Základy zpracování dat pomocí AWK (pokračování)

Tomáš Kühr

KATEDRA INFORMATIKY
UNIVERZITA PALACKÉHO V OLOMOUCI

Větvení v AWK

- pomocí příkazu if-else
- lze libovolně zanořovat
- část else není povinná
- příklady:

```
awk '{if ($1 >= $2*50) print $3}' data.txt

awk '{if ($1 < 0) print "zaporne";
else if ($1 == 0) print "nula";
else print "kladne"}' data.txt
```

Cykly v AWK

For

- obvykle cyklus s předem daným počtem opakování
- v závorce inicializace řídící proměnné, podmínka a výraz pro úpravu řídící proměnné před další iterací
- podmínka se testuje před začátkem každého průchodu cyklu
- pokud je podmínka splněná, pokračuje se dalším průchodem
- jinak se pokračuje příkazy za cyklem
- příklad:

```
BEGIN{
   cislo = 5
   for (i = 1; i <= 10; i++){
      print i * cislo
   }
}
```

Cykly v AWK

While

- cyklus s podmínkou testovanou na začátku cyklu
- pokud je podmínka splněná, pokračuje se dalším průchodem
- jinak se pokračuje příkazy za cyklem
- příklad:

```
BEGIN{
   cislo = 5
   i = 1
   while (i <= 10) {
      print i * cislo
      i++
   }
}
```

Cykly v AWK

Do-while

- cyklus s podmínkou testovanou na konci cyklu
- pokud je podmínka splněná, pokračuje se dalším průchodem
- jinak se pokračuje příkazy za cyklem
- příklad:

```
BEGIN{
   cislo = 5; i = 1
   do {
      print i * cislo
      i++
   } while (i < 10)
}
```

Přerušení cyklu

- hodí se u složitějších cyklů
- okamžité přerušení cyklu (příkaz break)
- přechod k dalšímu průchodu cyklu (příkaz continue)

Pole v AWK

- na rozdíl od jiných jazyků není třeba pole předem definovat
- „indexy“ prvků mohou být libovolná čísla i řetězce
- vložení prvku do pole:
  `pole[index] = prvek`
- přečtení prvku z pole:
  `pole[index]`
- odstranění prvku z pole:
  `delete pole[index]`
- mohou být i vícerozměrná
- příklad:

```
{
   pocet[$1]++
}
END{
   for (i=-3; i<=3; i++)
      print i, pocet[i]
}
```

Vestavěné funkce v AWK

Aritmetické

- atan2, cos, exp, int, log, rand, sin, sqrt, srand

Práce s řetězci

- asort, asorti, gsub, index, length, match, split, printf, strtonum, sub, substr, tolower, toupper

Práce s časem

- systime, mktime, strftime,

A mnoho dalších

- close, exit, fflush, getline, next, system

Vlastní funkce v AWK

Definice funkce

- obvykle na začátku skriptu, před použitím
- definice obecně:

```
function function_name(argument1, argument2, ...) {
   function body
}
```

- příklad:

```
# Returns minimum number
function find_min(num1, num2){
   if (num1 < num2) return num1
   return num2
}
```

Vlastní funkce v AWK

Volání funkce

- kdekoli v jiných funkcích i blocích zpracovávající řádky či blocích BEGIN a END
- volání obecně:

```
function_name(argument1, argument2, ...)
```

- příklad použití funkce:

```
BEGIN {
   # Find minimum number
   result = find_min(10, 20)
   print "Minimum =", result
}
```

Bodované úlohy

1. Implementujte v awk převrácení tabulkových dat (sloupce oddělené mezerami) podle hlavní diagonály, tj. výměnu řádků a sloupců. (1 body)
2. Pomocí awk vypočítejte pro zadaný soubor s čísly (1 sloupec) jejich průměr a směrodatnou odchylku. (2 body)