AWK THAM KHẢO NHANH

Pattern, field, array, hàm, xử lý văn bản

Cơ Bản

Chạy AWK

```
awk '{ print }' file.txt          # in mọi dòng
awk '{ print $1 }' file.txt       # in field đầu tiên
awk -F: '{ print $1 }' /etc/passwd  # dấu phân tách tùy chỉnh
awk -f script.awk file.txt        # chạy từ file
cmd | awk '{ print $2 }'          # đầu vào qua pipe
```

Cấu Trúc Chương Trình

`awk 'pattern { action }'`	Dạng cơ bản — action chạy khi pattern khớp
`BEGIN { ... }`	Chạy một lần trước khi xử lý đầu vào
`END { ... }`	Chạy một lần sau khi xử lý xong tất cả đầu vào
`Không có pattern`	Action chạy cho mỗi dòng
`Không có action`	Action mặc định là `{ print }`

Pattern & Action

Các Loại Pattern

```
awk '/error/' file.txt            # khớp regex
awk '$3 > 100' file.txt           # so sánh
awk 'NR >= 5 && NR <= 10' file.txt  # phạm vi dòng
awk '/start/,/end/' file.txt      # pattern phạm vi
```

Tham Khảo Pattern

`/regex/`	Khớp dòng với regex
`$1 ~ /pat/`	Field khớp regex
`$1 !~ /pat/`	Field không khớp regex
`expr1, expr2`	Phạm vi: từ lần khớp đầu đến lần khớp thứ hai
`expr1 && expr2`	AND logic
`expr1 || expr2`	OR logic
`!expr`	NOT logic

Biến

Biến Built-in

`NR`	Số record (dòng) hiện tại
`NF`	Số field trong record hiện tại
`FS`	Dấu phân tách field đầu vào (mặc định: khoảng trắng)
`OFS`	Dấu phân tách field đầu ra (mặc định: dấu cách)
`RS`	Dấu phân tách record đầu vào (mặc định: newline)
`ORS`	Dấu phân tách record đầu ra (mặc định: newline)
`FILENAME`	Tên file đầu vào hiện tại
`FNR`	Số record trong file hiện tại

Biến Người Dùng

```
awk '{ total += $1 } END { print total }' file.txt
awk -v threshold=50 '$1 > threshold' file.txt
awk 'BEGIN { count = 0 } /pat/ { count++ }
     END { print count }' file.txt
```

Fields

Truy Cập Field

`$0`	Toàn bộ dòng hiện tại
`$1, $2, ...`	Field thứ nhất, thứ hai, ...
`$NF`	Field cuối cùng
`$(NF-1)`	Field kế trước cuối

Dấu Phân Tách Field

```
awk -F, '{ print $2 }' data.csv     # dấu phẩy
awk -F'\t' '{ print $1 }' data.tsv  # tab
awk 'BEGIN { FS = "[,:]" } { print $1 }' f  # nhiều ký tự
awk 'BEGIN { OFS = "," } { print $1, $3 }' f  # dấu phân tách đầu ra
```

Luồng Điều Khiển

Điều Kiện & Vòng Lặp

```
awk '{ if ($1 > 50) print "high"; else print "low" }' f
awk '{ for (i = 1; i <= NF; i++) print $i }' f
awk '{ i = 1; while (i <= NF) { print $i; i++ } }' f
awk '/skip/ { next } { print }' f  # bỏ qua dòng khớp
```

Câu Lệnh Điều Khiển

`if (cond) { ... } else { ... }`	Điều kiện
`for (i = 0; i < n; i++) { ... }`	Vòng lặp for kiểu C
`for (key in array) { ... }`	Duyệt key của array
`while (cond) { ... }`	Vòng lặp while
`do { ... } while (cond)`	Vòng lặp do-while
`next`	Bỏ qua, chuyển sang record tiếp theo
`exit`	Dừng xử lý, chạy khối END

Hàm

Hàm Tự Định Nghĩa

```
awk 'function max(a, b) {
    return (a > b) ? a : b
}
{ print max($1, $2) }' file.txt
```

