Sed

A ferramenta **sed** é un editor de texto, así permite modificar grupos de caracteres. Un dos usos máis empregado é a sustitución de caracteres.

Sumario

- 1 Sintaxe
  - 1.1 Argumentos
  - 1.2 Opcións
  - 1.3 Instrucións
  - 1.4 Estruturas de control
  - 1.5 O espazo modelo e o espazo contenedor
- 2 Utilizando *Sed*
  - 2.1 Modificar un texto por outro
  - 2.2 Concatenar varias sentenzas
  - 2.3 Filtrar liñas
    - 2.3.1 **Facer cambios nunha liña escollida pola súa posición no documento de texto:**
    - 2.3.2 Visualizar só as primeiras liñas
    - 2.3.3 Visualizar as últimas liñas
  - 2.4 Executar máis dun comando nunha liña individual
  - 2.5 Borrar liñas
  - 2.6 Insertar e engadir texto
  - 2.7 Cambiar o contido dunha liña
  - 2.8 O comando "transformar"
  - 2.9 Comandos para imprimir
  - 2.10 Usando arquivos
  - 2.11 Emprego de *program-file*
  - 2.12 Comandos multiliña
    - 2.12.1 Os comandos *next*
    - 2.12.2 Comando de borrado multiliña
    - 2.12.3 Comando de impresión multiliña
  - 2.13 O "*Hold space*"
  - 2.14 Negando un comando
  - 2.15 Cambiando o fluxo
    - 2.15.1 Derivación
    - 2.15.2 Testing
- 3 Exemplos
  - 3.1 Exemplo de cambio da configuración IP
- 4 Enlaces interesantes

Sintaxe

Unha liña de comando *sed* ten a seguinte sintaxe:

```
sed [-n] program [file-list]
sed [-n] -f program-file [file-list]
```

Argumentos

- O ***program*** é un programa *sed* que se inclúe na liña de comandos.
- Este formato nos permite escribir sinxelos e breves programas *sed* sen crear un *program-file* separado.
- O ***program-file*** é o nome de ruta dun ficheiro que conteña un programa *sed*.
- A ***file-list*** contén os nomes de ruta dos ficheiros ordinarios que procesa *sed*. Estes son os ficheiros de entrada. Cando non especificamos un ficheiro, *sed* toma a súa entrada dende a entrada estándar.

Opcións

- **-f *program-file*** fai que *sed* lea o seu programa dende o ficheiro chamado *program-file* en vez de dende a liña de comando. Podemos utilizar esta opción en mais dunha ocasión na liña de comando.

- **-i [suffix]** edita os ficheiros no seu lugar. Sen esta opción *sed* envía a súa saída á saída estándar. Con esta opción *sed* rempraza o ficheiro que está procesando coa súa saída. Cando especificamos un sufixo, *sed* fai unha copia do ficheiro orixinal. Esta copia ten o nome do ficheiro orixinal có sufixo engadido. Debemos incluír un punto no sufixo se queremos que apareza un punto entre o nome do ficheiro orixinal e o sufixo.
- **--help** resume cómo empregar *sed*.
- **--quiet** ou **-n -silent** : Fai que *sed* non copie as liñas á saída estándar a excepción das que especificou a instrución *Print(p)* ou o *flag*.

Instrucións

- **d (*delete*):** A instrución *delete* fai que *sed* non escriba as liñas que seleccione e non finalice de procesalas. Despois de executar *sed*, unha instrución *delete* le a seguinte liña de entrada e comeza de novo coa primeira instrución do *program* ou *program-file*.
- **n (*next*):** Se é axeitada, a instrución *next* escribe a liña seleccionada neste intre, le a seguinte liña de entrada e comeza a procesar a nova liña coa seguinte instrución de *program* ou *program-file*.
- **a (*append*):** A instrución *append* adxunta unha ou mais liñas á liña seleccionada neste momento. Se precedemos a instrución *append* con dúas direccións, adxunta cada liña que estea seleccionada polas direccións; as antigas versións de *sed* non aceptaban instrucións *append* con dúas direccións. Se non precedemos unha instrución *append* cunha dirección, adxúntase a cada liña de entrada.
- **i (*insert*):** A instrución *insert* é idéntica á instrución *append* a excepción de que coloca o novo texto antes da liña seleccionada.
- **c (*change*):** A instrución *change* é parecida a *append* e *insert* a excepción de que cambia as liñas seleccionadas para que conteña o novo texto. Cando especificamos un rango de dirección, *change* rempraza todo o rango de liñas cunha soa ocurrencia do novo texto.
- **s (*substitute*):** Esta instrución en *sed* ten o seguinte formato:

```
[address[,address]] s/pattern/replacement-string/[g] [p] [w file]
```

