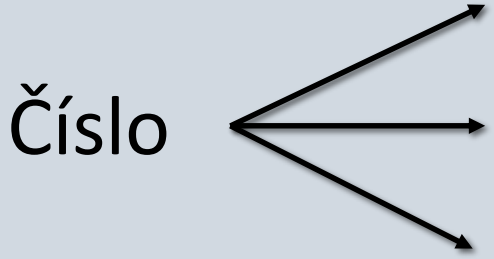
Programování (PRO)

7. cv.

Základní datové typy

Datový typ – definuje druh hodnoty, kterou smí nabývat proměnná nebo konstanta.



Číslo	Celé číslo	5	int 8 (16,32,64)
	Reálné číslo	2.3	double
	Komplexní číslo	5+2i	complex
Znak	Symbol	`'s'`	char
	Řetězec	`'slovo'`	char array
	Věta (text)	“May the Force be with you“	string
Logická hodnota	Logická hodnota (pravdivostní, boolovská)	1 (true) X 0 (false)	logical

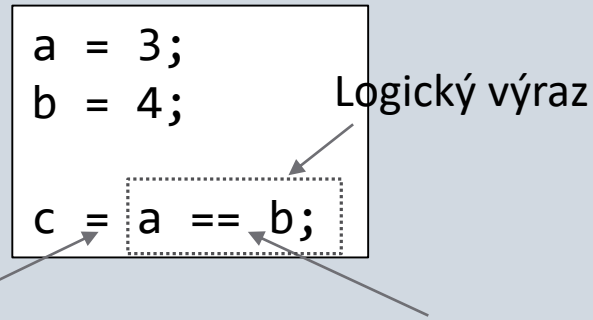
Relační operátory

MATLAB	Čeština	Matematika
==	je rovno	=
~=	není rovno	≠
<	je menší	<
>	je větší	>
<=	je menší nebo rovno	≤
>=	je větší nebo rovno	≥

Logické výrazy

Nový datový typ `logical` a `logical array`

Logika	Čeština
0	Nepravda
1	Pravda



Vyzkoušejte:

```
a = 3;
b = 4;

a == b
b ~= a
a > b
a < b
a <= b
a >= b
```

Výsledkem používání relačních operátorů je vždy logická nula nebo logická jednička. Nebo vektor logických nul nebo logických jedniček, pokud alespoň jeden z operandů je vektor.

Logické vektory a matice

Logické na matice (vektory) jsou výsledkem logických výrazů aplikovaných na matice (vektory)

Matice číselných hodnot

Matice logických hodnot
(1 kde byla hodnota větší než 5)

Součet všech log. jedniček
(-> počet čísel větších než 5 v matici)

```
>> A = randi([0,20], 4, 6)
A =
    17    13    20    20     8    13
    19     2    20    10    19     0
     2     5     3    16    16    17
    19    11    20     2    20    19

>> B = A>5
B =
  4×6 logical array

   1   1   1   1   1   1
   1   0   1   1   1   0
   0   0   0   1   1   1
   1   1   1   0   1   1

>> C = sum(B, "all")
C = 18
```

Indexovaní pomocí logických vektorů

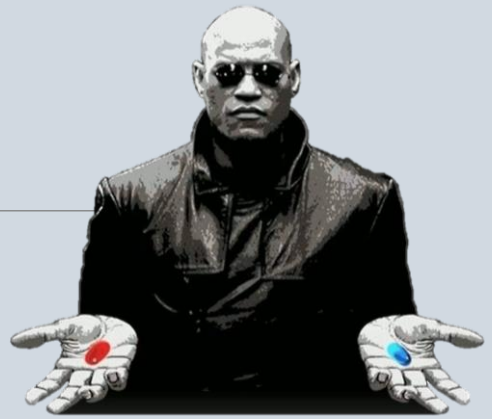
```
>> A = randi([0,10], 2, 3)
A =

     8     1     6
     9    10     1

>> I = logical([0, 1, 1, 0, 1, 1]);
>> X = A(I)

X =

     9     1     6     1
```

```
1 3 5
2 4 6
```

