C2110
Operační systém UNIX a základy programování

11. lekce

Petr Kulhánek

kulhanek@chemi.muni.cz

Národní centrum pro výzkum biomolekul, Přírodovědecká fakulta
Masarykova univerzita, Kotlářská 2, CZ-61137 Brno

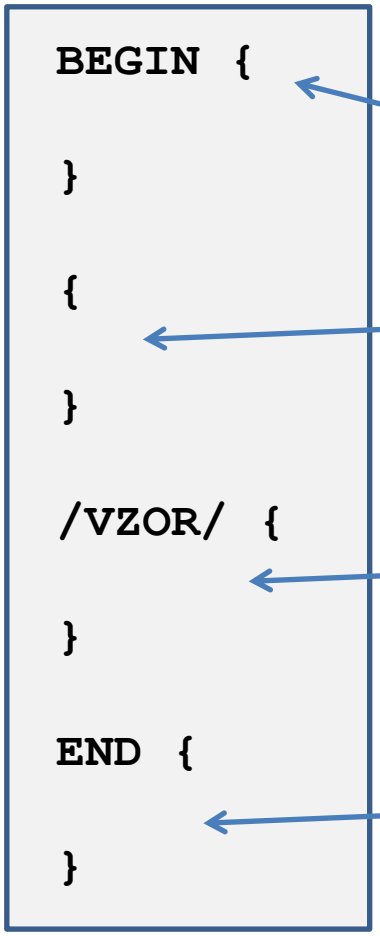
AWK

http://www.gnu.org/software/gawk/gawk.html

AWK je skriptovací jazyk navržený pro **zpracovávání textových dat**, ať už v podobě textových souborů nebo proudů. Jazyk využívá **řetězcové datové typy**, **asociativní pole** (pole indexovaná řetězcovými klíči) a **regulární výrazy**.

adaptováno z www.wikipedia.org

Struktura skriptu AWK



vykoná se před zpracováním souboru (volitelná sekce)

vykoná se pro každý záznam

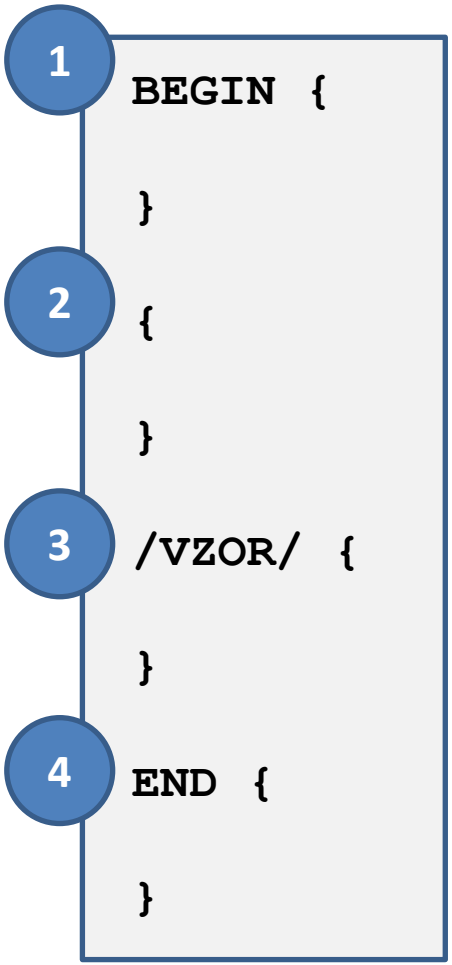
vykoná se pro každý záznam vyhovující VZORu

libovolný počet sekcí

vykoná se po zpracování souboru (volitelná sekce)

Blok je uzavřen do složených závorek {}.
Výše uvedené programové bloky jsou volitelné.
Ve výchozím nastavení je záznamem řádek souboru.

