

# ***C2110 Operační systém UNIX a základy programování***

**11. lekce / modul 1**

**PS/2021 Prezenční forma výuky: Rev7**

**Petr Kulhánek**

[kulhanek@chemi.muni.cz](mailto:kulhanek@chemi.muni.cz)

Národní centrum pro výzkum biomolekul, Přírodovědecká fakulta  
Masarykova univerzita, Kamenice 5, CZ-62500 Brno

## ➤ AWK

- K čemu slouží jazyk AWK?
- Struktura skriptu, průběh vykonávání
- Struktura bloku, regulární výrazy, spuštění skriptů

# AWK

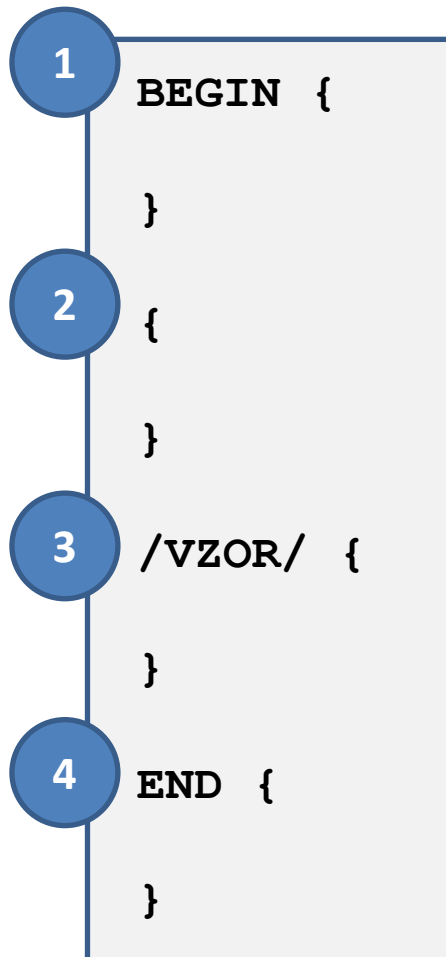
---

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

AWK je skriptovací jazyk navržený pro **zpracování 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](http://www.wikipedia.org)