Indexujeme pomocí logického vektoru

Vektor musí mít stejnou délku jako je počet prvku matice

Větvení programu



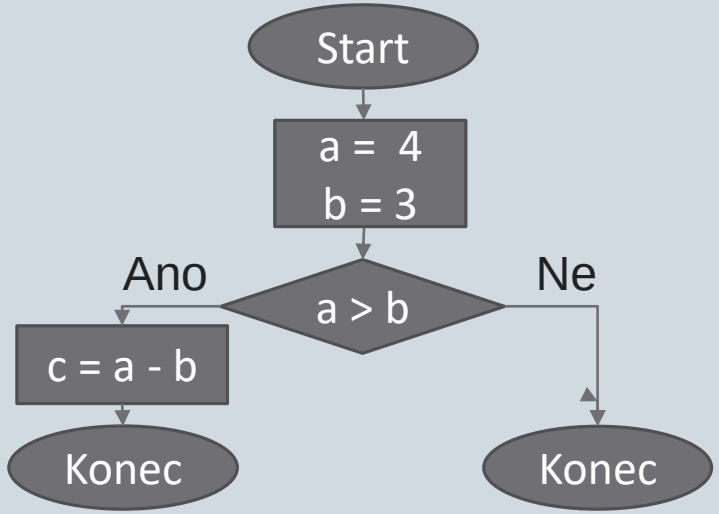
Umožňuje rozhodnout zda určité řádky vykonat X nevykonat anebo vykonat jiné

Program se nevětví

```
a = 4;
b = 3;

c = a - b;
```

Program se větví
(podmínka pro vykonaní řádků)

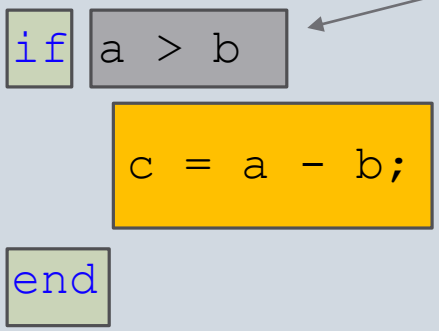
```
a = 4;
b = 3;

if a > b
    c = a - b;
end
```



Konstrukce podmínky if

Klíčové slovo, které oznamuje **začátek** podmíněného výrazu

Podmínka. Když je pravdivá, tak se provádí veškeré příkazy z bloku příkazů mezi začátkem a koncem podmíněného výrazu.

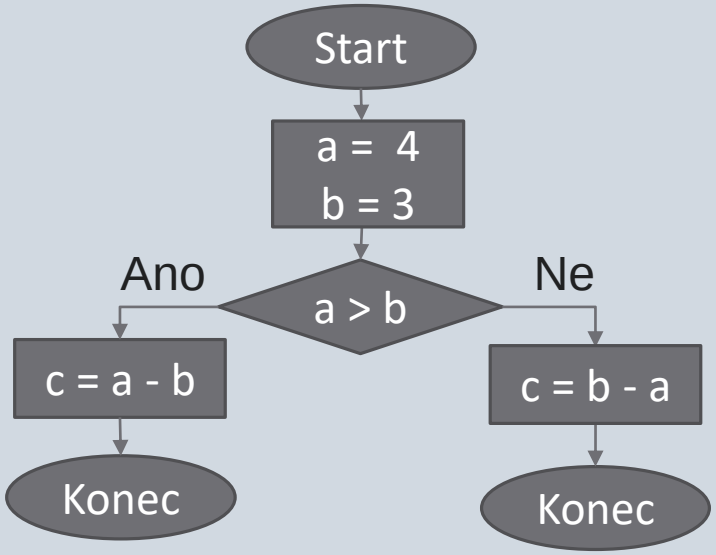


Blok příkazů. Jeden nebo několik příkazů, které se provádí pokud je podmínka pravdivá. Žádný příkaz z bloku příkazů se neprovede pokud podmínka není pravdivá.

Klíčové slovo, které oznamuje **konec** podmíněného výrazu.