Průběh vykonávání skriptu



- Blok BEGIN (1) se vykoná (pokud je ve skriptu obsažen) před analýzou souboru.
  - Načte se záznam ze souboru. Ve výchozím nastavení je záznamem celý řádek analyzovaného souboru nebo proudu. Záznam se rozdělí na pole. Ve výchozím nastavení jsou pole jednotlivá slova v záznamu.
  - Pro daný záznam se vykoná blok (2).
  - Pokud záznamu vyhovuje VZOR, vykoná se blok (3).
  - .... vykonají se případně další bloky ....
- Blok END (4) se vykoná (pokud je ve skriptu obsažen) po analýze celého souboru.

Struktura bloku

komentáře jsou uvezeny znakem #

```
# tento blok pocita mezisoucet a analyzuje
# hodnotu ctvrteho sloupce

{
    # toto je komentar
    i = i + 1;
    f = f + $2; # tady pocitam mezisoucet
    printf("Mezisoucet je %10.3f\n",f);
    if( $3 == 5 ) {
        k = k + $4;
    }
}
```

příkazy se uvádějí na samostatné řádky, které je vhodné ukončit středníkem
středník je nutné použít, pokud uvádíme dva a více příkazů na jeden řádek

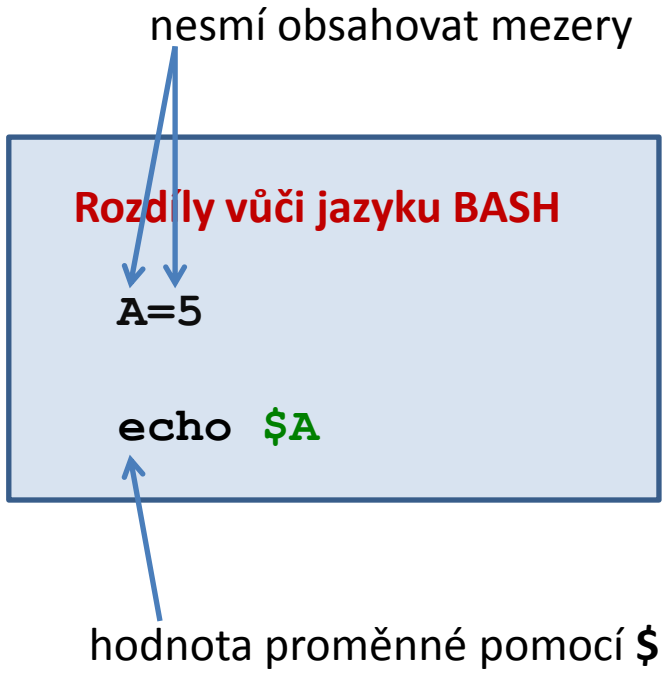
Proměnné

Přiřazení do proměnné:

```
A = 10;
B = "toto je text"
C = 10.4567;
D = A + C;
```

Hodnota proměnné:

```
print A + C;
print B;
```



Speciální proměnné:

NF	počet polí v aktuálně prováděném záznamu (Number of Fields)
NR	pořadí prováděného záznamu (Number of Records)
FS	oddělovač polí v záznamu (Field Separator), **výchozí je znak nové řádky \n**
RS	oddělovač záznamů (Record Separator) , **výchozí je mezera a tabulátor**
$0	celý záznam
$1, $2, $3 ...	jednotlivé pole záznamu

Proměnné, ...

$0	celý záznam
$1, $2, $3 ...	jednotlivé pole záznamu

znak **$** umožňuje programový přístup k jednotlivým polím záznamu

Příklad:

```
i=3;
print $i;
```

vytiskne hodnotu třetího pole

Matematické operace

Pokud lze proměnnou interpretovat jako číslo, lze použít následující aritmetické operátory:

