**, **'help'**, **'warn'**) se zobrazí v okně se zprávou.

Pozn.: další okna viz help - pod heslem Predefined Dialog Boxes

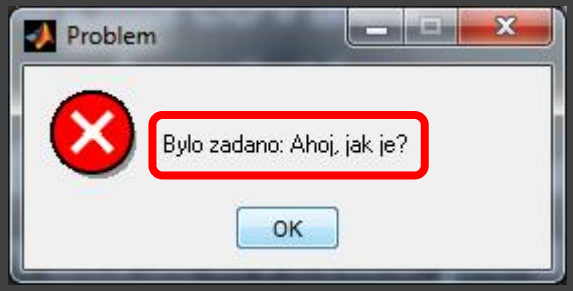
Funkce pro vstup a výstup

Tisk do řetězce

sprintf

Příklady:

```
zadany_text = input('Zadejte text: ', 's');
Zadejte text: Ahoj, jak je?
hlaska = sprintf('Bylo zadano: %s\n', zadany_text);
msgbox(hlaska,'Problem','error');
```



msgbox(zpráva,titulek,ikona) - vytvoří okno se **zprávou**, **titulek** se objeví v liště, **ikona** určuje, jaký obrázek (**'error'**, **'help'**, **'warn'**) se zobrazí v okně se zprávou.

Pozn.: další okna viz help - pod heslem Predefined Dialog Boxes

Funkce pro vstup a výstup

Příklad: Vytvořte uživatelskou funkci **s2cos_graf_s_titulkem** bez parametrů. Tato funkce **s2cos_graf_s_titulkem** bude vykreslovat graf křivky dané rovnicí

$$y = \sin^2(x)\cos(x)$$

pro x z intervalu, jehož jednu mez zadá uživatel z klávesnice, druhá mez je 4π. Je třeba ošetřit, aby první mez nebyla stejná nebo přibližně stejná jako druhá, aby rozdíl mezi nimi byl větší nebo roven 0,025.

Pro výpočet $y = \sin^2(x)\cos(x)$ bude vytvořena další funkce **s2cos** s jedním vstupním parametrem x a jedním výstupním parametrem y. **s2cos** bude volána ve **s2cos_graf_s_titulkem**.

Funkce mohou být napsány každá v samostatném souboru a uloženy ve stejné složce nebo mohou být napsány v jednom souboru (nejprve hlavní funkce **s2cos_graf_s_titulkem**, a potom funkce **s2cos**). Nejdříve vytvoříme funkci **s2cos_graf_s_titulkem**:

```
function s2cos_graf_s_titulkem
horni = 4*pi;
dolni = input('Zadejte pocatecni hodnotu: ');
if ((horni == dolni)||(abs(horni-dolni) < 0.025))
 hlaska = sprintf('Dolni mez %.2f je stejná s horni!',dolni);
 msgbox(hlaska,'Problem','error');
 return;
end
osa_x = linspace(dolni,horni,abs(horni-dolni)*100);
osa_y = s2cos(osa_x);
plot(osa_x, osa_y)
xlabel('x')
ylabel('y')
titulek = sprintf('Graf y = sin(x)^2 cos(x) v mezich od %.2f do
 %.2f\n', dolni, horni);
title(titulek)
end
```

```
function y = s2cos(x)
y = (sin(x).^ 2).* cos(x);
end
```

```
function s2cos_graf_s_titulkem
horni = 4*pi;
dolni = input('Zadejte pocatecni hodnotu: ');
if ((horni == dolni)||(abs(horni-dolni) < 0.025))
 hlaska = sprintf('Dolni mez %.2f je stejná s Horní!',dolni);
 msgbox(hlaska,'Problem','error');
 return;
end
osa_x = linspace(dolni,horni,abs(horni-dolni)*100);
osa_y = s2cos(osa_x);
plot(osa_x, osa_y)
xlabel('x')
ylabel('y')
titulek = sprintf('Graf y = sin(x)^2 cos(x) v mezich od %.2f do
 %.2f\n', dolni, horni);
title(titulek)
end
```

velikost, absolutní hodnota

je požadován rozdíl mezi mezemi >= 0,025

volání funkce s2cos se vstupem osa_x pro výpočet *y*

```
function y = s2cos(x)
y = (sin(x).^ 2).* cos(x);
end
```

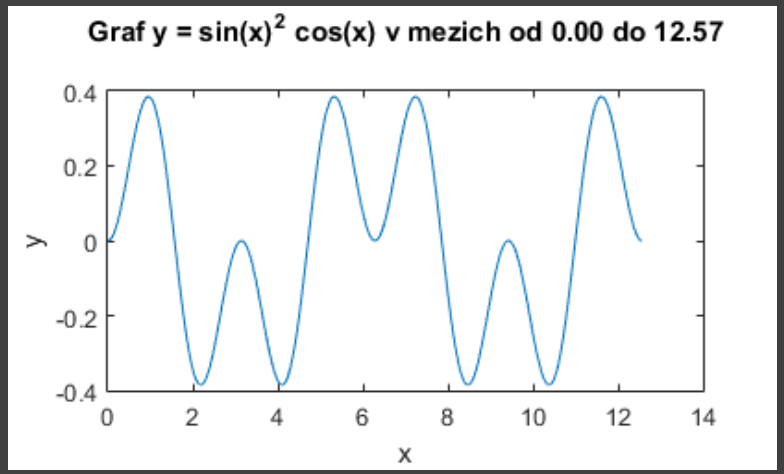
Funkce pro vstup a výstup

Pokračování příkladu:

Volání funkce jejím názvem:

```
s2cos_graf_s_titulkem
Zadejte pocatecni hodnotu: 0
```

a už se vykreslí graf
s příslušným titulkem ...



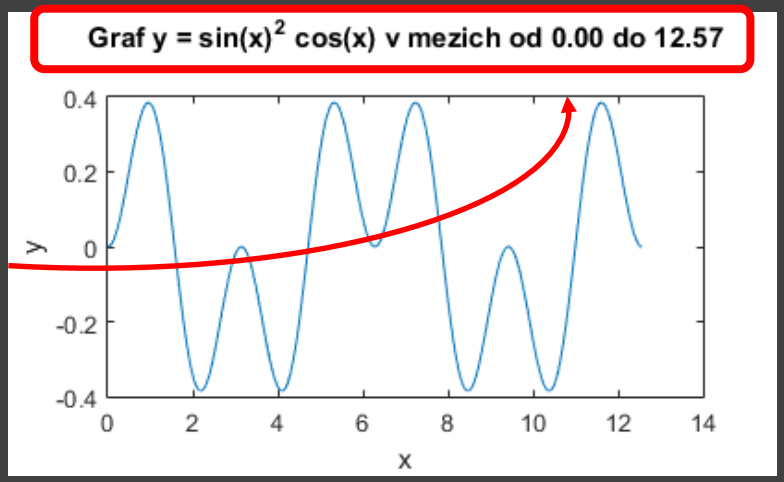
Funkce pro vstup a výstup

Pokračování příkladu:

Volání funkce jejím názvem:

```
s2cos_graf_s_titulkem
Zadejte pocatecni hodnotu: 0
```

a už se vykreslí graf
s příslušným titulkem ...



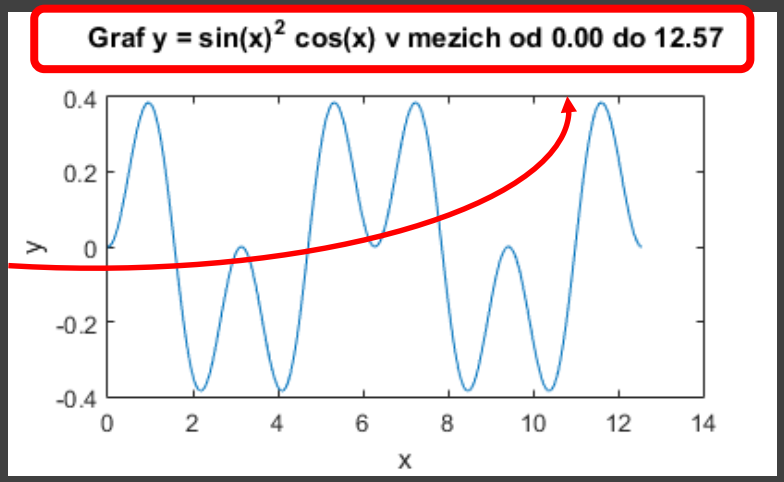
Funkce pro vstup a výstup

Pokračování příkladu:

Volání funkce jejím názvem:

```
s2cos_graf_s_titulkem
Zadejte pocatecni hodnotu: 0
```

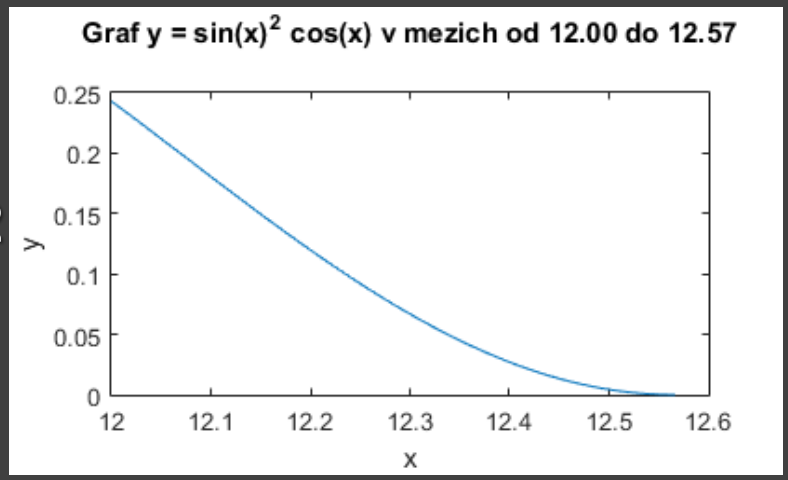
a už se vykreslí graf
s příslušným titulkem ...



nebo:

```
s2cos_graf_s_titulkem
Zadejte pocatecni hodnotu: 12
```

a už se vykreslí graf
s příslušným titulkem ...



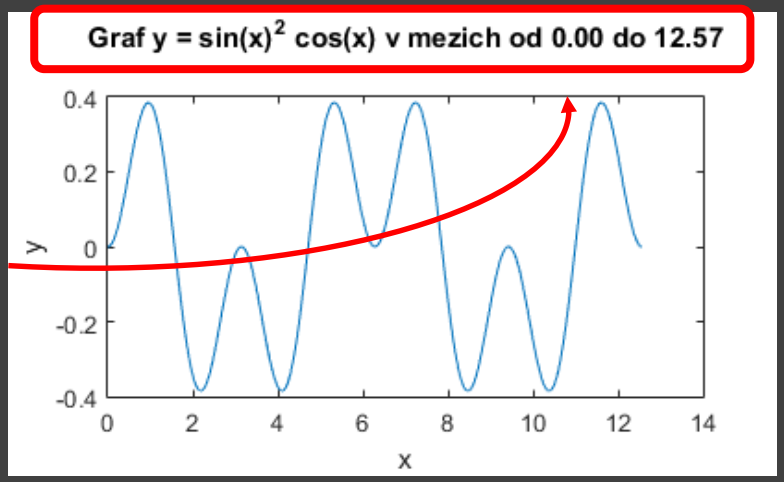
Funkce pro vstup a výstup

Pokračování příkladu:

Volání funkce jejím názvem:

```
s2cos_graf_s_titulkem
Zadejte pocatecni hodnotu: 0
```

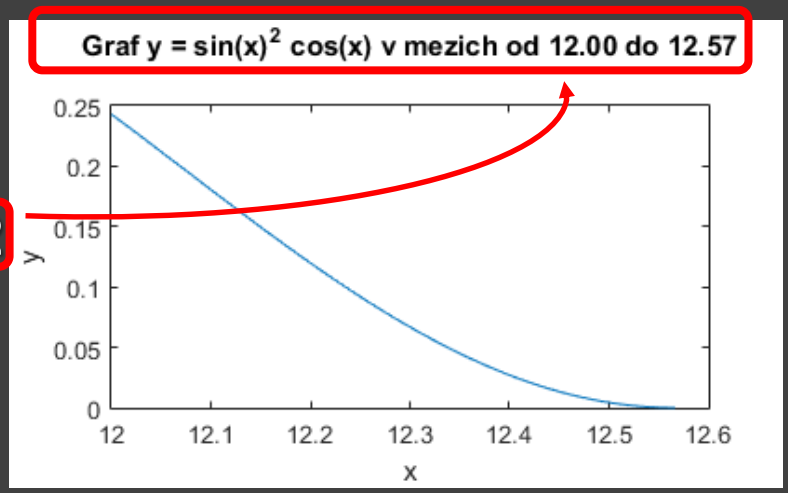
a už se vykreslí graf
s příslušným titulkem ...



nebo:

```
s2cos_graf_s_titulkem
Zadejte pocatecni hodnotu: 12
```

a už se vykreslí graf
s příslušným titulkem ...