· O modelo é unha expresión regular que se delimita por calquera carácter que non sexa un espazo ou unha nova liña, normalmente é unha barra (/). A cadea de remprazo comeza inmediatamente despois do segundo delimitador e debe finalizarse có mesmo delimitador. O delimitador final (o terceiro) é necesario. A cadea de remprazo pode conter un ampersand (&), que *sed* rempraza có modelo que concorda. A menos que empreguemos o indicador **g**, a instrución *substitute* rempraza só a primeira ocurrencia do modelo en cada liña seleccionada.
· **p (*print*):** A instrución *print* escribe as liñas seleccionadas na salida estándar.
· **w file (*write*):** Esta instrución é parecida á instrución *print* a excepción de que envía a saída ao ficheiro especificado por *file*.
· **r file (*read*):** A instrución *read* lé os contidos do ficheiro especificado e os adxunta á liña seleccionada.
· **q (*quit*):** A instrución *quit* fai que *sed* remate inmediatamente.

Estruturas de control

- **! (NOT):** Fai que *sed* aplique a seguinte instrución, situada na mesma liña, a cada unha das liñas non seleccionadas pola sección da dirección da instrución.
- **{} (grupo de instrucións):** Cando encerramos un grupo de instrucións entre un par de chaves, unha soa dirección (ou unha parella de direccións) selecciona as liñas nas que opera o grupo de instrucións. Emprégase o punto e coma (;) para separar varios comandos que aparecen nunha soa liña.
- **: label:** Identifica unha localización dentro dun programa *sed*. A etiqueta é útil como obxectivo para as instrucións disxuntivas **b** e **t**.
- **b [label]:** Transfire o control incondicionalmente a *label*. Sen *label*, salta o resto de instrucións na liña actual da entrada e le a seguinte liña da entrada.
- **t [label]:** Transfire o control a *label* só se tivo éxito unha instrución *substitute* dende a liña mais recente de entrada que se leu. Sen *label*, salta o resto das instrucións da liña actual de entrada e le a seguinte liña de entrada.

O espazo modelo e o espazo contenedor

- A utilidade *sed* ten dous *buffer*. Os comandos revisados ata o de agora funcionan có espazo modelo, que inicialmente contén a liña de entrada que leu *sed*. O espazo contenedor pode conter datos mentres manipulamos datos no espazo modelo; é un *buffer* temporal. Vexamos os comandos que moven datos entre o espazo modelo e o espazo contenedor.

· **g :** Copia os contidos do espazo contenedor ao espacio modelo. Pérdense os contidos orixinais do espazo modelo.
· **G :** Adxunta unha nova liña e os contidos do espazo contenedor ao espazo modelo.
· **h :** Copia os contidos do espazo modelo ao espazo contenedor. Pérdense os contidos orixinais do espazo contenedor.
· **H :** Adxunta unha *NEWLINE* e os contidos do espazo modelo ao espazo contenedor.
· **x :** Intercambia os contidos do espazo modelo e o espazo contenedor.

Utilizando *Sed*

Modificar un texto por outro

Para modificar un texto por outro podemos empregar o parámetro "s":

```
$ cat texto.txt
ola e adeus, ola e adeus
ola e adeus, ola e adeus
#
# Cambia a primeira das coincidencias:
$ sed 's/ola/OLA/' texto.txt
OLA e adeus, ola e adeus
OLA e adeus, ola e adeus
#
# Cambiar a segunda das coincidencias:
$ sed 's/ola/OLA/2' texto.txt
ola e adeus, OLA e adeus
ola e adeus, OLA e adeus
#
# Cambiar todas as coincidencias existentes:
$ sed 's/ola/OLA/g' texto.txt
OLA e adeus, OLA e adeus
OLA e adeus, OLA e adeus
#
# Quitar todos os espazos:
$ sed 's/ //g' texto.txt
olaeadeus,olaeadeus
olaeadeus,olaeadeus
#
# Gardar a saída nun arquivo de texto:
$ sed 's/ola/OLA/w salida.txt' texto.txt
OLA e adeus, ola e adeus
OLA e adeus, ola e adeus
$ cat salida.txt
OLA e adeus, ola e adeus
OLA e adeus, ola e adeus
#
# Se NON queremos que o comando mostre a saída por pantalla, só a saída "-n":
$ sed -n 's/ola/OLA/gw salida.txt' texto.txt
$ cat salida.txt
OLA e adeus, OLA e adeus
OLA e adeus, OLA e adeus
```