Hàm Số

`int(x)`	Cắt bỏ phần thập phân
`sqrt(x)`	Căn bậc hai
`sin(x), cos(x)`	Hàm lượng giác
`log(x), exp(x)`	Logarithm tự nhiên và hàm mũ
`rand()`	Số thực ngẫu nhiên trong khoảng 0 đến 1
`srand(seed)`	Khởi tạo seed cho bộ tạo số ngẫu nhiên

Arrays

Associative Arrays

```
awk '{ count[$1]++ }
     END { for (k in count) print k, count[k] }' f
awk '{ arr[NR] = $0 }
     END { for (i = NR; i >= 1; i--) print arr[i] }' f
```

Thao Tác Array

`arr[key] = val`	Đặt phần tử
`arr[key]`	Lấy phần tử (tự tạo khi truy cập)
`key in arr`	Kiểm tra key có tồn tại không
`delete arr[key]`	Xóa một phần tử
`delete arr`	Xóa toàn bộ array
`for (k in arr)`	Duyệt qua các key (không theo thứ tự)
`length(arr)`	Số phần tử (gawk)

Hàm Chuỗi

Tham Khảo Chuỗi

`length(s)`	Độ dài chuỗi
`substr(s, start, len)`	Chuỗi con (chỉ số bắt đầu từ 1)
`index(s, target)`	Vị trí của target trong s (0 nếu không tìm thấy)
`split(s, arr, sep)`	Tách chuỗi thành array
`sub(pat, repl, s)`	Thay thế lần khớp đầu tiên
`gsub(pat, repl, s)`	Thay thế tất cả lần khớp
`match(s, pat)`	Vị trí khớp regex (đặt RSTART, RLENGTH)
`tolower(s) / toupper(s)`	Chuyển đổi chữ hoa/thường
`sprintf(fmt, ...)`	Định dạng chuỗi (như C printf)

Ví Dụ Chuỗi

```
awk '{ gsub(/old/, "new"); print }' f    # thay thế kiểu sed
awk '{ print toupper($0) }' f             # viết hoa tất cả
awk '{ print substr($0, 1, 40) }' f       # cắt bớt 40 ký tự
```

I/O

Đầu Ra

```
awk '{ print $1, $2 }' f            # cách nhau bởi dấu cách
awk '{ printf "%s,%d\n", $1, $2 }' f  # đầu ra có định dạng
awk '{ print $1 > "out.txt" }' f    # chuyển hướng ra file
awk '{ print $1 >> "out.txt" }' f   # thêm vào file
```

Tham Khảo I/O

`print`	In với ORS (newline mặc định)
`printf fmt, ...`	In có định dạng (không có newline cuối)
`print > file`	Chuyển hướng đầu ra ra file
`print >> file`	Thêm vào cuối file
`print | cmd`	Pipe đầu ra vào lệnh
`getline < file`	Đọc một dòng từ file
`cmd | getline var`	Đọc đầu ra lệnh vào biến
`close(file)`	Đóng file hoặc pipe

Các Pattern Thường Gặp

One-Liners

```
awk '{ sum += $1 } END { print sum }' f   # tổng cột
awk 'END { print NR }' f                   # đếm dòng
awk '!seen[$0]++' f                        # bỏ trùng lặp
awk 'NF' f                                 # bỏ dòng trống
awk '{ print NF }' f                       # số field mỗi dòng
```

Công Thức

`CSV to TSV`	`` `awk -F, 'BEGIN{OFS="\t"}{$1=$1; print}'` ``
`Tổng cột 2`	`` `awk '{ s += $2 } END { print s }'` ``
`Top N dòng`	`` `awk 'NR <= 10'` `` (như head)
`Đếm tần suất`	`` `awk '{ c[$1]++ } END { for (k in c) print k, c[k] }'` ``
`Giữa các marker`	`` `awk '/BEGIN/,/END/'` ``
`In field thứ N`	`` `awk '{ print $N }'` `` (thay N)