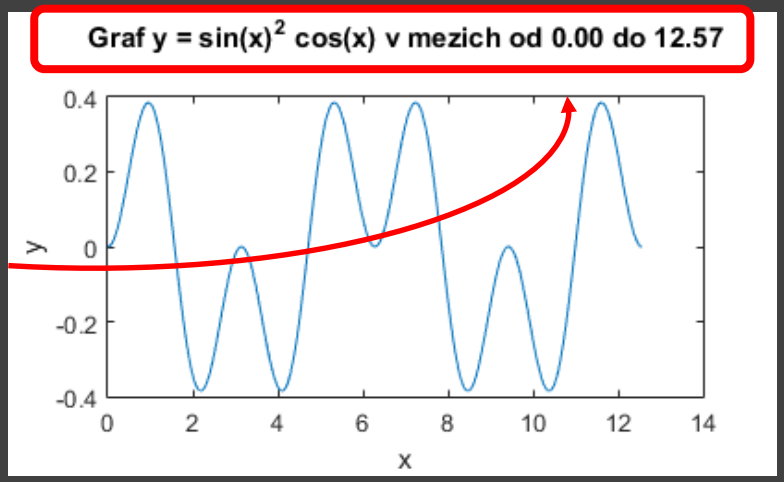
Funkce pro vstup a výstup

Pokračování příkladu:

Volání funkce jejím názvem:

```
s2cos_graf_s_titulkem
Zadejte pocatecni hodnotu: 0
```

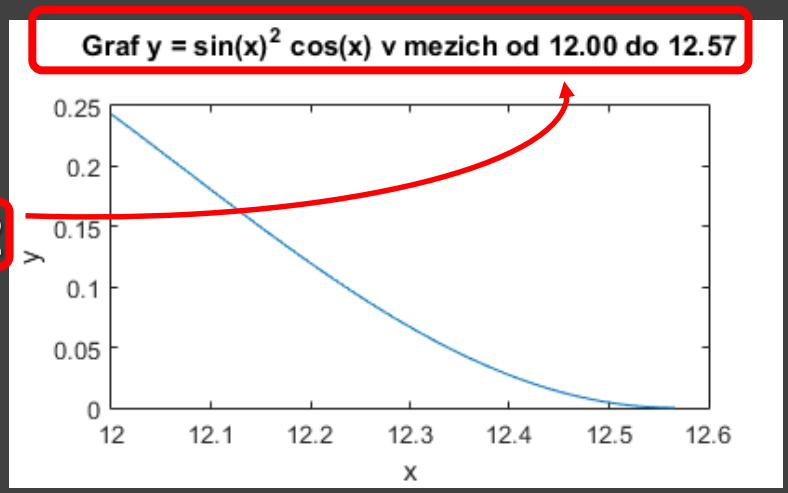
a už se vykreslí graf
s příslušným titulkem ...



nebo:

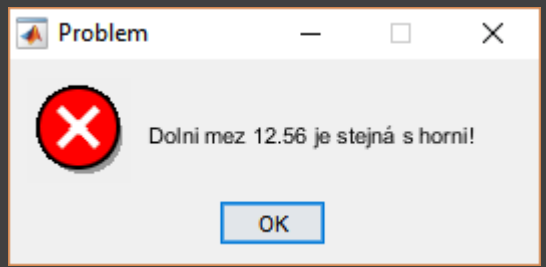
```
s2cos_graf_s_titulkem
Zadejte pocatecni hodnotu: 12
```

a už se vykreslí graf
s příslušným titulkem ...



nebo:

```
s2cos_graf_s_titulkem
Zadejte pocatecni hodnotu:4*pi-0.01
```



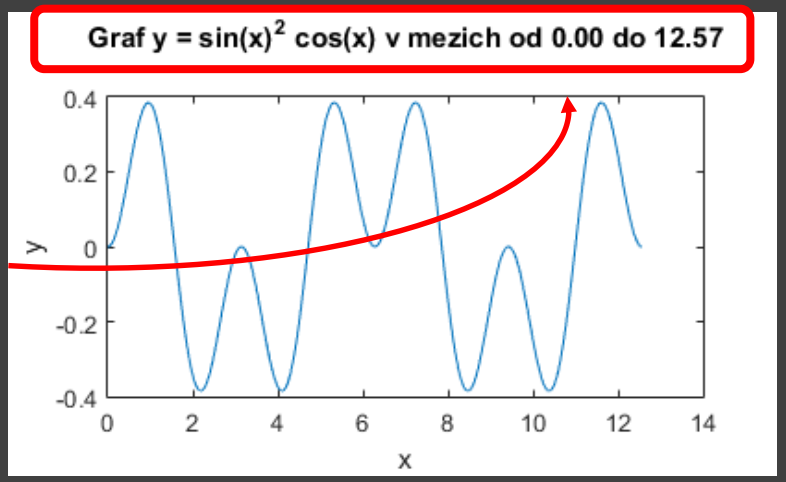
Funkce pro vstup a výstup

Pokračování příkladu:

Volání funkce jejím názvem:

```
s2cos_graf_s_titulkem
Zadejte pocatecni hodnotu: 0
```

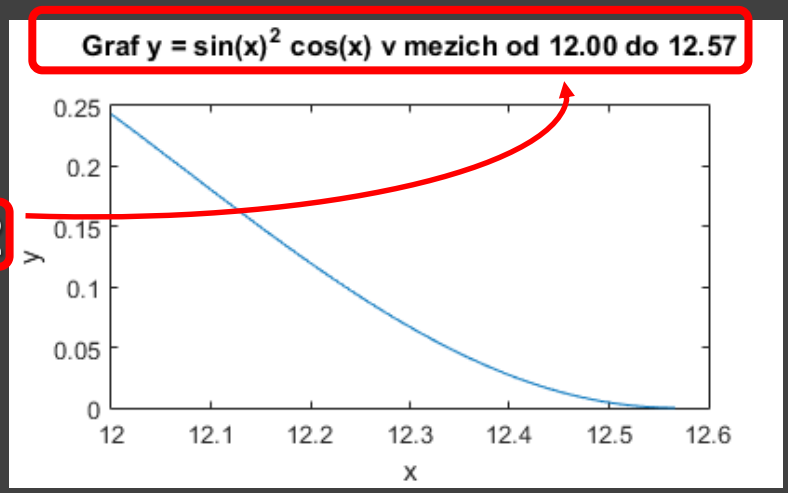
a už se vykreslí graf
s příslušným titulkem ...



nebo:

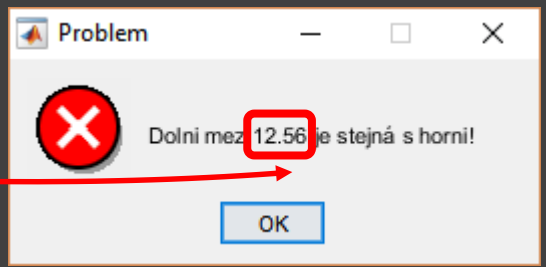
```
s2cos_graf_s_titulkem
Zadejte pocatecni hodnotu: 12
```

a už se vykreslí graf
s příslušným titulkem ...



nebo:

```
s2cos_graf_s_titulkem
Zadejte pocatecni hodnotu: 4*pi-0.01
```



Funkce pro vstup a výstup

Příklad:
Funkce – jednoduchá kalkulačka

Uživatel zadá dvě čísla a bude vybírat z operací:
`+ ,- ,* ,/ ,\ .`

Pokud uživatel zadá matici nebo vektor, jako zadané číslo je uvažován prvek v prvním řádku a prvním sloupci.

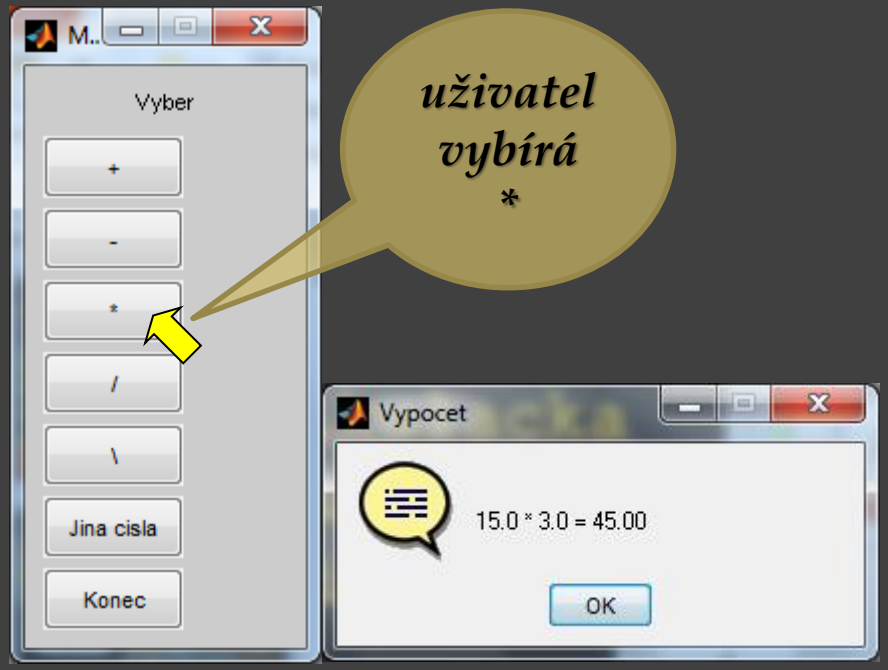
Uživateli bude umožněno po skočení výpočtu provádět další výpočet nebo zadat jiná čísla a opět pokračovat ve výpočtech.

```
function kalkulacka
while(1)
  a = input('Zadej prvni cislo:'); if (length(a)>1) a = a(1,1); end
  b = input('Zadej druhe cislo:'); if (length(b)>1) b = b(1,1); end
  while(1)
    znak = menu('Vyber','+','-','*','/','\','Jina cisla','Konec');
    switch (znak)
      case 1
        s = sprintf('%.1f + %.1f = %.2f',a,b,a+b);
      case 2
        s = sprintf('%.1f - %.1f = %.2f',a,b,a-b);
      case 3
        s = sprintf('%.1f * %.1f = %.2f',a,b,a.*b);
      case 4
        s = sprintf('%.1f / %.1f = %.2f',a,b,a./b);
      case 5
        s = sprintf('%.1f \\ %.1f = %.2f',a,b,a.\b);
      case 6
        msgbox('Zadej jina cisla','Informace','warn');  break;
      case 7
        msgbox('Konec','Informace','warn'); return;
    end
    msgbox(s,'Vypocet','help');
  end
end
end
```

Funkce pro vstup a výstup

Pokračování příkladu:
Volání funkce **kalkulacka**

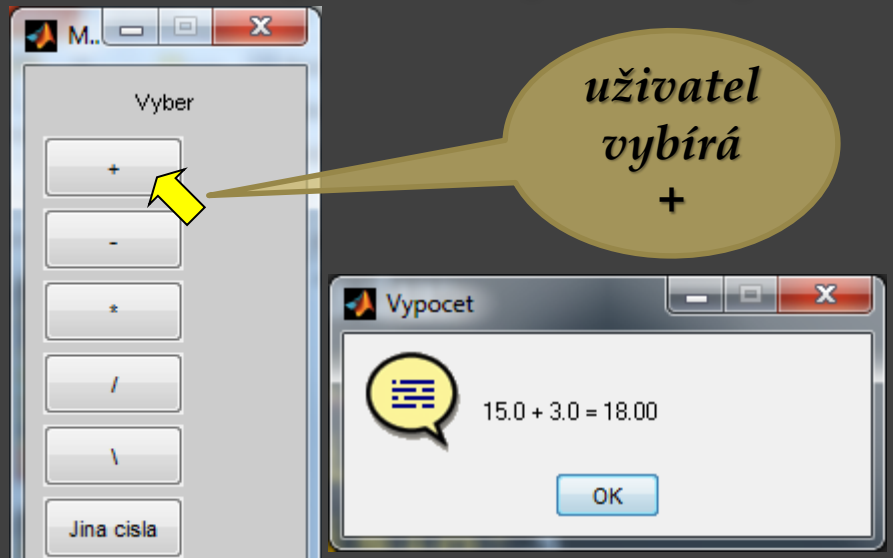
```
Zadej prvni cislo:15
Zadej druhe cislo:3
```