Podmínka if-else

```
a = 4;
b = 3;

if a > b            %Když a je větší než b
   c = a - b;

else                %Když a je menší nebo se rovna b
   c = b - a;
end
```



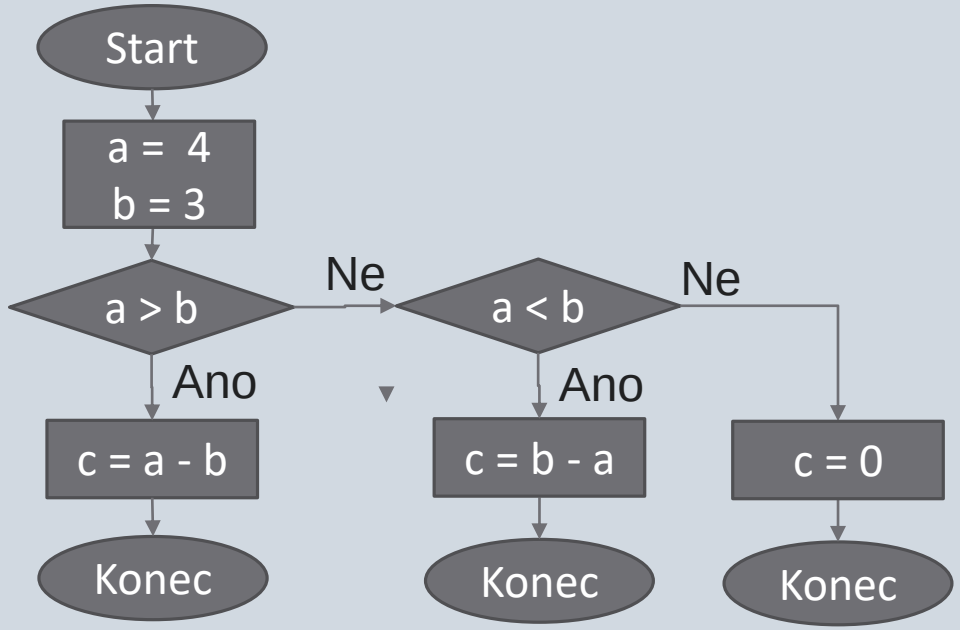
Konstrukce podmínky if-else

```
a = 4;
b = 3;

if a > b
    c = a - b;
else
    c = b - a;
end
```

Klíčové slovo, které oznamuje **začátek** bloku příkazů, které se provedou když předchozí podmínka není pravdivá. V tomto případě když `a < b` nebo `a == b`

Konstrukce podmínky if-elseif-else

```
a = 4;
b = 3;

if a > b          %Když a je větší než b
   c = a - b;

elseif a < b      %Když a není větší než b
   c = b - a;     % a zároveň a je menší než b

else              %Když a == b
   c = 0
end
```



Podmínky if-elseif-else

```
a = 4;
b = 3;

if a > b
   c = a - b;

elseif a < b
   c = b - a;

else
   c = 0;
end
```