++ hodnotu proměnné zvýší o jedničku

```
A++;
```

-- hodnotu proměnné sníží o jedničku

```
A--;
```

+ sečte dvě hodnoty

```
A = 5 + 6;
A = A + 1;
```

- odečte dvě hodnoty

```
A = 5 - 6;
A = A - 1;
```

***** vynásobí dvě hodnoty

```
A = 5 * 6;
A = A * 1;
```

/ vydělí dvě hodnoty

```
A = 5 / 6;
A = A / 1;
```

+= k proměnné přičte hodnotu

```
A += 3;
A += B;
```

-= od proměnné odečte hodnotu

```
A -= 3;
A -= B;
```

***=** proměnnou vynásobí hodnotou

```
A *= 3;
A *= B;
```

/= proměnnou podělí hodnotou

```
A /= 3;
A /= B;
```

Příkaz print

Příkaz **print** slouží k neformátovanému vypisování řetězců a čísel.

Syntaxe:

```
print hodnota1[,] hodnota2[,] ...;
```

pokud jsou hodnoty oddělené čárkou, ve výstupu se hodnoty oddělí mezerou

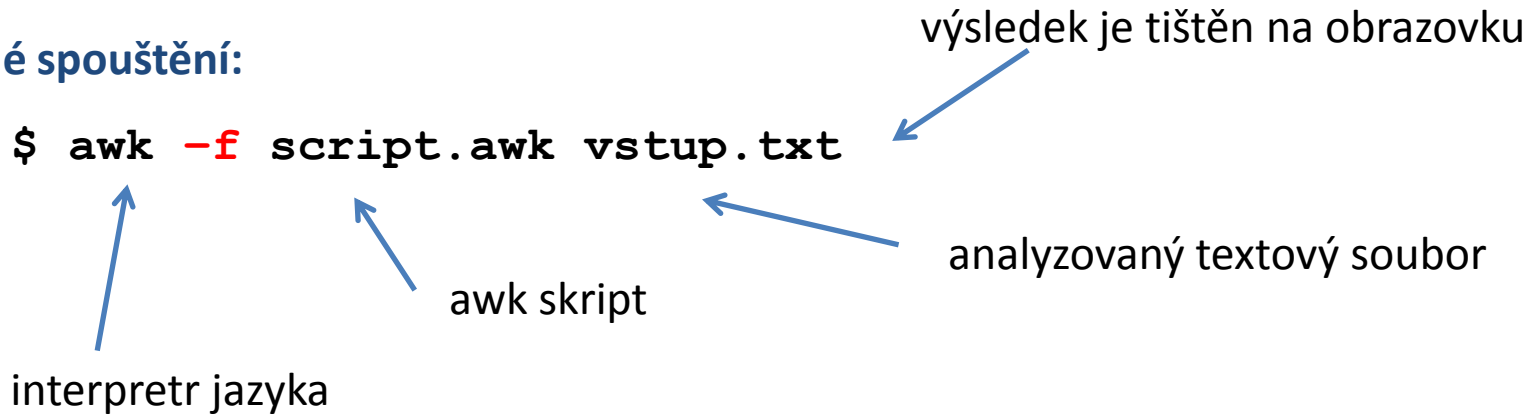
Příklady:

```
i = 5;
k = 10.456;
j = "hodnota promenne i =";
print j, i;
print "hodnota promenne k =", k;
```

Spouštění AWK skriptů

Zpracování textového souboru:

Nepřímé spouštění:



Analyzovaná data lze zaslat přes standardní vstup:

```
$ awk -f script.awk < vstup.txt

$ cat soubor.txt | awk -f script.awk
```

Spouštění AWK skriptů, ...