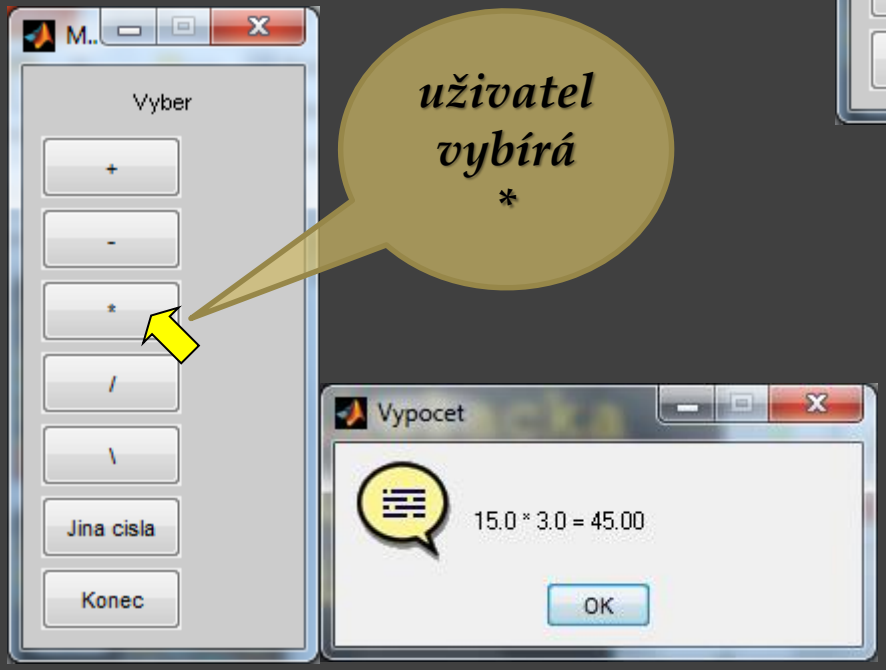


Funkce pro vstup a výstup

Pokračování příkladu:
Volání funkce **kalkulacka**

```
Zadej prvni cislo:15
Zadej druhe cislo:3
```

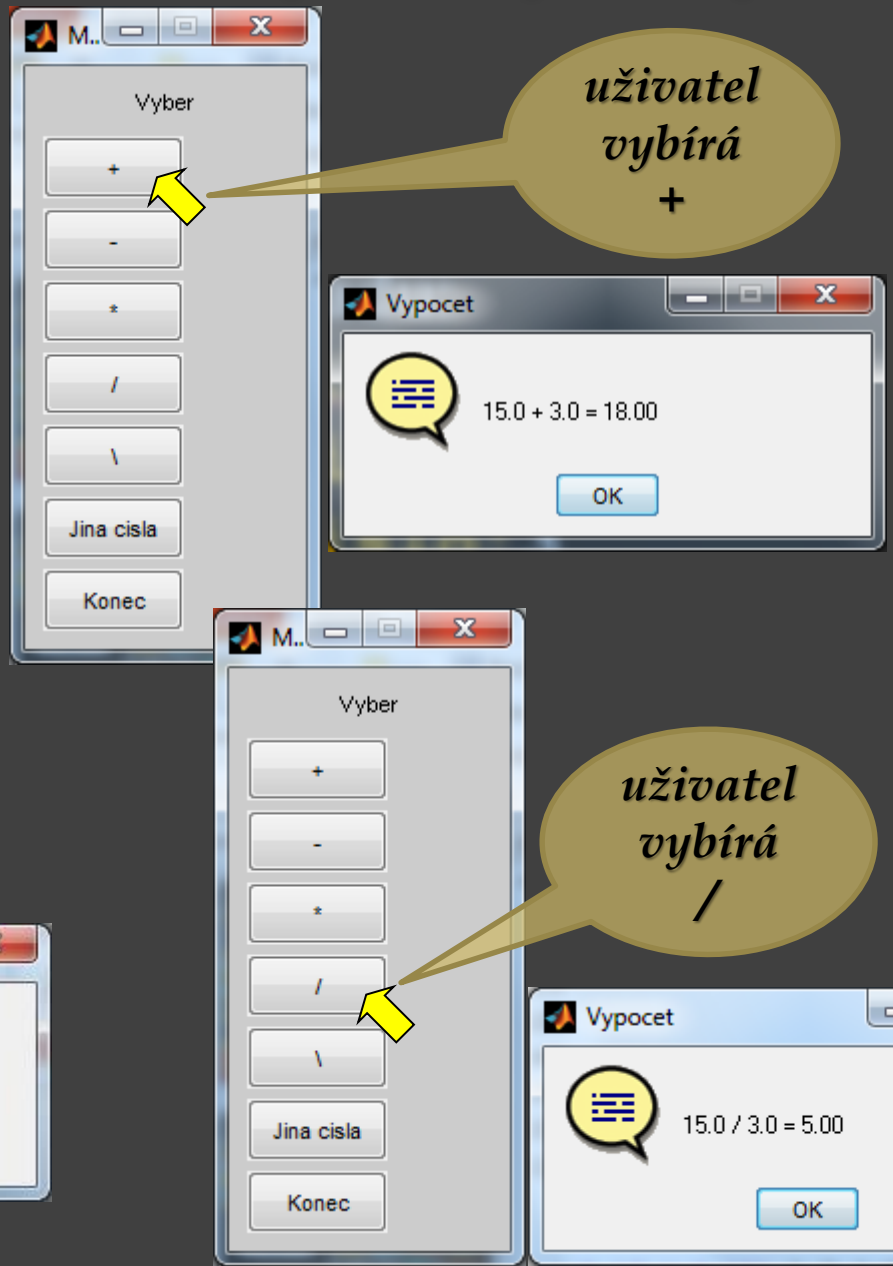




Funkce pro vstup a výstup

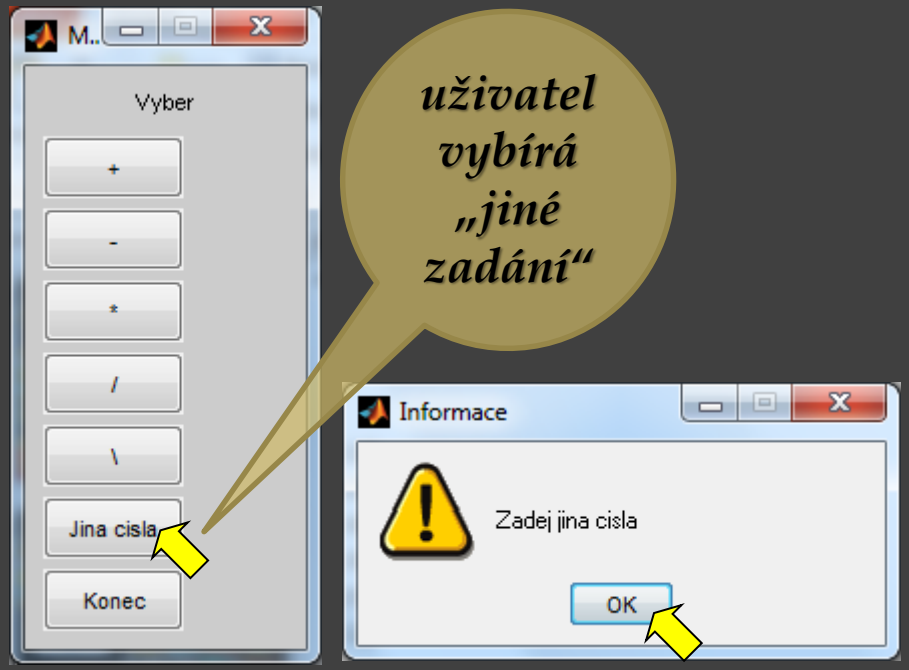
Pokračování příkladu:
Volání funkce **kalkulacka**

```
Zadej prvni cislo:15
Zadej druhe cislo:3
```



Funkce pro vstup a výstup

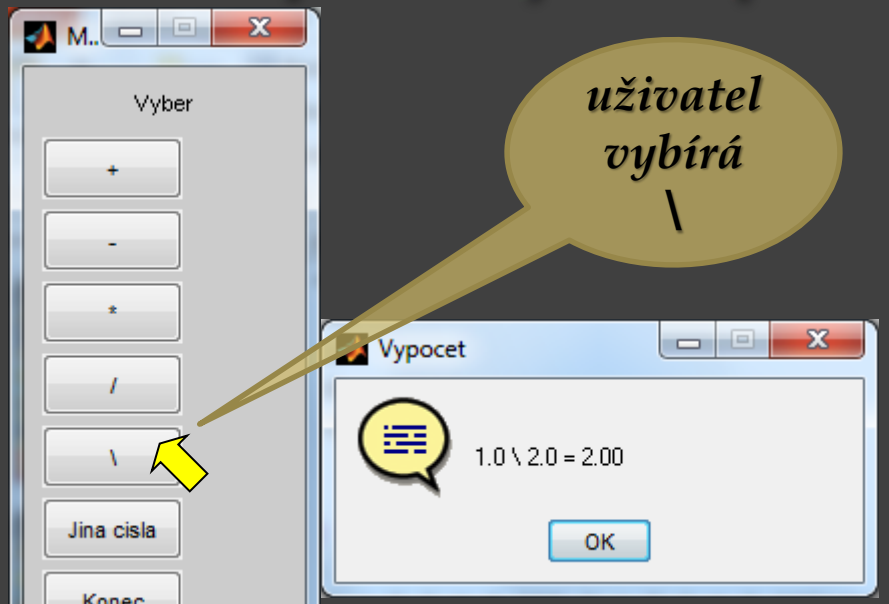
Pokračování příkladu:
Volání funkce **`kalkulacka`**

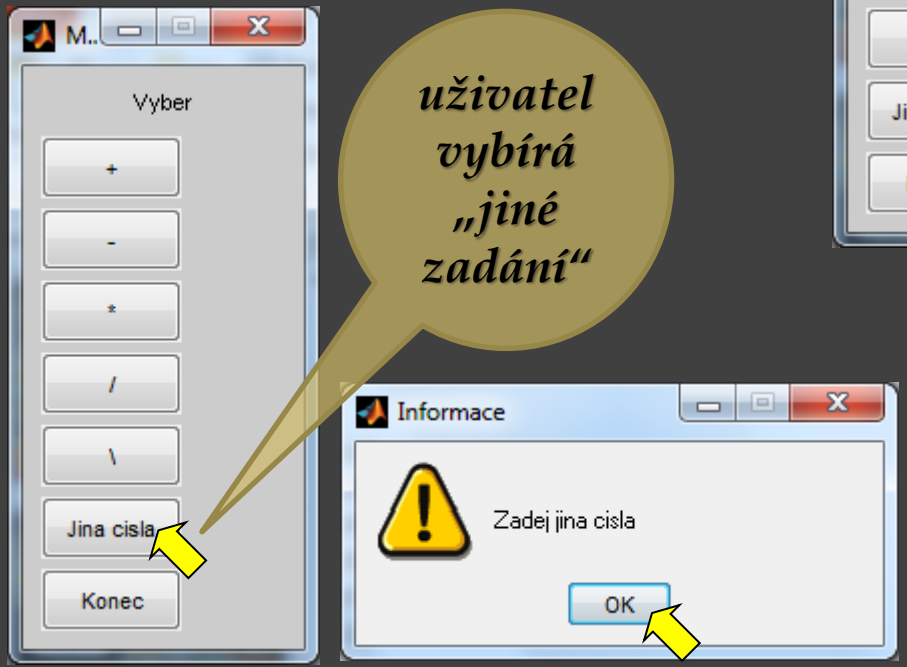



```
Zadej prvni cislo:1
Zadej druhe cislo:2
```

Funkce pro vstup a výstup

Pokračování příkladu:
Volání funkce **`kalkulacka`**

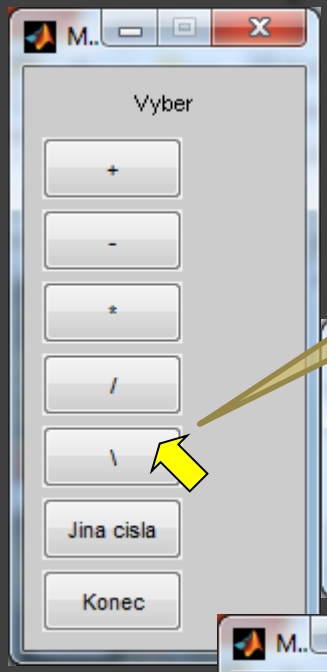




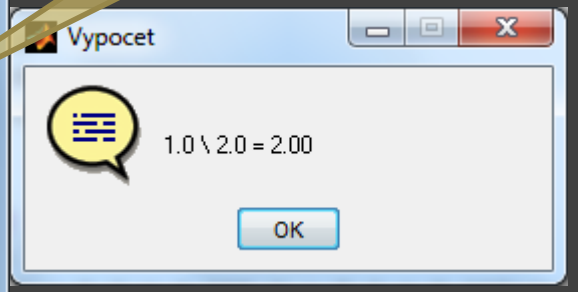

```
Zadej prvni cislo:1
Zadej druhe cislo:2
```