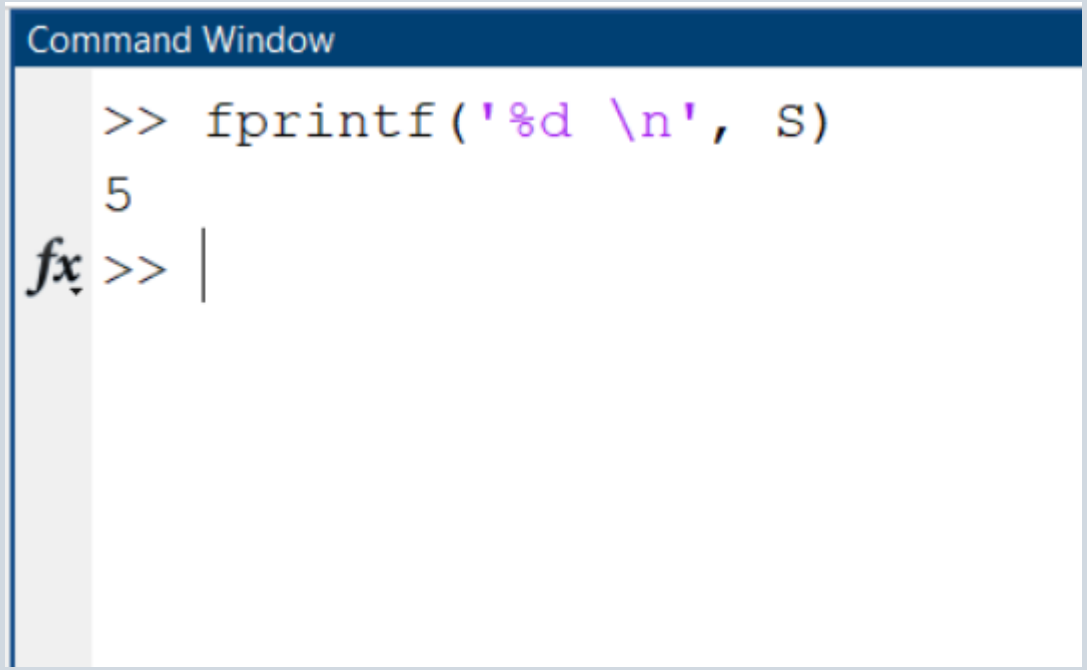
Klíčové slovo, které oznamuje **začátek** bloku příkazů, které se provedou když předchozí podmínka není pravdivá a zároveň je pravdivá podmínka příslušná tomuto bloku. V tomto případě když `a` není větší než `b` a zároveň `a < b`

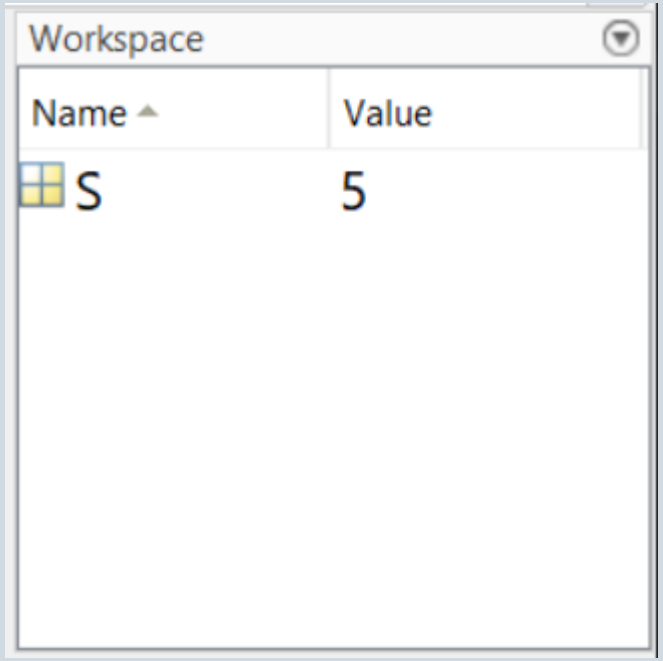
Podmínka

Hra „Bomba“

```
function Bomba(a)
%Zkuste zneškodnit bombu
%   a - jakékoliv číslo od 1 do 3

n = randi(3);          % toto je číslo, které je potřeba odhadnout

if a ~= n              %pokud a se nerovná
    fprintf('Číslo pro zneškodnění bomby bylo %d\n',n);
    fprintf('BOOM!!!\n');

else                   %ve všech ostatních případech
    fprintf('Gratulace!!! Bomba je zneškodněná\n');
end
end
```

Vypisování hodnot formátovaným textem





Vypisování hodnot formátovaným textem

```
disp('Ahoj Pepo') ↔  fprintf('Ahoj Pepo \n')
```

skok na další řádek

```
disp( promenna ) ↔     fprintf('%d',promenna)
```

%d – celé číslo
%f – desetinné číslo
%c – symbol (písmeno)
%s – text, věta

Skript:

```
cena = 25;
fprintf ('Pivo je za %d Kč', cena)
```

Výstup:

```
Pivo je za 25 Kč
```