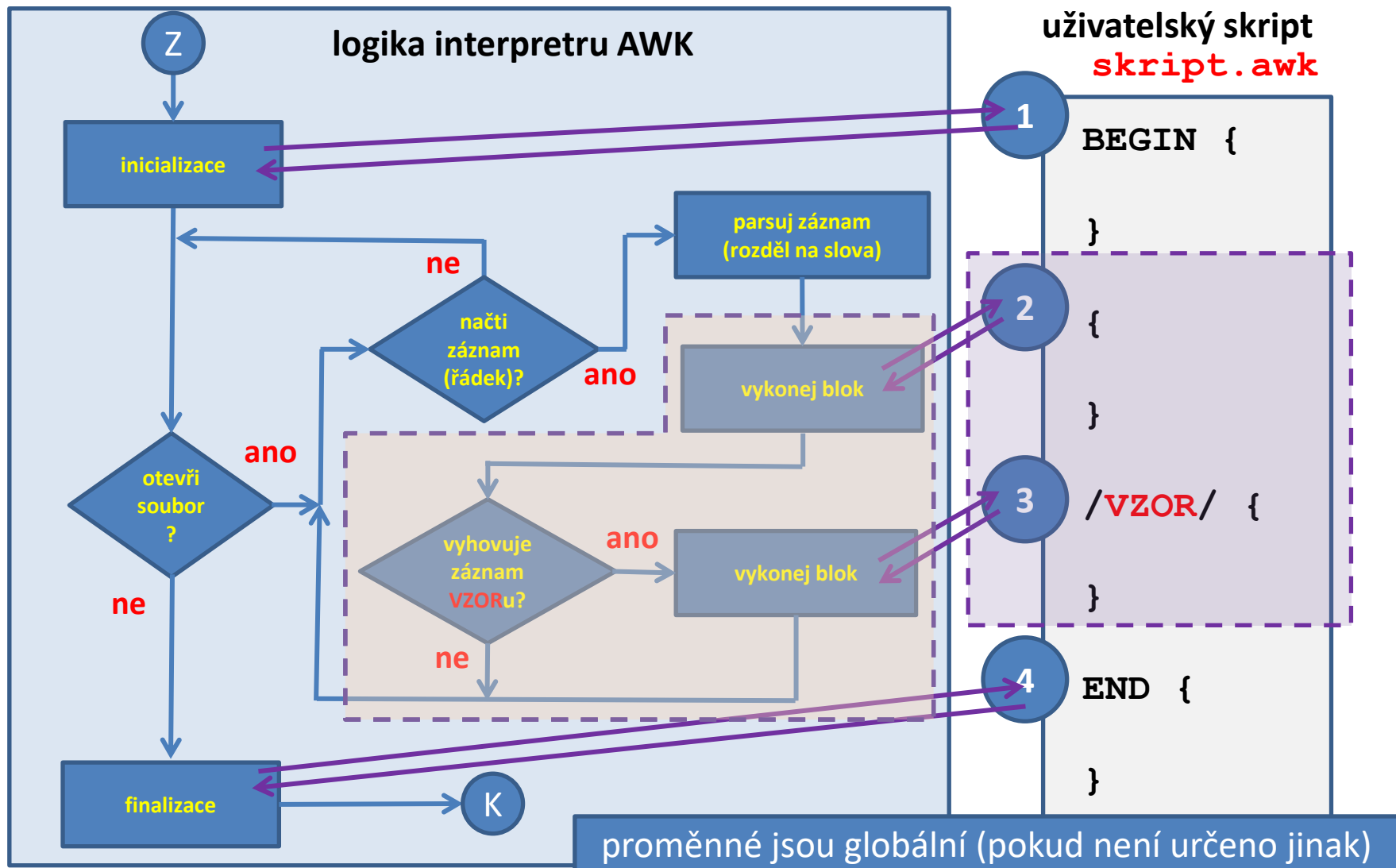
# Průběh vykonávání skriptu



- 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 VZORu, vykoná se blok (3).
  - .... vykonají se případně další bloky ....
- Blok END (4) se vykoná (pokud je ve skriptu obsažen) po analýze celého souboru.

# Průběh vykonávání skriptu

```
awk -f skript.awk soubor1.txt soubor2.txt
```



# Analýza textových souborů

54.7332	295.7275	128.4090	-508.1302	-155.6037	0.0000
51.3204	292.3619	176.5980	-494.7423	-164.7991	0.1822
40.6154	273.9238	164.5827	-488.9232	-163.0629	0.3793
52.5044	281.5944	153.4570	-484.6533	-168.5328	0.3528
62.5486	294.2701	155.3607	-483.6872	-169.1747	0.0033

Potential function:

ntf	=	2,	ntb	=	0,	igb	=	5,	nsnb	=	25
ipol	=	0,	gbsa	=	0,	iesp	=	0			
dielc	=	1.00000,	cut	=	999.00000,	intdiel	=	1.00000			

# Analýza textových souborů

záznam

pole záznamu

54.7332	295.7275	128.4090	-508.1302	-155.6037	0.0000
51.3204	292.3619	176.5980	-494.7423	-164.7991	0.1822
40.6154	273.9238	164.5827	-488.9232	-163.0629	0.3793
52.5044	281.5944	153.4570	-484.6533	-168.5328	0.3528
62.5486	294.2701	155.3607	-483.6872	-169.1747	0.0033

pole záznamu

záznam

```
Potential function:  
ntf = 2, ntb = 0, igb = 5, nsnb = 25  
ipol = 0, gbsa = 0, iesp = 0  
dielc = 1.00000, cut = 999.00000, intdiel = 1.00000
```