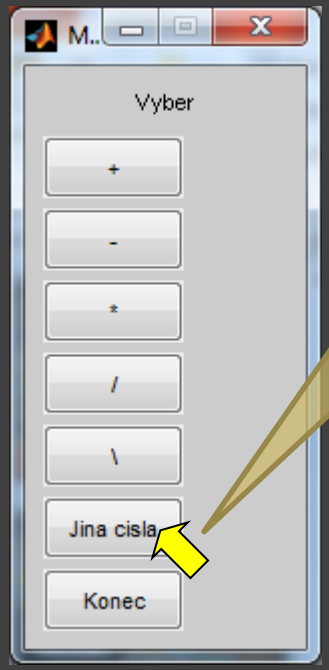
Funkce pro vstup a výstup

Pokračování příkladu:
Volání funkce **`kalkulacka`**

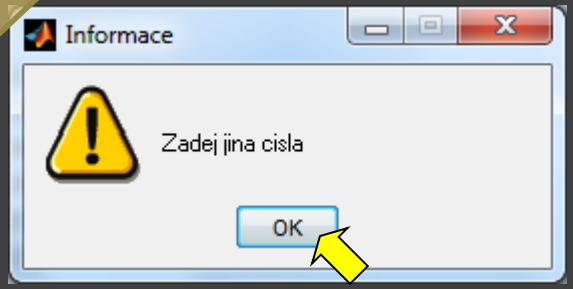


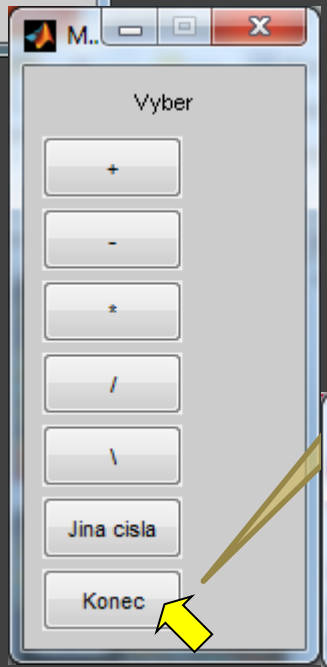
*uživatel vybírá *



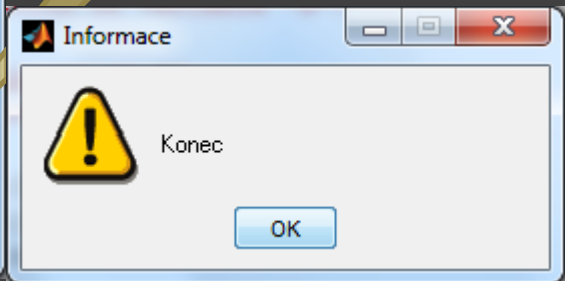


uživatel vybírá „jiné zadání“





uživatel vybírá „Konec“




```
Zadej prvni cislo:1
Zadej druhe cislo:2
```

Základy práce se soubory

```
f = fopen('nazev_soubor', 'režim');
```

fopen otevře soubor s názvem **nazev_soubor** pro přístup pro čtení nebo zápis podle zvoleného **režimu**. Řetězec **nazev_soubor** obsahuje název souboru, který má být otevřen. Pokud fopen nemůže otevřít soubor, vrátí hodnotu -1. Jetliže soubor není v aktuální složce, lze zadat i cestu k souboru, např.:

```
f = fopen('C:\\cesta_k_souboru\\soubor', 'režim');
```

kde **režim** může být:

- `'r'` – číst (read)
- `'w'` – zápis, přepis, vytvoření nového souboru (write)
- `'a'` – připsat na konec existujícího (append)
- `'r+'` – čtení nebo zápis
- `'w+'` – čtení nebo zápis, přepis, vytvoření nového souboru
- `'a+'` – čtení nebo zápis, přepis, vytvoření nového souboru a přidávat na konec souboru další data

Základy práce se soubory

Zápis do textového souboru:
`fprintf()` – jako na obrazovku, jediný rozdíl je, že je nutné uvést proměnnou, do které je otevřen soubor, např.

```
muj_soubor = fopen('C:\\cesta\\soubor.txt','w');

fprintf(muj_soubor,'format.sekvence',argumenty);
```

formátovací sekvence a argumenty používáme stejné jako při výpisu na obrazovku

Použitý soubor je třeba nakonec zavřít pomocí **`fclose()`**, jež vrací 0 v případě úspěšného zavření souboru nebo -1 v případě neúspěchu .

```
fclose(muj_soubor);
```

Pozn.: test konce souboru
`feof(muj_soubor)` – pokud je konec souboru, vrátí 1

Základy práce se soubory

'w' – zápis (write)

Zápis do textového souboru:
`fprintf()` – jako na obrazovku, jediný rozdíl je, že je nutné uvést proměnnou, do které je otevřen soubor, např.

```
muj_soubor = fopen('C:\\cesta\\soubor.txt','w');
```

```
fprintf(muj_soubor,'format.sekvence',argumenty);
```

formátovací sekvence a argumenty používáme stejné jako při výpisu na obrazovku

Použitý soubor je třeba nakonec zavřít pomocí `fclose()`, jež vrací 0 v případě úspěšného zavření souboru nebo -1 v případě neúspěchu .

```
fclose(muj_soubor);
```

Pozn.: test konce souboru
`feof(muj_soubor)` – pokud je konec souboru, vrátí 1

Základy práce se soubory

Příklad: Funkce, v níž do textového souboru uloženého v aktuální složce bude zapsán **text**, **znak** a **číslo**, které zadá uživatel z klávesnice.

```
function soubor_ulozeni
f = fopen('soubor.txt', 'w');
zadany_text = input('Zadejte text: ', 's');
zadany_znak = input('Zadejte znak: ', 's');
cislo = input('Zadejte cislo: ');
fprintf(f,'Bylo zadan text: %s\n', zadany_text);
fprintf(f,'Bylo zadan znak: %c\n', zadany_znak);
fprintf(f,'Bylo zadano cislo: %g\n', cislo);
fclose(f);
end
```

Základy práce se soubory

Příklad: Funkce, v níž do textového souboru uloženého v aktuální složce bude zapsán **text**, **znak** a **číslo**, které zadá uživatel z klávesnice.

'w' – zápis (write)

```
function soubor_ulozeni
f = fopen('soubor.txt', 'w');
zadany_text = input('Zadejte text: ', 's');
zadany_znak = input('Zadejte znak: ', 's');
cislo = input('Zadejte cislo: ');
fprintf(f,'Bylo zadan text: %s\n', zadany_text);
fprintf(f,'Bylo zadan znak: %c\n', zadany_znak);
fprintf(f,'Bylo zadano cislo: %g\n', cislo);
fclose(f);
end
```

Základy práce se soubory

Pokračování příkladu:
Volání funkce `soubor_ulozeni`:

```
Zadejte text: Ahoj!
Zadejte znak: x
Zadejte cislo: 12345
```

V textovém souboru `soubor.txt`, který je uložen v aktuálním adresáři, je zapsáno:

```
Bylo zadan text: Ahoj!
Bylo zadan znak: x
Bylo zadano cislo: 12345
```

Základy práce se soubory

Příklad:

```
function soubor_cisla
p = input('Zadej, kolik chces ulozit cisel? ');
if(p > 0)
    f = fopen('data.txt','w');
    for n = 1:p
        fprintf(f,'%g\n',rand);
    end
    fclose(f);
    fprintf('Cisla byla ulozena\n');
else
    fprintf('Chybne zadani\n');
end
end
```

Základy práce se soubory

Pokračování příkladu:
Volání funkce **soubor_cisla**:

```
Zadej, kolik chces ulozit cisel? -5
Chybne zadani
```

nebo

```
Zadej, kolik chces ulozit cisel? 20
Cisla byla ulozena
```

V textovém souboru **data.txt**, který je uložen v aktuální složce, je pak zapsáno např.:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

Základy práce se soubory

fscanf – čtení z textového souboru uloženého na disku, soubor musí nejprve být otevřen pomocí fopen

Např.

Načtení všech čísel ze souboru **data.txt** uloženého na disku C

```
soubor = fopen('C:\\data.txt', 'r');

cisla = fscanf(soubor, '%g');
fclose(soubor);
```

'r' – číst (**r**ead)

Načtení jednoho čísla ze souboru **data.txt** uloženého na disku D (**1 řádek**, **1 sloupec**)

```
soubor = fopen('D:\\data.txt', 'r');
cisla = fscanf(soubor, '%g', [1,1]);
fclose(soubor);
```

Velikost je nepovinná, ale pomocí ní lze omezit počet prvků, které je možné číst ze souboru.

Základy práce se soubory

Velikost je tedy nepovinná, ale pomocí ní lze omezit počet prvků, které je možné číst ze souboru, je-li zadán rozměr matice, vyplní se matice o dané velikosti, např.

Čtení z textového souboru `soubor.txt` uloženého v aktuálním adresáři:

```
muj_soubor = fopen('soubor.txt', 'r');

cisla = fscanf(muj_soubor, '%g', [2,inf]);
```

fscanf má přečíst reálná čísla, organizovaná ve **2 řádcích** a **neznámém počtu sloupců** (chci **číst** až **do konce souboru**), na pozici řádek **nesmí** být `inf`.

Použitý soubor je třeba nakonec zavřít:

```
fclose(muj_soubor)
```

Základy práce se soubory

Příklad: čtení z textového souboru `soubor.txt`

```
f1=fopen('soubor.txt','r');
t = fscanf(f1,'%c');
fclose(f1);

disp(t)
Bylo zadan text: Ahoj!
Bylo zadano znak: x
Bylo zadano cislo: 12345
```

V textovém souboru `soubor.txt`, který je uložen v aktuálním adresáři, je zapsáno:

```
Bylo zadan text: Ahoj!
Bylo zadan znak: x
Bylo zadano cislo: 12345
```

Základy práce se soubory

Příklad: čtení z textového souboru `soubor.txt`

```
f1=fopen('soubor.txt','r');
t = fscanf(f1,'%c');
fclose(f1);
```

`'r'` – číst (read)

`'%c'` – znak

```
disp(t)
Bylo zadan text: Ahoj!
Bylo zadano znak: x
Bylo zadano cislo: 12345
```

V textovém souboru `soubor.txt`, který je uložen v aktuálním adresáři, je zapsáno:

```
Bylo zadan text: Ahoj!
Bylo zadan znak: x
Bylo zadano cislo: 12345
```

Základy práce se soubory

Příklad: čtení z textového souboru **soubor.txt**

```
f1=fopen('soubor.txt','r');
t = fscanf(f1,'%c');
fclose(f1);
```

'r' – číst (read)

'%c' – znak

V textovém souboru **soubor.txt**, který je uložen v aktuálním adresáři, je zapsáno:

```
Bylo zadan text: Ahoj!
Bylo zadan znak: x
Bylo zadano cislo: 12345
```

```
disp(t)
Bylo zadan text: Ahoj!
Bylo zadano znak: x
Bylo zadano cislo: 12345

f2=fopen('soubor.txt','r');
s = fscanf(f2,'%s');
fclose(f2);

disp(s)
Bylozadantext:Ahoj!Bylozadanoznak:xBylozadanocislo:12345
```

Základy práce se soubory

Příklad: čtení z textového souboru **soubor.txt**

```
f1=fopen('soubor.txt','r');
t = fscanf(f1,'%c');
fclose(f1);

disp(t)
Bylo zadan text: Ahoj!
Bylo zadano znak: x
Bylo zadano cislo: 12345

f2=fopen('soubor.txt','r');
s = fscanf(f2,'%s');
fclose(f2);

disp(s)
Bylozadantext:Ahoj!Bylozadanoznak:xBylozadanocislo:12345
```

'r' – číst (read)

'%c' – znak

V textovém souboru **soubor.txt**, který je uložen v aktuálním adresáři, je zapsáno:

```
Bylo zadan text: Ahoj!
Bylo zadan znak: x
Bylo zadano cislo: 12345
```

'r' – číst (read)

'%s' – řetězec

Bez mezer, bez konců řádek

Základy práce se soubory

Příklad: čtení z textového souboru `data.txt`

```
fr=fopen('data.txt','r');
c = fscanf(fr,'%g');
fclose(fr);
```

```
disp(c)
```

```
0.1067
0.9619
0.0046
0.7749
0.8173
0.8687
0.0844
0.3998
0.2599
0.8001
0.4314
0.9106
0.1818
0.2638
0.1455
0.1361
0.8693
0.5797
0.5499
0.1450
```

V textovém souboru `data.txt`, který je uložen v aktuálním adresáři, je zapsáno:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

Základy práce se soubory

Příklad: čtení z textového souboru `data.txt`

```
fr=fopen('data.txt','r');
c = fscanf(fr,'%g');
fclose(fr);
```

`'r'` – číst (read)

`'%g'` – destinné číslo
příp. i v exp. tvaru, bez nevýznamných nul

```
disp(c)
```

```
0.1067
0.9619
0.0046
0.7749
0.8173
0.8687
0.0844
0.3998
0.2599
0.8001
0.4314
0.9106
0.1818
0.2638
0.1455
0.1361
0.8693
0.5797
0.5499
0.1450
```

V textovém souboru `data.txt`, který je uložen v aktuálním adresáři, je zapsáno:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

Základy práce se soubory

Příklad: čtení z textového souboru **data.txt**

```
fr=fopen('data.txt','r');
c = fscanf(fr,'%g',[4,Inf]);
fclose(fr);
```

