Chapter 5

Input and Output

Overview

- Output
- Input
- Passing Parameters Into awk Script

Lesson: Output

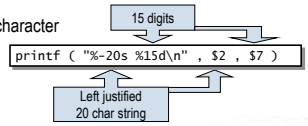
- Formatted output
- Output Into Files
- Output Into pipes

Formatted Output

- **Syntax:**

```
printf ( "format", expression ...)
```

- **When printing formatted text, the following can be used:**
  - Flag character
  - Field width
  - Precision
  - Conversion character



Output Into Files

- **The results of `print` and `printf` commands can be redirected**
  - `>` - write to file
  - `>>` - append to file

```
...
if (FNR < 2)
  print "processing file", FILENAME >> debug.out
...
```

Output Into Pipe

- **Instead of redirection, the output can also piped**

```
cat /etc/passwd | awk 'BEGIN {FS = ":"
print " User UID Real Name Home Dir Login Shell"}
$3 > 100 {
 printf ("%-9s%-6d%-15s%-15s%-15s\n",$1,$3,$5,$6,$7)
   |"sort -rn -k2"}'
```

- **There can only be one pipe open at a time; if another pipe is desired, you must close the old one first**

```
close ("sort -rn -k2")
```

- **You can use variable contents instead of literal strings**

Lesson: Input

- Input Separators
- Multiline Records
- The `getline` Function

Input Separators

- The variable FS contains input field separator. By default, the value is " ".
- The value can be any extended regular expression.
- The value can be changed from inside script or from command line:

```
$ awk -F: '$3 > 100 {$2 = "hidden"
                     print NR "\t" $0}
' /etc/passwd
```

```
$ awk ' BEGIN { FS=","
                OFS="\t"
}
{ print $1,$2,$3 }' contacts
```

Multiline Records

- The variable RS contains input record separator. By default, the value is „\n“.
- The value can be any single character.

```
$ echo $PATH | awk 'BEGIN {RS=":"} ; {print}'
```

- If the value is null, then awk will separate records based on blank lines.

```
BEGIN {RS= ""; FS= "\n"}
```

The `getline` Function

- **Syntax:**

```
getline [variable]
```

  - Next record form input is read. If variable name is specified, record is read into variable.
- **`getline` can read also from redirected input or from pipe**

```
$ awk ' BEGIN {print "Current Mount Table"
  while ( "mount -p" | getline ) {
    entries++
    print $1,"is mounted onto:",$2,"as type",$3
  }
  print entries, "entries in the mount table"
}'
```

Lesson: Passing Parameters Into awk Script

- **Any variable can be assigned value with `-v` command line option:**

```
$ awk -v OFS=' . ' -v RS=$myRS -f scriptfile filex
```

- **Environment variables can be accessed through array ENVIRON**

```
$ awk 'BEGIN { print "Hello,", ENVIRON["LOGNAME"]}'
```

- **There are also ARGV array and ARGC variable available during BEGIN action:**

```
$ awk 'BEGIN {print ARGC," Args: „
    print ARGV[0], ARGV[1], ARGV[2]}' abc def
3 Args:
awk abc def
```

Review Exercises

- **Complete the exercises from the Learning Guide**

Topics for Review