AWK

By Mohsen Mahdavifar

گردآوری شده توسط محسن مهدوی فر

Laitec.ir

(Mahdavifar@laitec.ir)

یکی از پر کاربردترین و پرقدرت ترین دستوراتی که توانایی جستجو ، ادیت ، جانشین کردن لغات در فایلهای متنی و حتی توانایی نوشتن دستورات شرطی و لوپ را نیز در داخل دستور داراست دستور awk می باشد .

نحوه نگارش دستور

```
awk 'pattern {action}' input-file
```

awk هر خط از فایل را مورد بازرسی و یا بازبینی قرار می دهد که هر کلمه در فایل به یک متغیر نسبت داده می شود که با علامت $ شروع می شود .

به طور مثال $1 نمایانگر اولین لغت $2 دومین لغت و $3 سومین لغت و به همین ترتیب.

نکته :

$0 به معنای کل یک خط می باشد .

مثالی ساده برای آشنایی با این دستور :

فایل information حاوی خط های زیر می باشد

```
Mohsen Mahdavifar OpenLdap

Mohammad Ardaneh Squid

Vahab SHlachian DNS
```

حال به بررسی دستوری ساده با awk می پردازیم

```
awk '{print "Aghaye",$1,"With Surename",$2,"Kargahe",$3}' information
```

خروجی به صورت زیر می باشد

Aghaye Mohsen With Surename Mahdavifar Kargahe Openldap

Aghaye Mohamamd With Surename Ardaneh Kargahe Squid

Aghaye Vahab With Surename Shalchian Kargahe DNS

در نظر بگیرید که بعضی از فیلدهای موجود در فایل عددی باشد

به طور مثال فایل Price را که حاوی اطلاعات پرداختی در ۳ ماه مختلف است را در نظر بگیرید

Farvardin 600 700 800

Ordibehesht 800 900 1300

Khordad 500 300 700

شما می توانید میانگین پرداختی هرماه را توسط دستور زیر حساب کنید

```
awk '{print "Avg for",$1,"is",($2+$3+$4)/3}' Price
```

```
Avg for Farvardin is 460
Avg for Ordibehesht is 1000
Avg for Khordad is 500
```

فایل price قبلی را در نظر بگیرید با زدن دستور

```
awk /Ordi/ ' {print "Third Field is",$3} ' Price
```

و یا

```
awk ' /Ordi/{print "Third Field is",$3}' Price
```

دستور بالا pattern مورد نظر که در اینجا Ordi است را در فایل Search کرده و فیلد سوم خطی را که pattern داده شده را دارد برای ما چاپ می کند .

```
The Third Field is 900
```

نکته : دقت شود بین /Ordi/ و عبارت print{' هیچ گونه فاصله ای نباید وجود نداشته باشد.

مثالهای پیشرفته تر از awk

نحوه نگارش دستور

```
awk 'BEGIN {start_action} {action} END {stop_action}' filename
```

مثال :

یک فایل با نام test همانند فایل زیر با زدن دستور ls -l > test ایجاد کنید

```
drwxr-xr-x  8 root root    4096 Nov 25 05:35 X11
drwxr-xr-x  4 root root    4096 Nov 25 05:25 xdg
drwxr-xr-x  2 root root    4096 Nov 25 05:22 xinetd.d
drwxr-xr-x  2 root root    4096 Nov 25 05:22 xml
```

```
-rw-r--r--  1 root root     585 Sep 21  2009 yp.conf
drwxr-xr-x  3 root root    4096 Nov 25 05:28 yum
-rw-r--r--  1 root root     346 Apr  4  2010 yum.conf
```

از اطلاعات فوق مشاهده می شود که فایل شامل تعدادی سطر و ستون می باشد که هر خط با کاراکتر \n یا به عبارتی newline از هم جدا شده اند و ستون ها نیز توسط کاراکتر space از یکدیگر جدا شده اند .