V textovém souboru **data.txt**, který je uložen v aktuálním adresáři, je zapsáno:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

```
disp(c)
    0.1067    0.8173    0.2599    0.1818    0.8693
    0.9619    0.8687    0.8001    0.2638    0.5797
    0.0046    0.0844    0.4314    0.1455    0.5499
    0.7749    0.3998    0.9106    0.1361    0.1450
```

Základy práce se soubory

Příklad: čtení z textového souboru `data.txt`

```
fr=fopen('data.txt','r');
c = fscanf(fr,'%g',[4,Inf]);
fclose(fr);
```


```
disp(c)
```

```
→ 0.1067   0.8173   0.2599   0.1818   0.8693
→ 0.9619   0.8687   0.8001   0.2638   0.5797
→ 0.0046   0.0844   0.4314   0.1455   0.5499
→ 0.7749   0.3998   0.9106   0.1361   0.1450
```

V textovém souboru `data.txt`, který je uložen v aktuálním adresáři, je zapsáno:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

Základy práce se soubory

Příklad: čtení z textového souboru `data.txt`

```
fr=fopen('data.txt','r');
c = fscanf(fr,'%g',[10,Inf]);
fclose(fr);
```

```
disp(c)
```

```
    0.1067    0.4314
    0.9619    0.9106
    0.0046    0.1818
    0.7749    0.2638
    0.8173    0.1455
    0.8687    0.1361
    0.0844    0.8693
    0.3998    0.5797
    0.2599    0.5499
    0.8001    0.1450
```

V textovém souboru `data.txt`, který je uložen v aktuálním adresáři, je zapsáno:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

Základy práce se soubory

Příklad: čtení z textového souboru `data.txt`

```
fr=fopen('data.txt','r');
c = fscanf(fr,'%g',[10,Inf]);
fclose(fr);
```

```
disp(c)
```

```
    0.1067    0.4314
    0.9619    0.9106
    0.0046    0.1818
    0.7749    0.2638
    0.8173    0.1455
    0.8687    0.1361
    0.0844    0.8693
    0.3998    0.5797
    0.2599    0.5499
    0.8001    0.1450
```

10 řádků a neznámý počet sloupců (chci číst až do konce souboru)

V textovém souboru `data.txt`, který je uložen v aktuálním adresáři, je zapsáno:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

Základy práce se soubory

Příklad: čtení z textového souboru **data.txt**

```
fr=fopen('data.txt','r');
c = fscanf(fr,'%g',[6,Inf]);
fclose(fr);
```

```
disp(c)
```

```
    0.1067    0.0844    0.1818    0.5499
    0.9619    0.3998    0.2638    0.1450
    0.0046    0.2599    0.1455         0
    0.7749    0.8001    0.1361         0
    0.8173    0.4314    0.8693         0
    0.8687    0.9106    0.5797         0
```

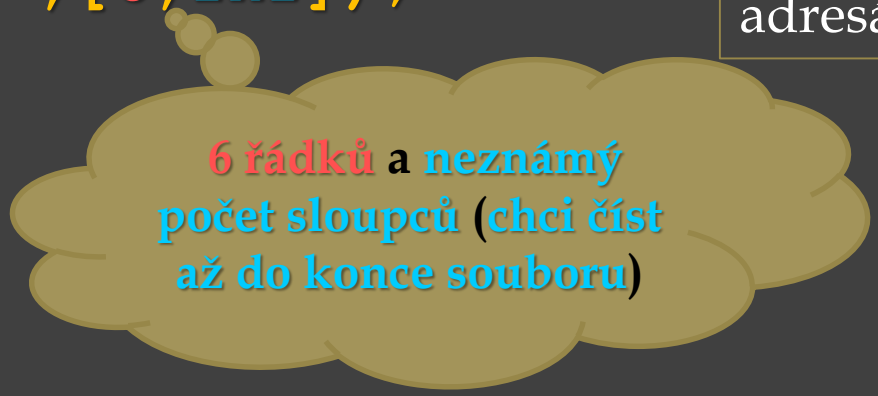
V textovém souboru **data.txt**, který je uložen v aktuálním adresáři, je zapsáno:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

Základy práce se soubory

Příklad: čtení z textového souboru `data.txt`

```
fr=fopen('data.txt','r');
c = fscanf(fr,'%g',[6,Inf]);
fclose(fr);
```




```
disp(c)
```

```
    0.1067    0.0844    0.1818    0.5499
    0.9619    0.3998    0.2638    0.1450
    0.0046    0.2599    0.1455         0
    0.7749    0.8001    0.1361         0
    0.8173    0.4314    0.8693         0
    0.8687    0.9106    0.5797         0
```

V textovém souboru `data.txt`, který je uložen v aktuálním adresáři, je zapsáno:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

Základy práce se soubory

Příklad: čtení z textového souboru `data.txt`

```
fr=fopen('data.txt','r');
c = fscanf(fr,'%g',[6,Inf]);
fclose(fr);
```

6 řádků a neznámý počet sloupců (chci číst až do konce souboru)

```
disp(c)
```

```
→ 0.1067    0.0844    0.1818    0.5499
→ 0.9619    0.3998    0.2638    0.1450
→ 0.0046    0.2599    0.1455         0
→ 0.7749    0.8001    0.1361         0
→ 0.8173    0.4314    0.8693         0
→ 0.8687    0.9106    0.5797         0
```

doplněno nulami

V textovém souboru `data.txt`, který je uložen v aktuálním adresáři, je zapsáno:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

Základy práce se soubory

Příklad: čtení z textového souboru **data.txt**

```
fr=fopen('data.txt','r');
c = fscanf(fr,'%g',[3,2]);
fclose(fr);
```

```
disp(c)
    0.1067    0.7749
    0.9619    0.8173
    0.0046    0.8687
```

V textovém souboru **data.txt**, který je uložen v aktuálním adresáři, je zapsáno:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

Základy práce se soubory

Příklad: čtení z textového souboru `data.txt`

```
fr=fopen('data.txt','r');
c = fscanf(fr,'%g',[3,2]);
fclose(fr);
```


```
disp(c)
```

```
   0.1067    0.7749
   0.9619    0.8173
   0.0046    0.8687
```

V textovém souboru `data.txt`, který je uložen v aktuálním adresáři, je zapsáno:

```
0.106653
0.961898
0.00463422
0.77491
0.817303
0.868695
0.0844358
0.399783
0.25987
0.800068
0.431414
0.910648
0.181847
0.263803
0.145539
0.136069
0.869292
0.579705
0.54986
0.144955
```

Grafy

Výpočetní systémy umožňují vykreslit více grafů do jednoho grafického okna:

- vedle sebe, pod sebe - rozdělení grafického okna (`subplot`)
- přes sebe – např. `plot(x1,y1,x2,y2,…,xn,yn)`
- přes sebe – `hold`

  `hold on` – přidrží aktuální graf v grafickém okně (zmrazí aktuální grafickou obrazovku) a všechny následující grafické výstupy do něho přikresluje, lze tedy nakreslit více grafů do jednoho grafického okna postupně
  `hold off` – vypnutí, konec možnosti kreslit více grafů do jednoho grafického okna (tj. opět mazání předchozích grafů)

`text(x,y,'nejaky text')` – umístí text na souřadnice x, y

`gtext('nejaky g-text')` – umístí text na souřadnice tam, kam je kliknuto myší (platí v MATLABu)

Dvojdimenzionální grafy

Soustavy souřadnic

- kartézská soustava souřadnic je taková soustava souřadnic, u které jsou souřadné osy vzájemně kolmé a protínají se v jednom bodě - počátku soustavy souřadnic.

  `plot(x,y),semilogx(x,y),semilogy(x,y), …`

- polární soustava souřadnic je taková soustava souřadnic v rovině, u které jedna souřadnice (označovaná *r*) udává vzdálenost bodu od počátku souřadnic, druhá souřadnice (označovaná *φ*) udává úhel spojnice tělesa a počátku od zvolené osy ležící v rovině.

  `polar(φ,r)`

Dvojdimenzionální grafy

`plot(x,y,S)` – rovinný graf s **lineárním** dělením na osách x, y (tj. „klasický“ graf x, y, v **kartézských souřadnicích**), jsou-li x a y vektory o stejné délce, pak **`plot(x,y)`** vykreslí x-y graf,

`semilogx(x,y,S)` – graf jako **`plot`**, ale s **logaritmickou** stupnicí (o základu 10) **na ose** x,

`semilogy(x,y,S)` – graf jako **`plot`**, ale s **logaritmickou** stupnicí (o základu 10) **na ose** y,

`loglog(x,y,S)` – graf jako **`plot`**, ale s **logaritmickou** stupnicí (o základu 10) **na obou osách** x, y.

`polar(uhel,velikost,S)` – graf v **polárních souřadnicích**

`compass(u,v,S)` – vektor se složkami u, v ve formě **šipky vycházející z počátku**

Parametr **`S`** je **řetězec**, který musí být v apostrofech (některé výpočetní systémy připouštějí i uvozovky) a specifikuje barvu, způsob vykreslení a styl křivky nebo typ značky bodu.

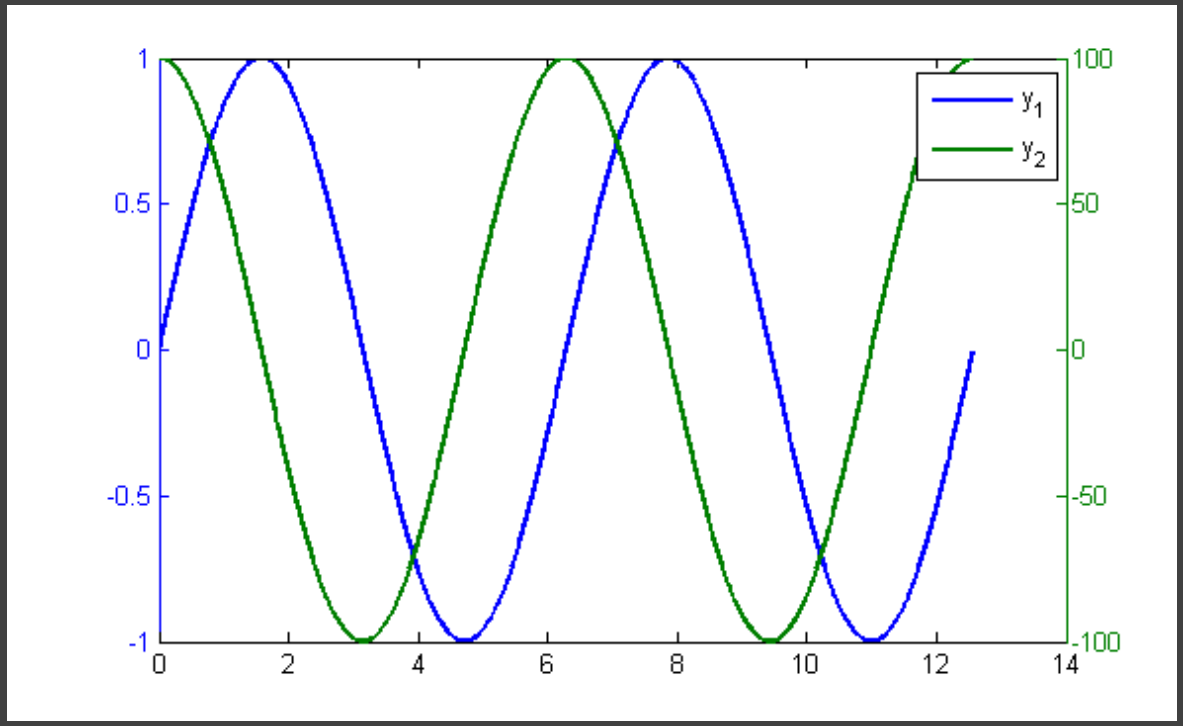
Pozn. barvy, typy značek bodů, styly čar – viz předchozí přednášky a viz **`help plot`**

Dvojdimenzionální grafy

plotyy – graf s dvěma různými osami y pro dvě různé křivky, např. **plotyy** (x_1, y_1, x_2, y_2) – osa pro graf y_1 vlevo, osa pro graf y_2 vpravo

Příklad:
Vykreslení grafů funkcí $y_1 = \sin(x)$, $y_2 = 100\cos(x)$

```
x=[0:0.01:4*pi];
y1=sin(x);
y2=100*cos(x);
plotyy(x,y1,x,y2)
legend('y_1','y_2')
```