# Analýza textových souborů

54.7332	295.7275	128.4090	-508.1302	-155.6037	0.0000
51.3204	292.3619	176.5980	-494.7423	-164.7991	0.1822
40.6154	273.9238	164.5827	-488.9232	-163.0629	0.3793
52.5044	281.5944	153.4570	-484.6533	-168.5328	0.3528
62.5486	294.2701	155.3607	-483.6872	-169.1747	0.0033

Potential function:

ntf = 2, ntb = 0, igb = 5, nsnb = 25  
ipol = 0, gbsa = 0, iesp = 0  
dielc = 1.00000, cut = 999.00000, intdiel = 1.00000



# Ukázka

vstup.txt

54.7332	295.7275	128.4090	-508.1302	-155.6037	0.0000
51.3204	292.3619	176.5980	-494.7423	-164.7991	0.1822
40.6154	273.9238	164.5827	-488.9232	-163.0629	0.3793
52.5044	281.5944	153.4570	-484.6533	-168.5328	0.3528
62.5486	294.2701	155.3607	-483.6872	-169.1747	0.0033

script.awk

```
{  
    print $2;  
}
```

jeden jednoduchý blok

\$ awk -f script.awk vstup.txt

nebo

\$ awk '{ print \$2; }' vstup.txt

295.7275  
292.3619  
273.9238  
281.5944  
294.2701

# Struktura bloku, příklad

```
# blok pocita mezisoucet druheho sloupce
# a mezisoucet ctvrteho sloupce, pokud je ve tretim
# sloupci hodnota 5
{
    # toto je komentar
    f = f + $2; # tady pocitam mezisoucet
    printf("Mezisoucet je %10.3f\n",f);
    if( $3 == 5 ) {
        k = k + $4; # mezisoucet pro ctvrty sloupec
    }
}
# blok pocita kumulativní sumu teploty (paty sloupec)
# na radcich obsahujících klicove slovo "TEMP"
/TEMP/ {
    temp = temp + $5;
}
```

- komentáře jsou uvozeny znakem #
- příkazy se uvádějí na samostatné řádky, které je vhodné ukončit středníkem
- středník je nutné použít, pokud uvádíme dva a více příkazů na jeden řádek

# VZOR - Regulární výrazy

```
/VZOR/ {  
  
}
```

Pokud záznamu vyhovuje VZOR, tak se blok vykoná.

Vzor je **regulární výraz**.

**Regulární výraz** je jazyk, který popisuje strukturu textového řetězce. Jazyk se využívá k vyhledávání textových řetězců, k nahrazování části řetězců.

**Příklady jednoduchých regulárních výrazů:**

- TEXT** - je splněno, pokud je v daném záznamu obsažen TEXT (může být kdekoliv)
- ^TEXT** - je splněno, pokud je v daném záznamu obsažen TEXT na začátku
- TEXT\$** - je splněno, pokud je v daném záznamu obsažen TEXT na konci

# Spouštění AWK skriptů

Zpracování textového souboru:

Nepřímé spouštění:

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

↑  
interpret jazyka

↑  
awk skript

←  
analyzovaný textový soubor

←  
výsledek je tištěn na obrazovku

Analyzovaná data lze zaslat přes standardní vstup:

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

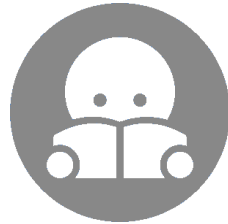
```
$ cat soubor.txt | awk -f script.awk
```

# Cvičení 1

1. Vytvořte adresář awk-data.
2. Do adresáře awk-data zkopírujte soubory matice.txt, produkt.log a rst.out z adresáře /home/kulhanek/Documents/C2110/Lesson10.
3. Napište skript, který vytiskne druhý sloupec ze souboru matice.txt.
4. Napište skript, který vytiskne druhý a čtvrtý sloupec ze souboru matice.txt.

# Samostudium

---



# 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;
}
```

**Tento způsob spouštění nedoporučuji používat.**