دستور زیر

```
awk '{print $1}' test
```

فیلد شماره ۱ از فایل تست را جدا می کند که خروجی دستور فوق همانند زیر است .

```
drwxr-xr-x
drwxr-xr-x
drwxr-xr-x
drwxr-xr-x
-rw-r--r--
drwxr-xr-x
-rw-r--r--
```

$1 نمانگر فیلد اول $2 نمانگر فیلد دوم و به همین ترتیب $3 نمانگر فیلد سوم و .. می باشد .

در مثال بالا ما بلاک شروع و پایان نداشتیم (BEGIN & END) به همین دلیل دستور print برای هر ردیف از فایل test که میخواند انجام می شود. حال میخواهیم از بلاک شروع و پایان استفاده کنیم .

دستور زیر را در نظر بگیرید

```
awk 'BEGIN {sum=0} {sum=sum+$2} END {print sum}' test
```

دستور بالا عبارت sum را برابر ۰ قرار داده و هر دفعه sum را با فیلد شماره ۲ جمع کرده و به سطر بعدی می رود و در نهایت پس از انجام دستور فوق برای تمامی سطرها مقدار sum را برای ما نمایش می دهد .

در حقیقت خروجی دستور بالا عدد ۲۱ خواهد بود .

مثال ۲ : عبارت زیر را برای فایل test در نظر بگیرید

```
awk '{ if($9 == "yum") print $0;}' test
```

در مثال فوق اگر مقدار فیلد شماره ۹ برابر yum بود کل خط مورد نظر چاپ خواهد شد .

خروجی دستور فوق عبارت زیر خواهد بود

```
drwxr-xr-x  3 root root    4096 Nov 25 05:28 yum
```

```
awk 'pattern {action}' input-file
```

معنی دستور بالا :

خط به خط فایل ورودی که در اینجا input-file است را بررسی کرده و اگر خطی شامل pattern مورد نظر باشد action مورد نظر را برای آن خط انجام داده و نتیجه action را برای ما چاپ می کند .

اگر patten از دستور فوق حذف گردد action مورد نظر برای تمامی خطوط اجرا خواهد شد.

```
awk '{ print $5 }' laitec.txt > output1.txt
```

دستور فوق عنصر پنجم سطر هر خط را از فایل laitec.txt جدا کرده و در یک خط جدید در فایل output می نویسد .

همانطور که قبل تر بحث شد $3 به معنای عنصر سوم از هر سطر و $2 و $1 به ترتیب به معنای عنصر دوم از هر سطر و $1 به معنای عنصر اول از هر سطر می باشد .

به صورت دیفالت عناصر هر سطر با space و یا tab از یکدیگر جدا شده اند که ما آن را space خطاب می کنیم و رفتار پیش فرض awk نیز بدین صورت است .

حال فایل laitec.txt را در نظر بگیرید که حاوی خطوط زیر است .

```
1, Mohsen Mahdavifar, Laitec.ir, Openldap, 60 Hour
2, Mohammad Ardaneh, laitec.ir, Squid, 50 Hour


3, Vahab Shalchina, Laitec.ir, Bind, 40 Hour
4, Behrouz Ebadi, Laitec.ir, Master, - Hour
```

با زدن دستور رو به رو

```
awk '{ print $5 }' laitec.txt
```

خروجی عبارت فوق به صورت زیر خواهد بود

```
,Openldap

,Squid

,Bind

,Master
```

هر سطر از فایل laitec.txt در صورتی که space را به عنوان جدا کننده فیلدها را از یکدیگر در نظر بگیریم شامل ۷ فیلد است که فیلد شماره ۵ آن را توسط دستور بالا ما جدا کرده ایم . دقت شود که دستور awk نیز به صورت پیش فرض جدا کننده فیلدها از یکدیگر را space یا tab که ما هر دو را space خطاب میکنیم در نظر می گیرد.

اگر جدا کننده فیلدها چیزی به غیر از space یا tab می بود به عنوان مثال , و یا : و یا ; و یا مواردی از این قبیل ما میتوانیم برای دستور awk جدا کننده فیلد ها و یا Field Separator را مشخص کنیم .