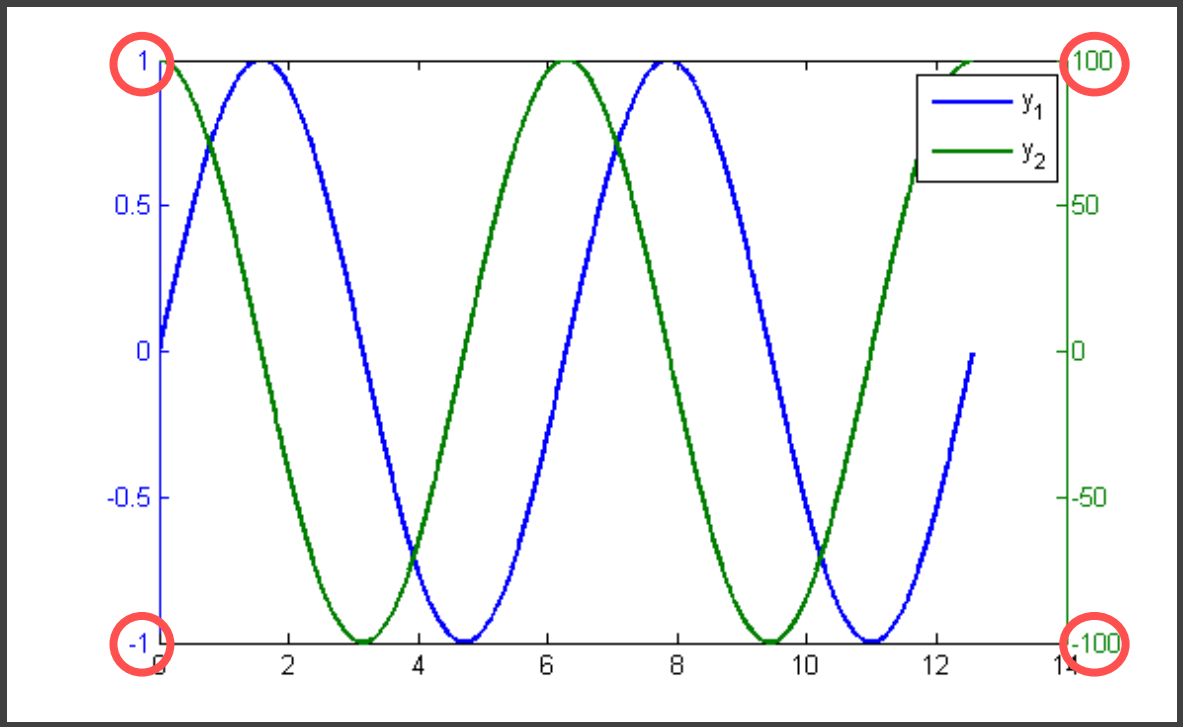


Dvojdimenzionální grafy

plotyy – graf s dvěma různými osami y pro dvě různé křivky, např. **plotyy** (x_1, y_1, x_2, y_2) – osa pro graf y_1 vlevo, osa pro graf y_2 vpravo

Příklad:
Vykreslení grafů funkcí $y_1 = \sin(x)$, $y_2 = 100\cos(x)$

```
x=[0:0.01:4*pi];
y1=sin(x);
y2=100*cos(x);
plotyy(x,y1,x,y2)
legend('y_1','y_2')
```



Dvojdimenzionální grafy

`compass(u,v,S)` – vektor se složkami *u*, *v* ve formě **šipky vycházející z počátku**

Příklady:

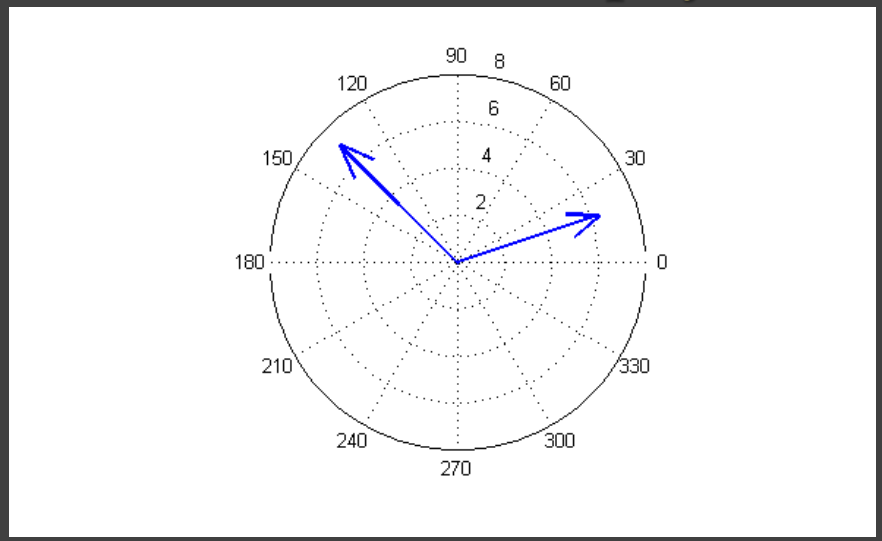
Zobrazení vektorů

a = [-5,5], *b* = [6,2]

```
compass([-5,6],[5,2])
```

nebo jiným způsobem:

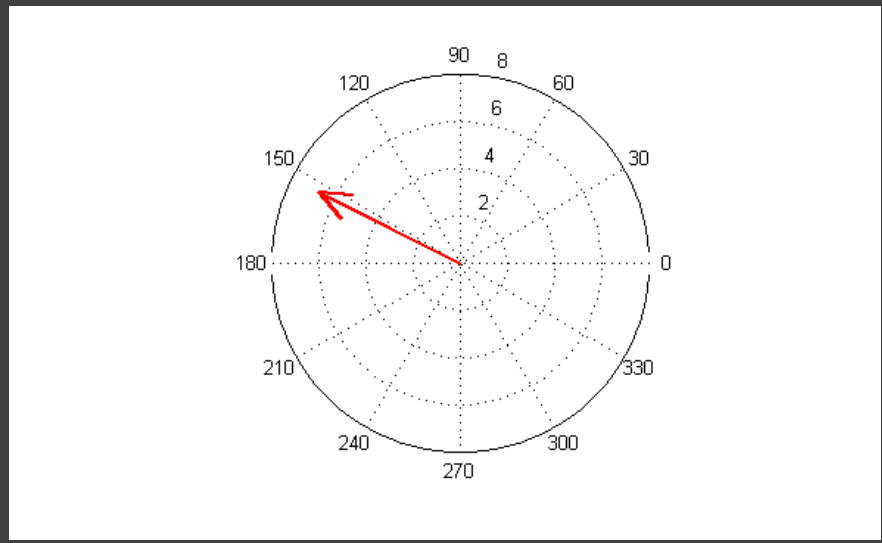
```
a = [-5,5]; b = [6,2];
compass([a(1),b(1)],[a(2),b(2)])
```



Zobrazení komplexních čísel – viz předchozí přednášky

Např.

```
compass(-6+3i,'r')
```



Dvojdimenzionální grafy

`polar(uhel,velikost,S)` – graf v **polárních souřadnicích**

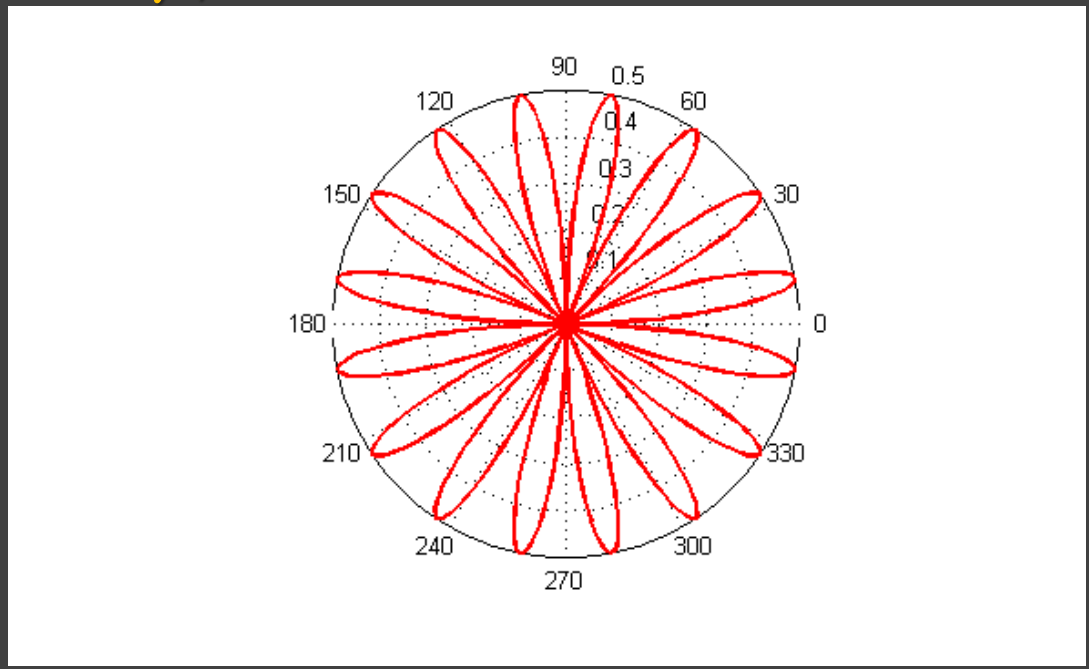
Příklad:

graf v polárních souřadnicích

$r = \sin(4\varphi)\cos(4\varphi)$,

kde φ je z intervalu od **0** do 2π.

```
fi = 0:0.01:2*pi;
y = sin(4.*fi).*cos(4.*fi);
polar(fi, y, 'r')
```

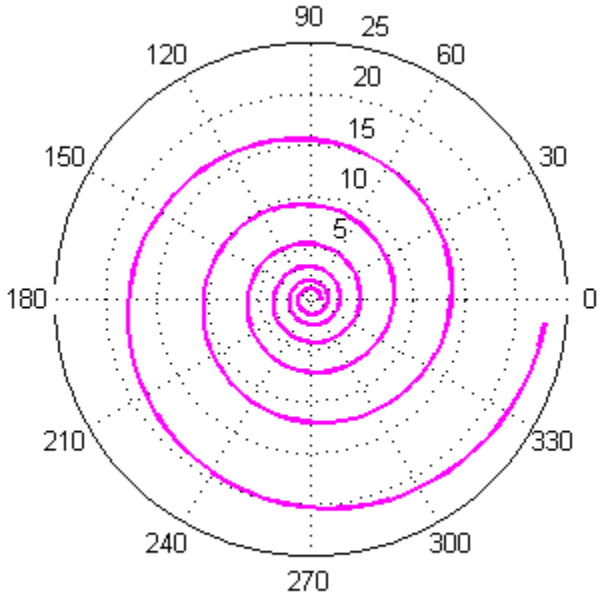


Dvojdimenzionální grafy

`polar(uhel,velikost,S)` – graf v **polárních souřadnicích**

Příklad:
graf logaritmické spirály v polárních souřadnicích
Logaritmická spirála se vyskytuje např. v kresbě ulit plžů, je dána rovnicí $r = a\, e^{m\varphi}$
Křivka vykreslena pro $a = 1$, $m = 1/12$, φ z intervalu od **0** do **12** π.

```
phi = 0:0.1:12*pi;
r = exp((1/12)*phi)
polar(phi,r,'m')
```



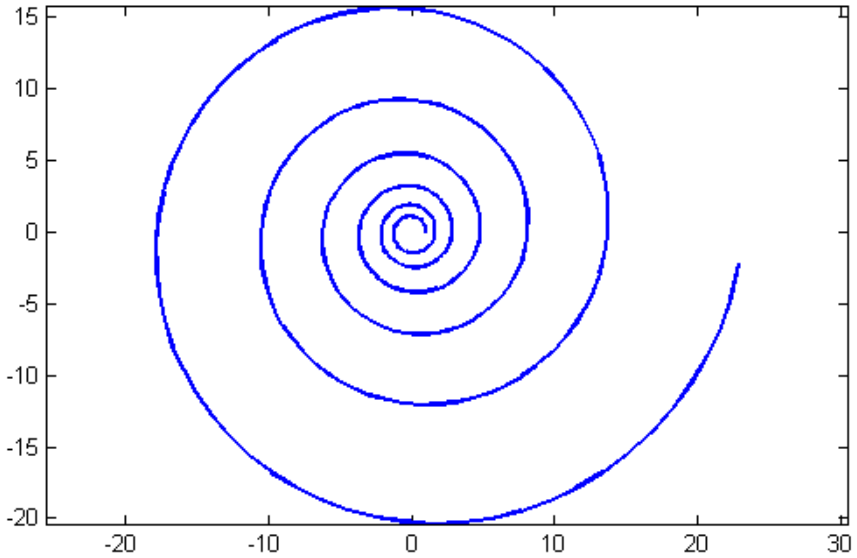
Dvojdimenzionální grafy

Pozn.: užitečná funkce **`pol2cart`** – převod souřadnic polárních na kartézské

```
[x,y] = pol2cart(uhel, vzdálenost_od_počátku);
```

Pokračování příkladu:
graf logaritmické spirály z předchozího příkladu $r = a\, e^{m\varphi}$, kde $a = 1$, $m = 1/12$, φ je z intervalu od **0** do **12** π, je převeden do kartézských souřadnic a vykreslen příkazem **`plot`**.

```
phi = 0:0.1:12*pi;
r = exp((1/12)*phi);
[x,y]=pol2cart(phi,r);
plot(x,y)
```