Con **sed** é moi interesante empregar expresións regulares (precisamos empregar o parámetro **-E**). Para ver o seu funcionamento, podemos facer a seguinte práctica, nela empregamos o contido do arquivo /**etc**/**network**/**interfaces**, configurado cunha ip estática e, buscando as IPs existentes nel, crear un novo arquivo chamado **interfaces.mod** coas IPs cambiadas por **i.i.i.i**. Recorda que a RegExp dunha IP pode ser algo como: **([0-9]{1,3}\.){3}[0-9]{1,3}**

```
# Gardamos unha copia de /etc/network/interfaces
$ cp /etc/network/interfaces /home/usuario/interfaces
# Buscamos as liñas que teñen unha IP,
# empregamos 'p' para que mostre a saída e 'n' para que non repita a entrada.
$ sed -En '/([0-9]{1,3}\.){3}[0-9]{1,3}/p' interfaces
# Creamos o arquivo interfaces.mod
$ sed -E 's/([0-9]{1,3}\.){3}[0-9]{1,3}/i.i.i.i/g' interfaces > interfaces.mod
# Ver a diferencia do contido de interfaces.mod creado do seguinte xeito
$ sed -En 's/([0-9]{1,3}\.){3}[0-9]{1,3}/i.i.i.i/gw interfaces.mod' interfaces
# Neste último só aparecen as liñas modificadas...
```

- **Caracteres especiais**

  Hai caracteres que poden darnos problemas, vexamos un exemplo:

```
# Podemos empregar o carácter "\":
$ sed 's/\/bin\/sh/\/bin\/csh/' /etc/passwd
#
```

```
# O comando sed admite outro carácter distinto como delimitador:
$ sed 's!/bin/sh!/bin/csh!' /etc/passwd
```

Concatenar varias sentenzas

Para concatenar varias sentenzas en sed utilízase a opción "-e":

```
$ cat texto.txt
ola e adeus, ola e adeus
ola e adeus, ola e adeus
$ cat texto.txt | sed -e "s/ola/OLA/g" -e "s/ //g"
OLAeadeus,OLAeadeus
OLAeadeus,OLAeadeus
```

Filtrar liñas

Facer cambios nunha liña escollida pola súa posición no documento de texto:

```
$ cat texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
#
# Modificar a Liña 2:
$ sed '2s/ola/OLA/' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: OLA e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
#
# Modificar as Liñas 2 e 3:
sed '2,3s/ola/OLA/' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: OLA e adeus, ola e adeus
Liña 3: OLA e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
#
# Modificar todas as Liñas dende a 2 ata o final:
$ sed '2,$s/ola/OLA/' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: OLA e adeus, ola e adeus
Liña 3: OLA e adeus, ola e adeus
Liña 4: OLA e adeus, ola e adeus
```

- **Facer cambios nunha liña escollida por unha coincidencia no seu contido:**

```
$ cat texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
#
# Modificamos a "Liña 3":
$ sed '/Liña 3/s/ola/OLA/' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: OLA e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
```

-> Recordade que se poden empregar as expresións regulares có parámetro **-E**.

- **Só visualizar liñas, sen facer cambios:**

-Visualizar todas as liñas do comando **ip a show enp0s3** que conteñen a palabra *inet*.

```
$ ip a show enp0s3 | sed '/inet/p'
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:45:33:71 brd ff:ff:ff:ff:ff:ff
```

```
    inet 192.168.1.110/24 brd 192.168.1.255 scope global enp0s3
    inet 192.168.1.110/24 brd 192.168.1.255 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe45:3371/64 scope link
    inet6 fe80::a00:27ff:fe45:3371/64 scope link
       valid_lft forever preferred_lft forever
```

-Se engadimos a opción **-n** *sed* só visualiza as liñas seleccionadas:

```
$ ip a show enp0s3 | sed -n '/inet/p'
    inet 192.168.1.110/24 brd 192.168.1.255 scope global enp0s3
    inet6 fe80::a00:27ff:fe45:3371/64 scope link
```

-Tamén podemos empregar Expresións Regulares engadindo o parámetro **-E**:

```
$ ip a show enp0s3 | sed -En '/\binet\b/p'
    inet 192.168.1.110/24 brd 192.168.1.255 scope global enp0s3
```

Visualizar só as primeiras liñas

Visualizar as 3 primeiras liñas

```
$ ip a show enp0s3 | sed '3 q'
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:45:33:71 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.110/24 brd 192.168.1.255 scope global enp0s3
```

Visualizar as últimas liñas

```
# Ver só a última liña
$ ip a show enp0s3 | sed -n '$p'
       valid_lft forever preferred_lft forever
#
# Mostrar dende a liña 3 ata o final:
$ ip a show enp0s3 | sed -n '3,$p'
    inet 192.168.1.110/24 brd 192.168.1.255 scope global enp0s3
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe45:3371/64 scope link
       valid_lft forever preferred_lft forever
```

