

A. TUJUAN

1. Menggunakan perintah **awk** untuk memanipulasi data dalam sebuah file.
2. Menggunakan program/script **awk** untuk mengambil nilai parameter posisi pada shell.
3. Menggunakan program/script **awk** untuk operasi aritmatika.

B. DASAR TEORI

Filter File Teks awk

Awk adalah sebuah perograman filter untuk teks, seperti halnya perintah "*grep*". Awk dapat digunakan untuk mencari bentuk/model dalam sebuah file teks. Awk juga dapat mengganti bentuk satu teks ke dalam bentuk teks lain. Awk dapat juga digunakan untuk melakukan proses aritmatika seperti yang dilakukan oleh perintah "*expr*". Awk adalah sebuah pemrograman seperti pada shell atau C yang memiliki karakteristik yaitu sebagai tool yang cocok untuk *jobs* juga sebagai pelengkap (complicated) untuk filter standard. Jika kecepatan merupakan hal yang pokok, awk adalah sesuatu produk bahasa yang sangat sesuai. Tetapi hanya untuk even jobs dimana kecepatan menjadi sesuatu hal yang esensial/pokok atau utama, maka awk dapat digunakan sebagai bentuk prototipe-nya. Meskipun awk dan shell keduanya adalah bahasa pemrograman, keduanya adalah spesialis untuk aplikasi yang berbeda. Awk sangat baik untuk manipulasi file teks, sedangkan shell sangat baik untuk pelaksana perintah UNIX.

Awk sangat cocok untuk menyelesaikan tugas sederhana, seperti halnya penyuntingan sebuah kata untuk dicetak oleh perintah `date` atau tugas kompleks lainnya seperti pelayanan pengolahan sebuah data base sederhana atau paket statistik. Adapun contoh awk untuk aplikasi sederhana yaitu sebuah program untuk menampilkan hari dan bulan, untuk lebih jelasnya perhatikan perintah-perintah berikut.

```
$ cat daymonth
date | awk '{print $3}'
$ date
$ daymonth
```

Simple Script

Sebuah awk script terdiri dari 2 bagian; yaitu "*pattern*" dan "*action*". Pattern adalah serupa dengan regular expressions, dan action sama seperti bahasa C yaitu aksi dari hasil sebuah perintah. Pattern dengan regular ekspresi diperlihatkan pada tabel dibawah ini.

Tabel 1. Pattern awk

Metacharacter	Meaning
. (Dot)	Match any character
*	Match zero or more character
^	Match beginning of line
\$	Match end of line
\	Escape character following
[]	List
{ }	Match range of instance
+	Match one more preceding
?	Match zero or one preceding
 	Separate choices to match

Sedangkan sintak umum dari script awk adalah sebagai berikut:

Sintak: ***awk -f {awk program file} filename***

Pernyataan didalam program awk:

```
Pattern {  
    action 1  
    action 2  
    action N  
}
```

Didalam program awk, pattern diikuti oleh action, dengan tanda kurung kurawal { } yang mengelilingi setiap action. Sebuah action adalah pelaksana jika pattern sesuai dengan sesuatu yang ada didalam baris input. Jika bagian pattern tidak ada, action akan melakukan aksi untuk tiap-tiap baris dalam file. Jika action absen/tidak ada, maka default action yaitu mencetak baris dilaksanakan. Bentuk pattern "paris" didalam standard input diperlihatkan pada contoh berikut:

```
$ cat findparis  
awk /paris/  
$ echo an american in Georgia | findparis  
$ echo an american in paris | findparis
```

Diperlihatkan pada contoh program diatas, terdiri dari sebuah pattern */paris/* dan tidak ada action. Ekuivalen program dengan menggunakan default action (print) seperti contoh dibawah ini.

```
$ cat findparis  
awk '/paris/ {print}'  
$ echo an american in Georgia | findparis  
$ echo an american in paris | findparis
```

Tanda kutip didaerah sekitar awk script merupakan suatu kebutuhan, sebab script akan bernilai kosong jika tanpa tanda kutip. Salah satu tanda kutip tunggal atau ganda sudah mencukupi untuk contoh diatas. Perintah print didalam program "findparis" untuk mencetak seluruh baris mungkin juga bisa untuk mencetak konstanta, karakter dari baris input, dan juga mencetak argument, disamping itu pada program ini tidak memiliki parameter.

Sebuah contoh print dengan argument dalam awk yang menggunakan parameter \$3 adalah sebagai berikut:

```
$ date | awk '{print $3}'
```

Program awk untuk mengambil argumen dari sebuah file seperti berikut ini:

```
$ cat prarg.awk
{ print $3 }
$ date | awk -f prarg.awk
```

Pada argumen flag `-f` dan menyertakan nama file yang berisi awk digunakan untuk menampilkan isi file program awk. `{ print $3 }` membuat awk membaca input dari nama file atau dari input standard (keyboard).

C. PROSEDUR

1. Masuk ke sistem operasi Linux.
2. Login sebagai stD3XXYYY.
3. Gunakan perintah awk pada pemrograman shell untuk aplikasi filter file teks dan operasi aritmatika.
4. Catatlah hasil dari setiap program pada prosedur no.3 sebagai laporan sementara.