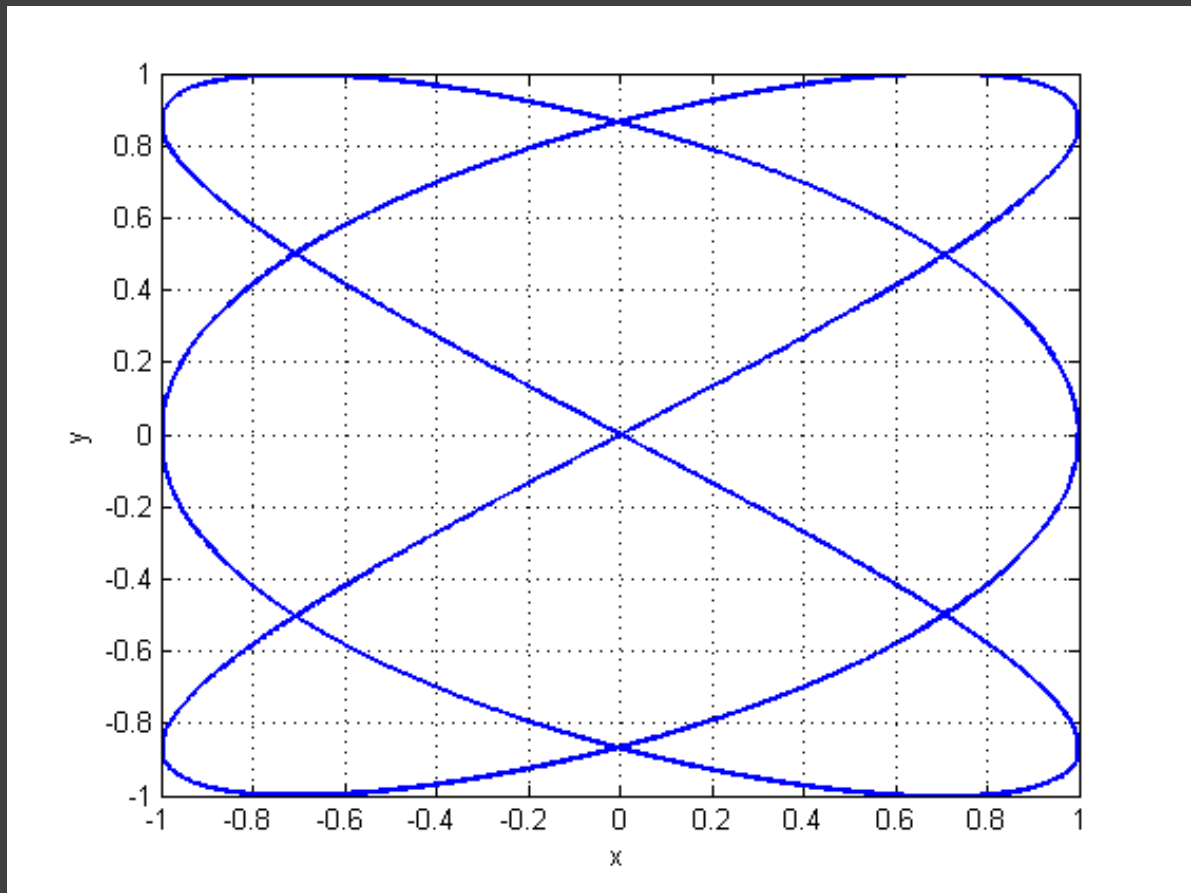
Naopak:
`cart2pol` – převod souřadnic kartézských na polární

Dvojdimenzionální grafy

Příklad:

- graf parametrizované křivky (popsán parametrickými rovnicemi)
- *t* je parametr, na kterém závisí *x*, *y*

```
t = 0:.001:2*pi;
x = cos(3.*t);
y = sin(2.*t);
plot(x,y)
grid
 % grid přikreslí do
 % grafu síť (mřížku)
xlabel('x')
ylabel('y')
 % popis os
```

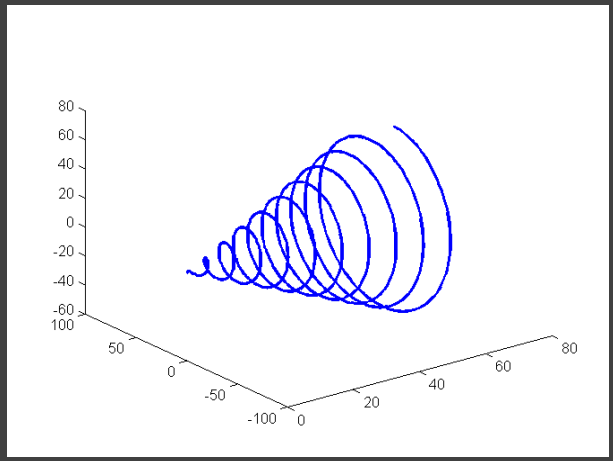


Trojdimenzionální grafy

- křivkové grafy

body, úsečky, křivky v prostoru

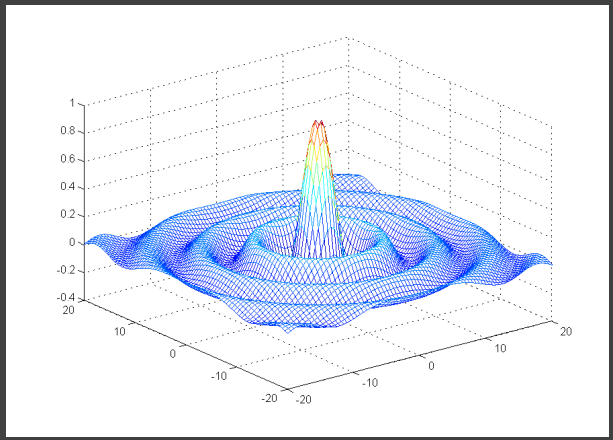
např. `plot3(x,y,z)` – vyjadřuje obvykle závislost y a z na x



- plošné grafy

síťové(drátové), povrchové grafy

např. `mesh(x,y,z)`, `surf(x,y,z)` atp. – vyjadřují obvykle závislost z na x a y



Trojdimenzionální grafy

`plot3(x,y,z,S)` – 3D graf křivkový

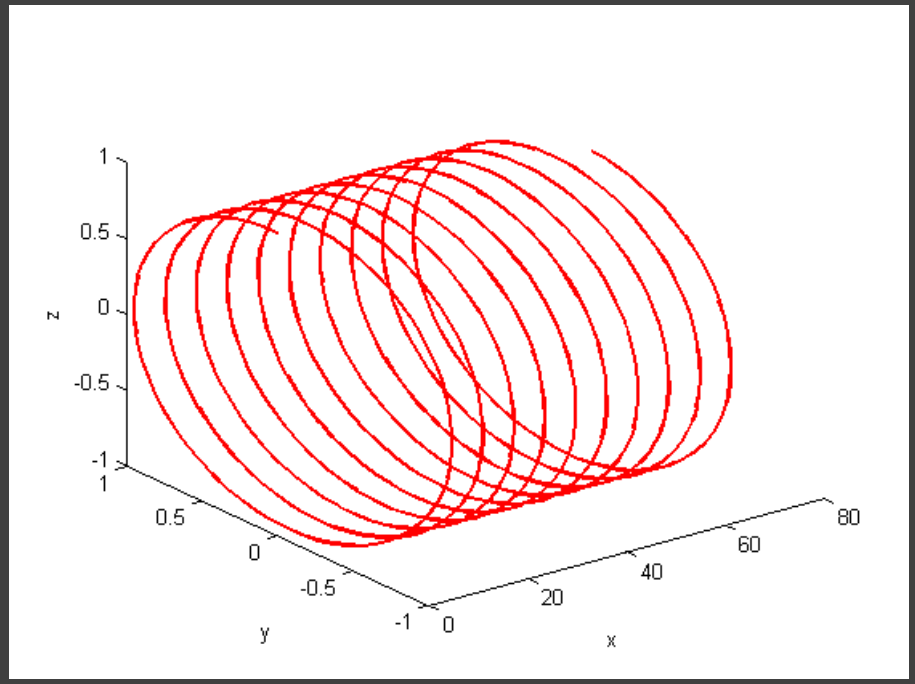
– vykreslí křivku v prostoru procházející body o souřadnicích x, y, z, které jsou prvky stejně dlouhých vektorů **`x`**, **`y`**, **`z`**,

– vyjadřuje obvykle závislost y a z na x,

– lze použít podobné příkazy a parametry jako ve 2D (**`plot`**), parametr **`S`** je **řetězec**, který specifikuje barvu, způsob vykreslení a styl křivky nebo typ značky bodu.

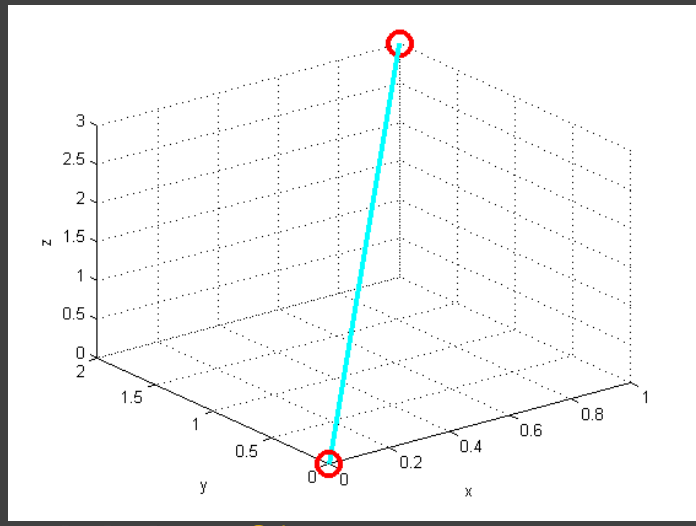
Příklad: graf křivky popsané parametrickými rovnicemi:
$x = t$, $y = \sin(t)$, $z = \cos(t)$,
pro t od 0 do 20π.

```
t = [0:0.1:20*pi]
plot3(t,sin(t),cos(t),'r')
xlabel('x')
ylabel('y')
zlabel('z')
```



Trojdimenzionální grafy

Příklad: vykreslení úsečky v prostoru z bodu [0, 0, 0] do bodu [1, 2, 3]

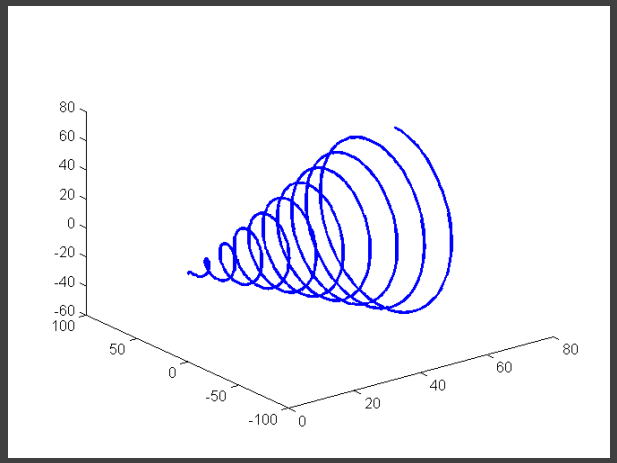



```
x=[0,1]; % x-ové souřadnice bodů
y=[0,2]; % y-ové souřadnice bodů
z=[0,3]; % z-ové souřadnice bodů
plot3(x,y,z,'c','LineWidth',3)
% zvolena barva - modrozelená (cyan)
% zvolena tloušťka čáry
                        % 'LineWidth',3
hold on
plot3(0,0,0,'ro','MarkerSize',15,'Linewidth',3)
plot3(1,2,3,'ro','MarkerSize',15,'LineWidth',3)
hold off
 % 'ro' - body jsou zobrazeny červenými kolečky
 % 'MarkerSize',15 - velikost značky
 % 'LineWidth',3 - tloušťka čáry kolečka (značky)
xlabel('x') % popis osy x
ylabel('y') % popis osy y
zlabel('z') % popis osy z
grid % přidá do grafu síť, mřížku
```

Trojdimenzionální grafy

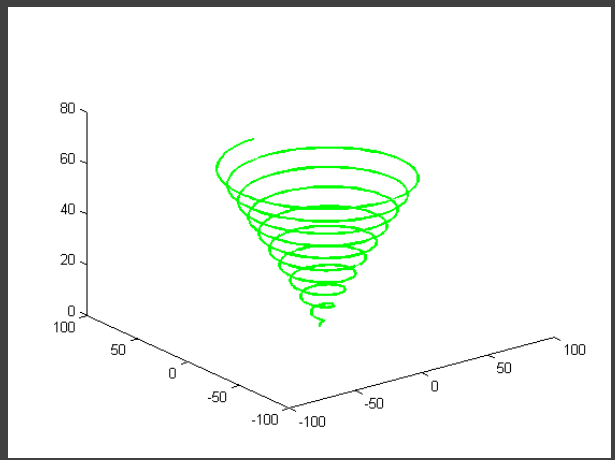
Příklady:
Graf křivky popsané parametrickými rovnicemi:
$x = t$, $y = t \sin(t)$, $z = t \cos(t)$,
pro t od **0** do 20π.

```
t = [0:0.1:20*pi];
plot3(t,t.*sin(t),t.*cos(t))
```