Máis omplicado de facer é mostrar un número determinado de liñas do final (emulación do comando tail), vexamos como:

- **Visualizar só as 3 últimas liñas dun documento:**

```
$ ip a show enp0s3 | sed '$q; :loop; N; 4,$D; b loop'
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe45:3371/64 scope link
       valid_lft forever preferred_lft forever
```

Explicación:

; - Para combinar varios comandos sed nunha liña.
:loop - Creamos a etiqueta "loop" para poder saltar a ela en calquera momento con "b loop". Créase así unha iteración (Explicación da creación de etiquetas en sed).
$q - Na última liña deixa de executar o comando (**q** é o *quit command*) e imprime o *pattern space*.
N - Salto á seguinte liña.
4,$D - Selecciónanse sempre 4 liñas do documento borrando a primeira delas e deixando así as 3 últimas das seleccionadas.
Como esta iteración faise dende o principio do documento ata o final, conséguese seleccionar só as 3 últimas liñas...

◊ Visualizar só as 2 últimas liñas dun documento:

```
$ ip a show enp0s3 | sed '$!N;$!D'
    inet6 fe80::a00:27ff:fe45:3371/64 scope link
       valid_lft forever preferred_lft forever
```

Executar máis dun comando nunha liña individual

Para executar máis dun comando nunha liña individual podemos empregar "chaves". Vexamos un exemplo:

```
$ ip a show enp0s3 | sed -E '{
s/([0-9]{1,3}\.){3}[0-9]{1,3}/iii.iii.iii.iii/g
s/\/[0-9]{1,2}/\/mm/g
}'
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:45:33:71 brd ff:ff:ff:ff:ff:ff
    inet iii.iii.iii.iii/mm brd iii.iii.iii.iii scope global enp0s3
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe45:3371/mm scope link
       valid_lft forever preferred_lft forever
```

Borrar liñas

O comando *delete*, **d**, permítenos borrar as liñas seleccionadas. Vexamos uns exemplos que ilustran o seu funcionamento:

```
# Arquivo de traballo:
$ cat texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
#
# Borrar todo o arquivo:
$ sed 'd' texto.txt
#
# Borrar a terceira liña:
$ sed '3 d' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
#
# Borrar as liñas 2 e 3:
$ sed '2,3 d' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
#
# Borrar as liñas dende a 3 ata o final:
$ sed '3,$ d' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
#
# Borrar a liña que ten o texto "Liña 2":
$ sed '/Liña 2/d' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
#
# Si es una expresión regular recuerda añadir el parámetro '-E'
```

Tamén se pode borrar un rango de liñas empregando dous texto de busca, pero hai que ter coidado con isto. O primeiro texto de busca especifica o "inicio" do borrado das liñas, e o segundo texto de busca especifica o "fin" do borrado de liñas. O editor **sed** borra as liñas existentes entre as dúas seleccionadas:

```
$ cat texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
$ sed '/2/,/3/d' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
#
# O problema está se volve a existir unha coincidencia do "inicio do borrado das liñas":
$ cat texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
```

```
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
...
Liña 12: ola e adeus, ola e adeus
...
$ sed '/2/,/3/d' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
...
```

Insertar e engadir texto

O editor *sed* permite insertar e engadir liñas de texto. Vexamos exemplos para ver as diferencias entre eles:

```
$ cat texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
...
Liña 12 outra vez: ola e adeus, ola e adeus
...
# Nova liña antes dunha seleccionada:
$ sed '7i\Liña 11: Nova liña\' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
...
Liña 11: Nova liña
Liña 12 outra vez: ola e adeus, ola e adeus
...
#
# Nova liña despois dunha seleccionada:
$ sed '5a\Liña 6: Nova liña\' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
Liña 6: Nova liña
...
Liña 12 outra vez: ola e adeus, ola e adeus
...
#
# Nova liña ao final:
$ sed '$a\Última liña\' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
...
Liña 12 outra vez: ola e adeus, ola e adeus
...
Última liña
#
# Nova liña ao principio do documento:
$ sed '1i\Primeira liña\' texto.txt
Primeira liña
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
...
Liña 12 outra vez: ola e adeus, ola e adeus
...
```