در فایل laitec.txt همانطور که مشخص است Filed Separator به جای space و یا tab کاما (,) می باشد . اگر ما اسامی موجود در فایل را نیاز داشته باشیم در صورتی که Field Separator را کاما (,) فرض کنیم فیلد شماره ۲ می شود . با زدن دستور زیر ما فیلد دومی از فایل laitec.txt را که عناصر آن با کاما (,) از یکدیگر جدا شده اند را چاپ می کنیم .

```
awk -F, '{ print $2 }' table1.txt > output1.txt
```

که خروجی دستور بالا به شرح زیر است

```
Mohsen Mahdavifar

Mohammad Ardaneh

Vahab Shalchina

Behrouz Ebadi
```

نکته :

لیست دستوراتی که در داخل کروشه {','} قرار داده می شوند block خوانده می شوند . اگر شما قبل از block یک عبارت شرطی قرار دهید دستورات و یا کارهای درون block در صورتی اجرا می شوند که عبارت شرطی قبل از آن true باشد .

نکته : توجه شود اگر از عبارت if قبل از block بخواهید استفاده کنید حتما کروشه باید قبل از if شروع شود .

مثال زیر را در نظر بگیرید

```
awk '$6=="60" { print $2 }' laitec.txt
awk '{ if ($6=="60")  print $2 }' laitec.txt
```

دو دستور فوق یکسان بوده و یک عملیت را انجام می دهند . (به محل قرار گیری کروشه در دستورات دقت شود)

عبارت بالا بدین معنی است که در صورتی که فیلد ششم از فایل مورد نظر اگر برابر۶۰ بود در آن صورت فیلد شماره ۲ را برای ما نمایش بده .

خروجی دستور بالا اسم Mohsen خواهد بود .

ما همچنین می توانیم از regular expression نیز در دستوراتمان استفاده کنیم .

مثال زیر را در نظر بگیرید

```
awk '/40/ { print $2 }' laitec.txt
```

مقداری که در بین دو slash (/) قرار می گیرد regular expression می باشد . که در این مورد فقط عدد ۴۰ می باشد .دستور فوق بدین معنی می باشد که اگر خطی از فایل laitec.txt شامل عبارت 40 بود فیلد شماره ۲ آنرا برای ما نمایش بده .

خروجی دستور بالا ۲ اسم Mohammad و Vahab خواهد بود .

نکته : اگر فیلد ها عددی باشند می توان عملیات ریاضی بر روی آنها انجام داد . به طور مثال به دستور زیر توجه کنید

```
awk '/40|50/ { print ($6*2)^3 }' laitec.txt
```

دستور فوق بدین معنی می باشد که هر خطی که شامل ۵۰ و یا ۴۰ بود فیلد ششم آنرا را در ۲ ضرب کرده و به توان ۳ برسان که خروجی همانند زیر خواهد بود

512000

1000000

متغیرهای داخلی awk

awk چندین متغیر داخلی قدرتمند دارد که در این قسمت به دو مورد از آنها اشاره می کنیم :

NF: Number of Fields in a record

NF تعداد فیلد ها در یک رکورد را سطر را به ما می دهد و یک ابزار قدرتمند برای بررسی صحت وجود تمامی فیلدها در یک سطر می باشد .

مثال: فایلی با نام Score را در نظر بگیرید که حاوی نمرات ۵ دانشجو در ۴ درس مختلف میباشد

```
Mohsen 14 12 10 18
Mohamamd 20 19 12 15
Vahab 20 20 15
Saeid 18 17 14 18
Hossein 16 16 17
```

برای صحت از اینکه تمامی نمره های افراد وارد شده است می توان تعداد فیلد ها را بررسی کرد .