Přímé spouštění

```
$ ./script.awk vstup.txt

$ ./script.awk < vstup.txt

$ cat soubor.txt | ./script.awk
```

Skript `script.awk` **musí** mít nastaven příznak x (**executable**) a interpreter AWK (součást skriptu).

```
#!/usr/bin/awk -f
{
    i += NF;
}
END {
    print "Pocet slov je:", i;
}
```

Cvičení

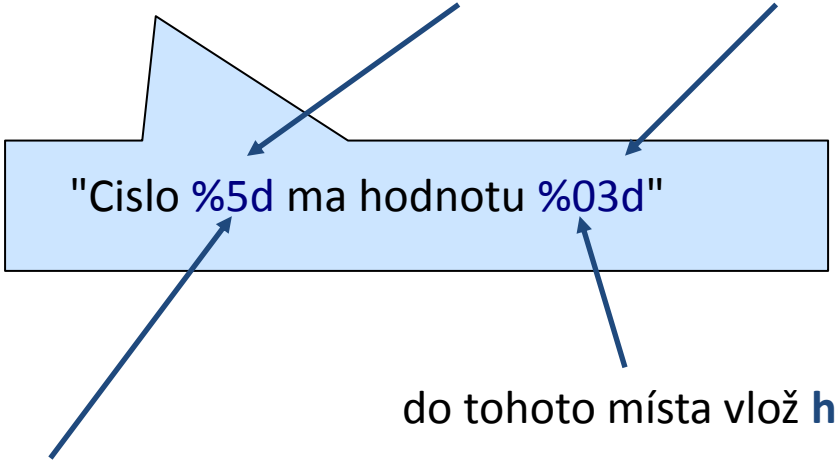
1. V domovském adresáři si vytvořte adresář **awk-data**.
2. Do adresáře **awk-data** zkopírujte soubory **matice.txt**, **produkt.log** a **rst.out** z adresáře **/home/kulhanek/Data/AWK**.
3. Napište skript, který vytiskne počet řádků, které obsahuje soubor **matice.txt**. Výsledek ověřte pomocí příkazu **wc**.
4. Napište skript, který vytiskne počet slov, které obsahuje soubor **matice.txt**. Výsledek ověřte pomocí příkazu **wc**.
5. Napište skript, který vytiskne druhý sloupec ze souboru **matice.txt**.
6. Napište skript, který sečte čísla v druhém sloupci souboru **matice.txt**.
7. Napište skript, který vypočítá průměrnou hodnotu čísel uvedených v druhém sloupci souboru **matice.txt**.

Funkce printf

Funkce **printf** slouží k vypisování formátovaných textů a čísel.

Syntaxe:

```
printf("format", hodnota1, hodnota2, ...);
```



Rozdíl vůči jazyku BASH:

```
printf [format] [hodnota1] [hodnota2] ...
```

příkaz

argumenty příkazu se oddělují mezerou

Podmínky

```
if( logicky_vyraz ) {
      prikaz2;
      ...
} else {
      prikaz3;
      ...
}
```

Pokud je **logicky_vyraz** pravda, vykoná se **prikaz2**. V opačném případě se vykoná **prikaz3**.

Příklad:

```
if( $1 > max ){
      max = $1;
}
```

Rozdíly vůči jazyku BASH

```
if prikaz1; then
      prikaz2
else
      prikaz3
fi
```

Logické operátory

Operátory:

`==`	rovná se
`!=`	nerovná se
`<`	menší než
`<=`	menší než nebo rovno
`>`	větší než
`>=`	větší než nebo rovno
`!`	negace
`&&`	logické ano
`||`	logické nebo

Příklady:

```
j > 5
(j > 5)   && (j < 10)
(j <= 5)   || (j >= 10)
```

Cykly

```
for(inicializace; podminka; zmena) {
      prikaz1;
      ...
}
```

Příklad:

```
for(I=1;I <= 10;I++){
      sum = sum + $I;
}
```

Rozdíly vůči jazyku BASH

```
for((inicializace;podminka;zmena)); do
      prikaz1
done
```

Cvičení

1. Napište skript, který vytiskne největší a nejmenší hodnotu ze třetího sloupce souboru **matice.txt**.
2. Napište skript, který vytiskne ze souboru **rst.out** řádky, které obsahují devět slov.
3. Napište skript, který sečte hodnoty všech čísel uvedených v souboru **matice.txt**.