Cambiar o contido dunha liña

Para cambiar o contido dunha liña hai que especificar separadamente a liña na que hai que traballar.

```
$ cat texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
...
Liña 12 outra vez: ola e adeus, ola e adeus
...
#
# Pódese especificar unha liña có número que ocupa:
$ sed '3c\Esta é a liña cambiada...' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Esta é a liña cambiada...
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
...
Liña 12 outra vez: ola e adeus, ola e adeus
...
#
# Pódese tamén especificar unha liña por coincidencia dun texto:
$ sed '/Liña 3/c\Esta é a liña cambiada...' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Esta é a liña cambiada...
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
...
Liña 12 outra vez: ola e adeus, ola e adeus
...
#
# Pódese intentar facer o cambio de varias liñas, pero o resultado non será o esperado:
$ sed '2,4c\Esta é a liña cambiada...' texto.txt
Liña 1: ola e adeus, ola e adeus
Esta é a liña cambiada...
Liña 5: ola e adeus, ola e adeus
...
Liña 12 outra vez: ola e adeus, ola e adeus
...
```

O comando "transformar"

O comando "transformar" (**y**) é o único comando *sed* que opera con caracteres simples. Vexamos un exemplo:

```
$ cat texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
...
Liña 12 outra vez: ola e adeus, ola e adeus
...
$ sed 'y/123/789/' texto.txt
Liña 7: ola e adeus, ola e adeus
Liña 8: ola e adeus, ola e adeus
Liña 9: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
...
Liña 78 outra vez: ola e adeus, ola e adeus
...
```

Comandos para imprimir

Existen tres comandos: **p**, **=** e **l**.

```
# Comando p:
$ sed 'p' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
...
...
Liña 12 outra vez: ola e adeus, ola e adeus
Liña 12 outra vez: ola e adeus, ola e adeus
...
...
#
# Comando = :
$ sed '=' texto.txt
1
Liña 1: ola e adeus, ola e adeus
2
Liña 2: ola e adeus, ola e adeus
3
Liña 3: ola e adeus, ola e adeus
4
Liña 4: ola e adeus, ola e adeus
5
Liña 5: ola e adeus, ola e adeus
6
...
7
Liña 12 outra vez: ola e adeus, ola e adeus
8
...
#
# Comando l :
$ sed 'l' texto.txt
Li\303\261a 1: ola e adeus, ola e adeus$
Liña 1: ola e adeus, ola e adeus
Li\303\261a 2: ola e adeus, ola e adeus$
Liña 2: ola e adeus, ola e adeus
Li\303\261a 3: ola e adeus, ola e adeus$
Liña 3: ola e adeus, ola e adeus
Li\303\261a 4: ola e adeus, ola e adeus$
Liña 4: ola e adeus, ola e adeus
Li\303\261a 5: ola e adeus, ola e adeus$
Liña 5: ola e adeus, ola e adeus
...$
...
Li\303\261a 12 outra vez: ola e adeus, ola e adeus$
Liña 12 outra vez: ola e adeus, ola e adeus
...$
...
```

Usando arquivos

- **Escribindo nun arquivo có comando *w* :**

Vexamos uns exemplos:

```
# Texto có que imos traballar:
$ cat texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
```

```
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
#
# Seleccionamos só as dúas primeiras liñas do arquivo anterior
# a saída a gardamos no arquivo "test"
# o parámetro "-n" nos garantiza que o comando non devolve echo:
$ sed -n '1,2 w test' texto.txt
$ cat test
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
#
# Tamén podemos seleccionar filas empregando unha coincidencia con ou sen RegExp:
$ sed -En '/^Liña 3/w test' texto.txt
$ cat test
Liña 3: ola e adeus, ola e adeus
```

- **Lendo dun arquivo có comando *r* :**

A idea deste comando é engadir datos dun arquivo á saída dunha sentenza *sed*:

```
# Arquivo de traballo "texto.txt":
$ cat texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
#
# Arquivo "novo.txt" que imos engadir ao outro:
$ cat novo.txt
Esta é unha liña nova
Esta é unha segunda liña nova
#
# Engadiremos o contido de "novo.txt" despois da Terceira Liña do arquivo "texto.txt":
$ sed '3r novo.txt' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Esta é unha liña nova
Esta é unha segunda liña nova
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
#
# Insertamos agora o contido de "novo.txt" despois da coincidencia "Liña 2":
$ sed '/Liña 2/r novo.txt' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Esta é unha liña nova
Esta é unha segunda liña nova
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
#
# Engadir texto ao final dun arquivo faise có símbolo "$":
$ sed '$r novo.txt' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
Esta é unha liña nova
Esta é unha segunda liña nova
```

O comando de lectura "r" é realmente interesante cando se combina có comando de borrado "d":

```
$ sed '/Liña 2/{
r novo.txt
d
}' texto.txt
Liña 1: ola e adeus, ola e adeus
Esta é unha liña nova
```

```
Esta é unha segunda liña nova
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
```