اگر تمامی نمرات وارد شده باشد تعداد فیلدها می بایست در هر سطر ۵ عدد باشد مطمئن خواهیم شد که تمامی نمرات وارد شده اند .

```
awk ' ( NF != 5 ) { print $0 } ' Score
```

و یا

```
awk '{ if ( NF != 5 ) print $0 } ' Score
```

۲ دستور بالا یکسان بوده و خطوطی را که تعداد رکورد های آن نا مساوی ۵ است را برای ما چاپ میکند

خروجی دستور بالا به صورت زیر است :

```
Vahab 20 20 15

Hossein 16 16 17
```

همچنین $NF به معنای آخرین فیلد می باشد صرف نظر از اینکه هر سطر و یا رکورد چند فیلد دارد

به مثال زیر توجه کنید

```
awk '{print $NF}' Score
```

دستور بالا فیلد آخر هر سطر را برای ما جدا می کند صرفنظر از اینکه هر سطر چند فیلد دارد .

خروجی دستور بالا به صورت زیر است

18

15

15

18

17

مثال : در نظر بگیرید در فایل Score که شامل اسامی و نمرات بود می خواهیم بدانیم چند رکورد و یا خط وجود دارد که به تمامی نمرات برای آنها ثبت نشده است . برای این کار به صورت زیر عمل می کنیم .

```
awk 'BEGIN {sum=0}{if(NF != 5)sum++} END{print sum}' Score
```

خروجی عدد ۲ خواهد بود

NR: Number of Records Variable

NR تعداد رکوردهایی را که awk مورد پردازش قرار میدهد و یا تعداد خطوط را چاپ می کند .

NR به شما این اجازه را می دهد تا خط خاصی را مورد بررسی قرار دهید .

```
awk '{ if (NR >= 30)print NR, $0}' /etc/passwd
```

در مثال فوق خطوط بعد از ۲۹ امین خط را شماره گذاری کرده (NR) و تمامی خطوط بعد از ۲۹ را چاپ می کند ($0) .

نمونه خروجی به صورت زیر می باشد

30rpcuser:x:29:29:RPC Service User:/var/lib/nfs:/sbin/nologin
31nfsnobody:x:65534:65534:Anonymous NFS User:/var/lib/nfs:/sbin/nologin
32haldaemon:x:68:68:HAL daemon:/:/sbin/nologin
33avahi-autoipd:x:100:104:avahi-autoipd:/var/lib/avahi-autoipd:/sbin/nologin
34gdm:x:42:42::/var/gdm:/sbin/nologin
35admin:x:500:500:Server:/home/admin:/bin/bas

مثال :

```
awk '{print NR,"--->",NF}' Score
```

مثال بالا ابتدا به دلیل استفاده از **NR** شماره خط را چاپ میکند و سپس با گذاشت علامت **--->** تعداد فیلدهای موجود در آن رکورد و یا خط را چاپ می کند .

خروجی دستور بالا به صورت زیر است

1---> 5

2--->5

3--->4

4--->5

5--->4

چندین دستور ساده و کاربردی از awk

۱- همانطور که قبل تر بحث شده دستور **awk** توانایی سرچ و جستجوی یک استرینگ خاص در فایل متنی را دارد . نحوه انجام این کار به صورت زیر است :

```
awk '/laitec.ir/' file
```

دستور فوق عبارت **laitec.ir** را در یک فایل **file** جستجو می کند .

۲- برای چاپ کردن خطوط **x** تا **y** از یک فایل نیز می توان از دستور فوق استفاده کرد

```
awk 'NR==14, NR==30' file
```

دستور فوق خطوط ۱۴ تا ۳۰ از **file** را برای ما نشان می دهد .

۳- برای بدست آوردن خطوط یک فایل :

```
awk 'END{print NR}' file
```

دستور فوق مشابه دستور **wc -l** می باشد

۴- برای نمایش تمامی خطوط از یک فایل در یک سطر همراه با شماره خط آنها :

```
awk '{printf "%3d %s", NR, $0}' file
```

۵- برای جایگزینی و یا پاک کردن فیلد شماره **n** از یک فایل

```
awk '{$n="";print }' file
```

۶- بدست آوردن تعداد خطوط خالی موجود در یک فایل

```
awk '/^$/ {x++} END {print x}' file
```

۷- بدست آوردن طول هر خط و یا رکورد

```
awk '{print length($0)}' file
```