Graf křivky popsané parametrickými rovnicemi:
$x = t \sin(t)$, $y = t \cos(t)$, $z = t$,
pro t od **0** do 20π.

```
t = [0:0.1:20*pi];
plot3(t.*sin(t),t.*cos(t),t,'g')
```



Trojdimenzionální grafy

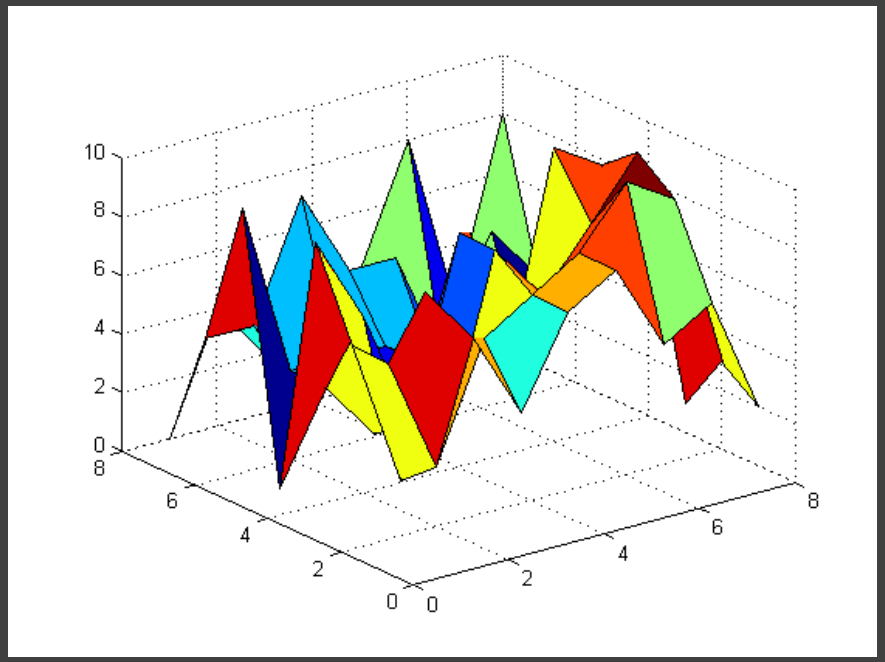
3D "plošné" grafy – 3D plochy a sítě

`mesh(X,Y,Z)` – vykreslí nad souřadnicemi *x*, *y* **síť** (**drátěný model**) tvarovanou podle *Z* (lze též uvést **`mesh(Z)`** – nemám potom regulérní hodnoty *x*, *y*)

`surf(X,Y,Z)` - vykreslí nad souřadnicemi *x*, *y* **plochu** (vystínovanou, vybarvenou) tvarovanou podle *Z* (lze též uvést **`surf(Z)`** – nemám potom regulérní hodnoty *x*, *y*)

Příklad:
Matice náhodných čísel s 8 řádky a 8 sloupci zobrazená pomocí **`surf`**. Rozsah osy *x* a *y* bude dán automaticky od 1 do 8 (indexy prvků matice).

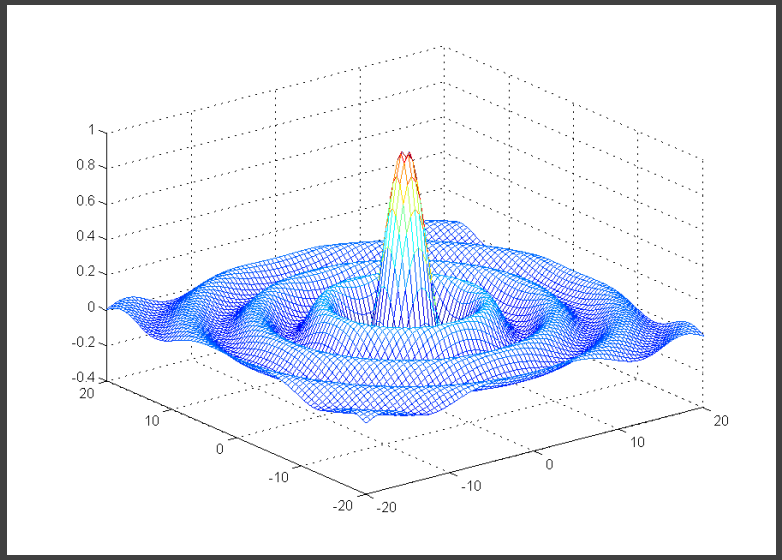
```
N = round(rand(8,8).*10);
surf(N)
```



Trojdimenzionální grafy

Příklad:
Graf funkce:

$$z(x,y) = \frac{\sin\left(\sqrt{x^2+y^2}\right)}{\sqrt{x^2+y^2}}$$



pro x, y od **-20** do **20** s krokem **0,5**.

```
x = -20:0.5:20;
y = x;
[X,Y] = meshgrid(x,y);
Z = sin(sqrt(X.^2+Y.^2))./sqrt(X.^2+Y.^2);
mesh(X,Y,Z)
```

Pokud mají x, y stejný rozsah **-20** až **20**, lze zapsat, např. takto:

```
[X,Y] = meshgrid(-20:0.5:20);
Z = sin(sqrt(X.^2+Y.^2))./sqrt(X.^2+Y.^2);
mesh(X,Y,Z)
```

Trojdimenzionální grafy

Proč používáme **`meshgrid`**?

- **`[X,Y] = meshgrid(x,y)`** vytvoří **pomocné matice X, Y**, jejichž prvky obsahují souřadnice bodů v rovině *xy*
- pro usnadnění zápisu výpočtu – s takto vytvořenými souřadnicemi mohu zapisovat rovnici pro výpočet **Z** "normálně" dle matematického zápisu, pouze nesmím zapomenout na operace prvek po prvku (tečka-notaci).

Pozn.: Funkce vracející jako výsledek dvě a více hodnot (mohou to být i dvě matice) bude mít hlavičku:

```
function [prvni,druha] = vraci_dve(vstupni_parametry)
...atd...
prvni = nějaký výsledek;
druha = nějaký výsledek;
```

Volání této funkce bude potom vypadat např.:

```
[a, b] = vraci_dve(x)
a = ...
b = ...
```

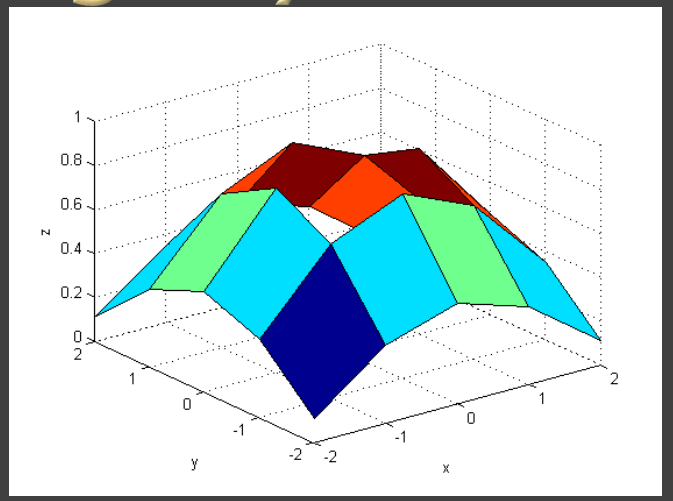
Příkladem takové funkce je **`meshgrid`**.

Trojdimenzionální grafy

Pokračování příkladu:

Graf funkce: $$z(x,y) = \frac{\sin\left(\sqrt{x^2+y^2}\right)}{\sqrt{x^2+y^2}}$$

pro x, y od **-2** do **2** s krokem **1**.




```
[X,Y] = meshgrid(-2:2)
Z = sin(sqrt(X.^2+Y.^2))./sqrt(X.^2+Y.^2)
surf(X,Y,Z)
xlabel('x')
ylabel('y')
zlabel('z')
```

Za příkazy nejsou středníky, vypíší se matice **X**, **Y**, **Z**

```
X =
    -2    -1     0     1     2
    -2    -1     0     1     2
    -2    -1     0     1     2
    -2    -1     0     1     2
    -2    -1     0     1     2
```

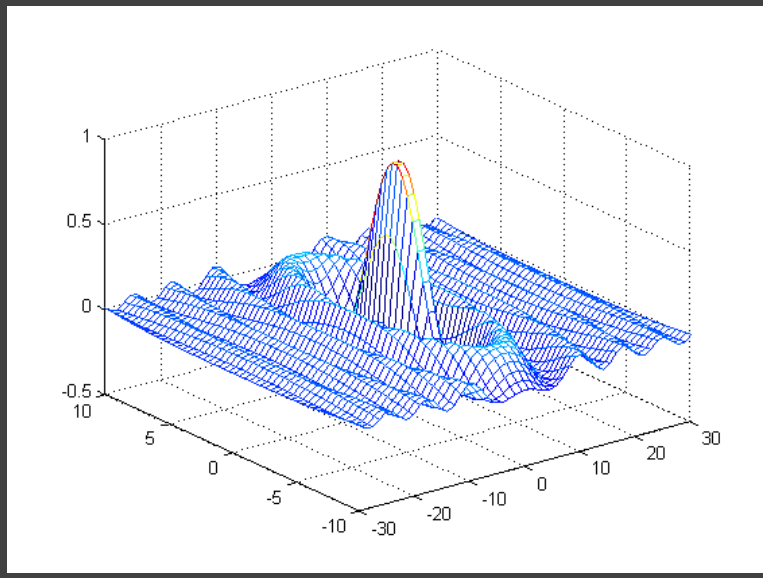
```
Y =
    -2    -2    -2    -2    -2
    -1    -1    -1    -1    -1
     0     0     0     0     0
     1     1     1     1     1
     2     2     2     2     2
```

```
Z =
    0.11    0.35    0.45    0.35    0.11
    0.35    0.70    0.84    0.70    0.35
    0.45    0.84     NaN    0.84    0.45
    0.35    0.70    0.84    0.70    0.35
    0.11    0.35    0.45    0.35    0.11
```

Trojdimenzionální grafy

Pokračování příkladu:

Graf funkce: $$z(x,y)=\frac{\sin\left(\sqrt{x^2+y^2}\right)}{\sqrt{x^2+y^2}}$$



pro x od -30 do 30 a y od -10 do 10
(různý rozsah os x, y)

```
x = linspace(-30,30,50);
y = linspace(-10,10,50);
[X,Y] = meshgrid(x,y);
Z = sin(sqrt(X.^2+Y.^2))./sqrt(X.^2+Y.^2);
mesh(X,Y,Z)
```

pro x od -20 do 20 a y od -40 do 40

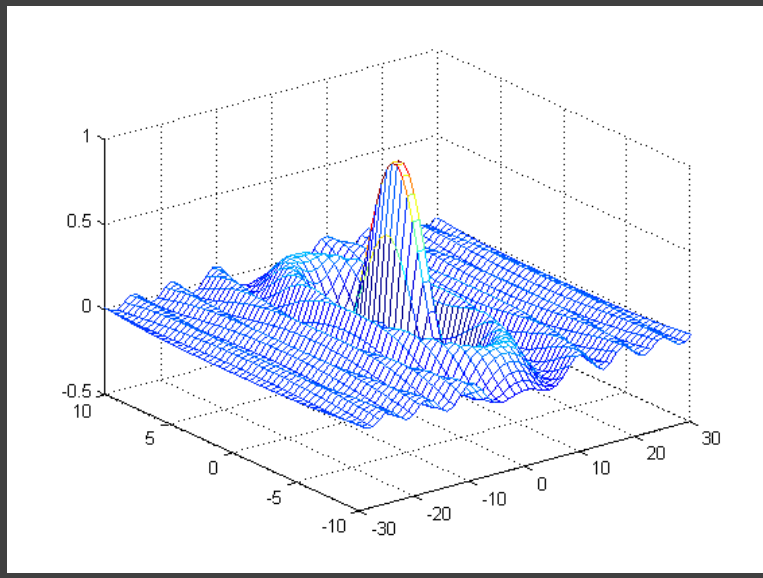
```
[X,Y] = meshgrid(-20:0.5:20,-40:0.5:40);
Z = sin(sqrt(X.^2+Y.^2))./sqrt(X.^2+Y.^2);
mesh(X,Y,Z)
```

Trojdimenzionální grafy

Pokračování příkladu:

Graf funkce:

$$z(x,y) = \frac{\sin\left(\sqrt{x^2+y^2}\right)}{\sqrt{x^2+y^2}}$$



pro x od -30 do 30 a y od -10 do 10
(různý rozsah os x, y)

```
x = linspace(-30,30,50);
y = linspace(-10,10,50);
[X,Y] = meshgrid(x,y);
Z = sin(sqrt(X.^2+Y.^2))./sqrt
mesh(X,Y,Z)
```

pro x od -20 do 20 a y od -40 do 40

```
[X,Y] = meshgrid(-20:0.5:20,-4
Z = sin(sqrt(X.^2+Y.^2))./sqrt
mesh(X,Y,Z)
```