Emprego de *program-file*

Cando precisamos dar a *sed* instrucións mais complexas ou mais longas, podemos empregar un *program-file*. A opción *-f* dille a *sed* que debería ler o seu programa dende o ficheiro chamado na liña de comando.

```
$ cat comandos.sed
s/([0-9]{1,3}\.){3}[0-9]{1,3}/i.i.i.i/g
s/\/[0-9]{1,2}/\/mm/g
####
####
$ ip a show enp0s3 | sed -E -f comandos.sed
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 08:00:27:45:33:71 brd ff:ff:ff:ff:ff:ff
    inet i.i.i.i/mm brd i.i.i.i scope global enp0s3
       valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe45:3371/mm scope link
       valid_lft forever preferred_lft forever
```

Comandos multiliña

Cós comandos vistos ata o de agora con **sed** vemos este traballa separando o arquivo "por liñas".
Hai situacións que requiren traballar en máis dunha liña (por exemplo se queremos substituir unha frase completa nun arquivo).
O editor **sed** posúe tres comandos especiais que permiten procesar textos multiliña:
- **N**: Engade a seguinte liña da escollida como parte da liña na que estamos a traballar.
- **D**: Borra unha liña simple nun grupo de liñas.
- **P**: Imprime unha liña simple nun grupo de liñas.

Vexamos o funcionamento destes comandos.

Os comandos *next*

Temos dous comandos *next* distintos:

◊ **O comando *next* dunha liña.**
O comando **n** dille a *sed* que pase á seguinte liña sen volver ao principio dos comandos.
Vexamos un exemplo onde "se elimina a liña seguinte da Liña 2":

```
$ cat texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
$ sed '/Liña 2/{
n
d
}' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
```

◊ **O comando *next* multiliña.**
O comando **N** dille a *sed* que agregue a seguinte liña ao texto seleccionado có que se vai traballar.
Vexamos un exemplo onde "se eliminan dúas liñas, a **Liña 2** seleccionada e a seguinte a ésta'":

```
$ cat texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
$ sed '/Liña 2/ {
```

```
> N
> d
> }' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
```

Outro exemplo de utilización do comando *next* multiliña:

```
$ cat texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
$ sed '=' texto.txt
1
Liña 1: ola e adeus, ola e adeus
2
Liña 2: ola e adeus, ola e adeus
3
Liña 3: ola e adeus, ola e adeus
4
Liña 4: ola e adeus, ola e adeus
5
Liña 5: ola e adeus, ola e adeus
$ sed '=' texto.txt | sed 'N; s/\n/ /'
1 Liña 1: ola e adeus, ola e adeus
2 Liña 2: ola e adeus, ola e adeus
3 Liña 3: ola e adeus, ola e adeus
4 Liña 4: ola e adeus, ola e adeus
5 Liña 5: ola e adeus, ola e adeus
```

Comando de borrado multiliña

Como pudimos comprobar no apartado anterior, o comando **d** nos permite borrar unha liña do arquivo có que estamos a traballar.
O editor **sed** tamén nos ofrece o comando de borrado multiliña **D** que só borrará a primeira liña dun conxunto de liñas seleccionadas.
Vexamos un exemplo onde se diferencia como traballan os comandos **d** e **D**:

```
$ cat texto.txt
Liña 1: ola e adeus, ola e adeus

Liña 2: ola e adeus, ola e adeus

Liña 3: ola e adeus, ola e adeus

Liña 4: ola e adeus, ola e adeus

Liña 5: ola e adeus, ola e adeus
...
#
# Borramos unha liña en branco e a seguinte liña se despois ten o texto "Liña 2":
$ sed '/^$/ {
N
/Liña 2/d
}' texto.txt
Liña 1: ola e adeus, ola e adeus

Liña 3: ola e adeus, ola e adeus

Liña 4: ola e adeus, ola e adeus

Liña 5: ola e adeus, ola e adeus
...
#
# Borramos a liña que ten o texto "Liña 2" se a anterior está en branco:
$ sed '/^$/ {
N
/Liña 2/D
}' texto.txt
Liña 1: ola e adeus, ola e adeus
```

```
Liña 2: ola e adeus, ola e adeus

Liña 3: ola e adeus, ola e adeus

Liña 4: ola e adeus, ola e adeus

Liña 5: ola e adeus, ola e adeus
...
```

Comando de impresión multiliña

Seguindo a mesma filosofía dos dous comandos "multiliña" vistos, o comando **P** imprimirá só a primeira liña dun conxunto de liñas seleccionadas.
Como ocorría có comando **p**, irá acompañado da opción **-n** para que *sed* non devolva a saída normal do comando.
Vexamos uns exemplos:

```
$ cat texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
...
#
# Comando que imprime todas as liñas do texto que ten no seu interior a palabra "Liña" e a seguinte a esta liña
# por iso sae tamén a última liña:
$ sed -n '/Liña/ {
N
p
}' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
...
#
# Comando que se queda só coa primeira das liñas seleccionadas de cada unha...
# Así conseguimos seleccionar "unha liña si e outra non":
$ sed -n '/Liña/ {
N
P
}' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
```