D. PERCOBAAN

1. Mencari pattern "paris" pada sebuah file menggunakan awk tanpa action.

```
$ cat > findparis
awk /paris/
^d

$ chmod u+x findparis
$ echo an american in georgia | ./findparis
$ echo an American in paris | ./findparis
```

2. Mencari pattern "paris" pada sebuah file menggunakan awk dengan default action (print).

```
$ cat > findparis
awk '/paris/ { print }'
^d

$ echo an american in georgia | ./findparis
$ echo an American in paris | ./findparis
```

3. Mencetak informasi tanggal dengan parameter posisi (\$3) dari perintah "date" menggunakan awk.

```
$ cat > dayofmonth
date | awk '{ print $3 }'
^d
$ chmod u+x dayofmonth
$ date
$ ./dayofmonth
```

4. Menggunakan parameter, default action dan tanpa pattern dalam program awk.

```
$ cat > par.awk
{ print $3 }
^d
$ date | awk -f par.awk
```

5. Menggunakan perintah awk untuk operasi arimatika

```
$ cat > ave
awk '{
    n = n + 1
    X = X + $1
    Y = Y + $2
    Z = Z + $3
    Print X/n, Y/n, Z/n
}'
^d

$chmod u+x ave
$ cat > test
1 2 3
4 5 6
7 8 9
0 1 2
^d

$ ./ave < test ("<" input redirection)
```

6. Menggunakan perintah awk dan BEGIN_END untuk operasi arimatika

```
$ cat > ave2
awk '
BEGIN { n = 0 ; X = 0 ; Y = 0 ; Z = 0 }
{
    n = n + 1
    X = X + $1
    Y = Y + $2
    Z = Z + $3
}
END { Print X/n, Y/n, Z/n }'
```

```
^d
$ chmod u+x ave2
$ ave2 < test
```

7. Menggunakan perintah awk dengan opsi -f untuk operasi arimatika

```
$ cat > math
{
    print $1 " + " $2 " = " $1 + $2 "\t"
    print $1 " - " $2 " = " $1 - $2 "\t"
    print $1 " / " $2 " = " $1 / $2 "\t"
    print $1 " x " $2 " = " $1 * $2 "\t"
    print $1 " mod " $2 " = " $1 % $2 "\t"
}
$ awk -f math
5 2
```

8. Memilih/manipulasi data dalam sebuah file berdasarkan pattern "good" menggunakan perintah awk.

```
$ cat > pilih
egg        order    4
cacke      good     10
cheese     okay      4
pen        good     12
floppy     good      5
^d

$ awk '/good/ { print $3 }' pilih
```

9. Manipulasi data dalam sebuah file dan operasi aritmatika dengan awk.

```
$ cat > invent
1. Pen      5    20.00
2. Pencil  10    2.00
3. Rubber   3    3.50
4. Cock     2   45.50
^d

$ awk '{ print $1 $2 "--> Rp." $3 * $4 }' invent
$ awk '{ print $2 }' invent
$ awk '{ print $2 $4}' invent
$ awk '{ print $0 }' invent
$ cat > prn_pen
/Pen/ { print $3 }
^d

$ awk -f prn_pen invent
```

10. Manipulasi data dalam sebuah file dan mencetak dengan awk.

```

$ cat > progsh
BEGIN {
    printf "Program Shell awk 2007.\n"
    printf "Lab. Jarkom PENS.\n"
    printf "-----\n"
}

{
    total = $3 * $4
    recno = $1
    item = $2
    gtotal += total
    printf "%d %s Rs.%f\n", recno, item, total
}

END {
    printf "-----\n"
    printf "Total Rs. %f\n", gtotal
    printf "=====\n"
}
$ awk -f progsh invent

```

11. Menampilkan daftar user pada system menggunakan awk

```

$ vi awk1.sh
#!/bin/bash
PASSWORD_FILE=/etc/passwd
n=1
for name in $(awk 'BEGIN{FS=":"}{print $1}' <
"$PASSWORD_FILE" )
do
    echo "USER #$n = $name"
    let "n += 1"
done
exit 0
$ ./awk1.sh

```

12. Penggunaan perintah awk untuk mengetahui informasi file

```

$ vi awk2.sh
#!/bin/bash
FILES="/usr/sbin/accept
/usr/sbin/pwck
/usr/sbin/chroot
/usr/bin/fakefile
/sbin/badblocks
/sbin/ypbind"
echo
for file in $FILES
do
    if [ ! -e "$file" ]

```

```

then
    echo "$file does not exist."; echo
    continue                                # On to next.
fi

ls -l $file | awk '{ print $9 "          file size:
" $5 }'
whatis `basename $file`
echo
done
exit 0

$ ./awk2.sh

```

E. Latihan

1. Diketahui tabel karakter ASCII dengan data dimulai dari 33 kode desimal hingga 125 seperti dibawah ini.

Desimal	Hex	Character
----	---	-----
33	21	!
34	22	"
35	23	#
.	.	.
.	.	.
.	.	.
125	7d	}

Buatlah program shell untuk menampilkan tabel karakter ASCII seperti diatas dengan awk.

2. Buatlah program shell untuk membangkitkan bilangan random menggunakan awk dengan banyak data tiap barisnya adalah 25 data.
3. Buatlah program shell untuk membangkitkan bilangan pseudo random diantara 0 sampai dengan 1 menggunakan awk.
4. Buatlah program shell dengan awk untuk menghitung sisi miring dari sebuah segitiga (c) seperti ditunjukkan pada gambar dibawah, jika nilai kedua sisi-sisi yang lain (misal: 3 dan 5) disertakan ketika script dijalankan.