O "*Hold space*"

Ata o de agora coñecíamos o *pattern space* que é un *buffer* empregado por *sed* para gardar o texto seleccionado mentres procesa sobre el os comandos que nos interesan.
Tamén existe en *sed* outro *buffer* chamado *hold space*, que podemos empregar para gardar liñas de texto mentres traballamos con outras liñas.
Temos cinco comandos asociados ao *hold space*:

Comando	Descrición
h	Copia *pattern space* a *hold space*
H	Engade *pattern space* a *hold space*
g	Copia *hold space* a *pattern space*
G	Engade *hold space* a *pattern space*
x	Intercambia os contidos de *hold space* e *pattern space*

Vexamos un exemplo:

```
$ cat texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
...
$ sed -n '/Liña 2/{
h
p
n
p
g
p
}' texto.txt
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
```

1. Primeiro seleccionamos a liña que contén o texto "Liña 2".
2. Cando unha liña contén "Liña 2", o comando **h** coloca esa liña no *hold space*.
3. O comando **p** imprime o contido do *pattern space*.
4. O comando **n** recupera a seguinte liña no *stream* de datos e coloca esta liña no *pattern space*.
5. O comando **p** imprime o contido do *pattern space*, que é a liña que se atopa despois da que ten o texto "Liña 2".
6. O comando **g** coloca o contido do *hold space* (a liña que contén o texto "Liña 2") no *pattern space*, remprazando o texto existente nel.
7. O comando **p** imprime o contido do *pattern space*, que volve a ser a liña que contén o texto "Liña 2".

Tendo esta explicación en conta, podemos cambiar a sentenza anterior para que poña as liñas 2 e 3 en orde inverso:

```
$ sed -n '/Liña 2/{
h
n
p
g
p
}' texto.txt
Liña 3: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
```

Negando un comando

O comando **!** emprégase para negar un comando. Isto significa que en situacións onde o comando estaría normalmente activado, pois agora xa non.
O mellor é ver un exemplo:

```
$ cat texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
...
#
# Faremos que se devolvan todas as liñas menos a que ten o texto "Liña 2":
$ sed -n '/Liña 2/!p' texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
...
```

Así podemos facer unha sentenza para darlle a volta a un texto, que tería que facer o seguinte:

1. Colocar a primeira liña no *hold space*.
2. Poñer a seguinte liña de texto no *pattern space*.

3. Agregar o *hold space* ao *pattern space*.
4. Colocar todo o *pattern space* no *hold space*.
5. Repetir os pasos do 2 ao 4 ata que xa estén todas as liñas no *hold space* pero ao revés que no texto base.
6. Recuperar as liñas e imprimilas.

A sentenza que consegue todo isto será a seguinte:

```
$ cat texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
#
$ sed -n '{
1!G
h
$p
}' texto.txt
Liña 5: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 1: ola e adeus, ola e adeus
```

1. **n -** : Non imprimir a saída.
2. **1!** : Todas as liñas menos a primeira.
3. **G** : Engadir o *hold space* ao *pattern space*.
4. **h** : Remprazar o contido do *hold space* có contido do *pattern space*.
5. **$p** : Cando se chega a última liña imprímese o contido do *pattern space*.

Cambiando o fluxo

Normalmente, o editor *sed* procesa os comandos de arriva a abaixo (con excepción ao comando **D**, que forza ao editor *sed* a retornar á parte de arriva do script sen ler a seguinte liña do texto).
O editor *sed* nos ofrece varios métodos para alterar o fluxo do *script*, producindo un resultado similar ao da programación estruturada.

Derivación

O formato do comando "derivación" é o seguinte:

```
[address]b [label]
```

O parámetro *address* determina que liña ou liñas de datos activan este comando.
O parámetro *label* define a "derivación".
Se o parámetro *label* non se introduce, a "derivación" estará no final do *script*.
Vexamos un exemplo:

```
$ cat texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
$ sed '{
2,3b
s/ola/OLA/
}' texto.txt
Liña 1: OLA e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: OLA e adeus, ola e adeus
Liña 5: OLA e adeus, ola e adeus
```

Nas Liñas 2 e 3 facemos unha substitución.

Vexamos outro exemplo onde empregamos *label* e así facemos unhas substitucións só nas liñas 2 e 3, e outra substitución distinta en todas as liñas do texto:

```
$ sed '{
2,3b salto
s/ola/OLA/
s/Liña/LIÑA/
:salto
s/adeus/ADEUS/
}' texto.txt
LIÑA 1: OLA e ADEUS, ola e adeus
Liña 2: ola e ADEUS, ola e adeus
Liña 3: ola e ADEUS, ola e adeus
LIÑA 4: OLA e ADEUS, ola e adeus
LIÑA 5: OLA e ADEUS, ola e adeus
```

Testing

O comando *test* (**t**) é tamén empregado para modificar o fluxo do *script* de *sed*.
O formato do comando *test* é o seguinte:

```
[address]t [label]
```

O comando *t* ofrécenos o xeito de implementar a declaración *if-then* nos *scripts sed*.
Vexamos un exemplo:

```
$ cat texto.txt
Liña 1: ola e adeus, ola e adeus
Liña 2: ola e adeus, ola e adeus
Liña 3: ola e adeus, ola e adeus
Liña 4: ola e adeus, ola e adeus
Liña 5: ola e adeus, ola e adeus
$ sed '{
s/Liña 3/Liña Nova/
t
s/ola/OLA/
}' texto.txt
Liña 1: OLA e adeus, ola e adeus
Liña 2: OLA e adeus, ola e adeus
Liña Nova: ola e adeus, ola e adeus
Liña 4: OLA e adeus, ola e adeus
Liña 5: OLA e adeus, ola e adeus
```

Se a primeira substitución non se da, executarase a segunda. Pero se a primeira substitución se executa, xa non se fará a segunda.

Exemplos

```
# A primeira coincidencia (palabra neste caso) dunha liña:
$ sed 's/\([A-Za-z]*\).*/\1/' texto.txt
#
# A segunda coincidencia dunha liña (neste caso, o resto logo da primeira palabra):
$ sed 's/\([A-Za-z]*\)\(.*\)/\2/' texto.txt
#
# Cambiar de orde dúas palabras:
$ echo "ola adeus" | sed 's/\([a-z]*\) \([a-z]*\)/\2 \1/'
adeus ola
#
# Eliminar unha palabra repetida:
$ echo "ola ola" | sed 's/\([a-z]*\) \1/\1/'
ola
#
# Poñer parénteses ao redor da primeira palabra
#### A expresión regular "[^ ]" significa calquera carácter menos o espazo:
$ sed 's/[^ ]*/(&)/' texto.txt
(Liña) 1: ola e adeus, ola e adeus
#
# Poñer parénteses ao redor de "todas" as palabras:
$ sed 's/[^. ]*/(&)/g' texto.txt
(Liña) (1:) (ola) (e) (adeus,) (ola) (e) (adeus)
```

```
#
# Cambiar a segunda letra "a" de cada liña por "A" e gardar a saída no arquivo "file":
$ sed -n 's/a/A/2pw file' texto.txt
Liña 1: olA e adeus, ola e adeus
#
# Se queremos traballar con sed nun script shell e empregar argumentos...
# O seguinte script emula o funcionamento de grep:
#!/bin/bash
sed -n 's/'"$1"'/&/p'
#### As dobres comiñas empréganse por se o argumento ten espazos...
#
#...
```

Exemplo de cambio da configuración IP

---- Este exemplo NON está totalmente rematado ----

- Facemos un arquivo chamado **interfaces.base** có seguinte contido:

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/*

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
#auto enp0s3
#iface enp0s3 inet dhcp
#iface enp0s3 inet static
#address ip
#netmask ms
#gateway pe
```

- E tamén creamos o arquivo **cambiaIP.sh** que será o script bash que modifique a configuración de rede:

```
#!/bin/bash

archivobase="./interfaces.base"
archivosalida="./interfaces"
ip="192.168.0.200"
ms="255.255.255.0"
pe="192.168.0.1"

#Facemos un arquivo de comandos sed
echo " /auto enp0s3/s/#/ / " > sedfile
echo " /static/s/#/ / " >> sedfile

echo " /address/s/#/ /" >> sedfile
tmp=" s/ip/$ip/ "
echo "$tmp" >> sedfile

echo " /netmask/s/#/ /" >> sedfile
tmp=" s/ms/$ms/ "
echo "$tmp" >> sedfile

echo " /gateway/s/#/ /" >> sedfile
tmp=" s/pe/$pe/ "
echo "$tmp"  >> sedfile

#Executamos o comando sed lendo o arquivo antes creado
##Vai sobre o arquivo base e a saída crea o interfaces
### que logo copiaremos no directorio correspondente /etc/network/
sed -f sedfile $archivobase > $archivosalida
```

Enlaces interesantes

- Sed con exemplos en gentoo.org
- FAQ de Eriq Pement
- Tutorial de Sed por Bruce Barnett