Файлове, директории и права с цел конрол на достъпа до системите

- Блокове
- Inodes
- Връзки – твърди и символни

Блокове:

При изчисляването блокът, понякога наричан физически запис, е последователност от байтове или битове, обикновено съдържаща цял брой записи, имащи максимална дължина; размер на блок.
Така структурирани данни се казва, че са блокирани (Blocked) .

Inodes:

Inode е структура на данни във файлова система в стил Unix, която описва обект на файлова система като файл или директория. Всеки inode съхранява атрибутите и дисковите блокови местоположения на данните на обекта.

Links (линкове):
Ефективно можем да ги наречем shortcuts или път към оригиналния файл.

Разделят се на твърди и символични: Hard links and Symbolic links.

Symoblic links или още така наречените soft линкове правят връзка между два файла като пишейки в създадения със командата ln ние можем да препращаме информацията в оригиналния.

Създаване на символична връзка и преглед:

```
nick@xmachine:~/Test/ln/etc$ ls -lah
total 8,0K
drwxrwxr-x 2 nick nick 4,0K окт 20 14:07 .
drwxrwxr-x 3 nick nick 4,0K окт 20 14:07 ..
nick@xmachine:~/Test/ln/etc$ touch config{1..2}.conf
nick@xmachine:~/Test/ln/etc$ ls -lah
total 8,0K
drwxrwxr-x 2 nick nick 4,0K окт 20 14:07 .
drwxrwxr-x 3 nick nick 4,0K окт 20 14:07 ..
-rw-rw-r-- 1 nick nick    0 окт 20 14:07 config1.conf
-rw-rw-r-- 1 nick nick    0 окт 20 14:07 config2.conf
nick@xmachine:~/Test/ln/etc$ cd ..
nick@xmachine:~/Test/ln$ ln -s etc/config
config1.conf  config2.conf
nick@xmachine:~/Test/ln$ ln -s etc/config1.conf conf1
nick@xmachine:~/Test/ln$ ls -lah
total 12K
drwxrwxr-x 3 nick nick 4,0K окт 20 14:08 .
drwxrwxr-x 8 nick nick 4,0K окт 19 19:53 ..
lrwxrwxrwx 1 nick nick   16 окт 20 14:08 conf1 -> etc/config1.conf
drwxrwxr-x 2 nick nick 4,0K окт 20 14:07 etc
```

Hard links или твърди линкове е файл който споделя същата inode структура. Тоест за да проверите да ли имате успешно направен hard link ще трябва да проверите файловият номер с ln -i . Сега идва въпроса какви са тези номера. Този номер който ще видите използвайки командата е следствие на факта, че цялата линукс система е изградена на идеята да споделя правилото “ВСИЧКО Е ФАЙЛ” което свежда, че за да се различават тези файлове те трябва да са маркирани от системата с цел да бъдат специфични за дадената среда.

Създаване на твърда връзка и преглед:

```
nick@xmachine:~/Test/ln$ ls -lah
total 12K
drwxrwxr-x 3 nick nick 4,0K окт 20 14:08 .
drwxrwxr-x 8 nick nick 4,0K окт 19 19:53 ..
lrwxrwxrwx 1 nick nick   16 окт 20 14:08 conf1 -> etc/config1.conf
drwxrwxr-x 2 nick nick 4,0K окт 20 14:07 etc
nick@xmachine:~/Test/ln$ ln -P etc/config2.conf conf2
nick@xmachine:~/Test/ln$ ls -lah
total 12K
drwxrwxr-x 3 nick nick 4,0K окт 20 14:10 .
drwxrwxr-x 8 nick nick 4,0K окт 19 19:53 ..
lrwxrwxrwx 1 nick nick   16 окт 20 14:08 conf1 -> etc/config1.conf
-rw-rw-r-- 2 nick nick    0 окт 20 14:07 conf2
drwxrwxr-x 2 nick nick 4,0K окт 20 14:07 etc
```

В примера горе виждате, че символичната връзка си личи много по ясно от твърдата.
За да проверите дали inode е споделен ще трябва да проверим дали conf файла има същият inode номер който има и файла в etc с име config2.conf.

Как изглежда:

```
nick@xmachine:~/Test/ln$ cd etc/
nick@xmachine:~/Test/ln/etc$ ls -li
total 0
19271312 -rw-rw-r-- 1 nick nick 0 окт 20 14:07 config1.conf
19271317 -rw-rw-r-- 2 nick nick 0 окт 20 14:07 config2.conf
nick@xmachine:~/Test/ln/etc$ cd ..
nick@xmachine:~/Test/ln$ ls -li
total 4
19271318 lrwxrwxrwx 1 nick nick   16 окт 20 14:08 conf1 -> etc/config1.conf
19271317 -rw-rw-r-- 2 nick nick    0 окт 20 14:07 conf2
19271609 drwxrwxr-x 2 nick nick 4096 окт 20 14:07 etc
```

Както виждате с командата ls -li аз правя лист с inode номера и файловете и в случая виждаме ясно, че inode номерата на etc/config2 и на conf2 си съвпадат. Тоест това което посочва това като факт за нас е, че данните между тези два файла се споделят и, че имат напълно една и съща структура данни.

Права и контрол на достъпа в Линукс файловата система.

Нещата които трябва да знаете първо са следните два факта. Привилегийте на всички файлове с системата на линукс се изписват буквено и цифрено.

Буквено изглеждат така:

r	Права за четене идва от read.
w	Права за писане идва от write.
x	Права за изпълняване идва от execution.
-	Посочва, че няма никакви права дадения файл или папка (директория).

Както е казано закона под който линукс работи е, че всичко е файл тоест и директорийте. Което посочва, че имаме и допълнителна информация посочваща това в началото на файловете и директорийте намира се най отпред на чело на правата:

-	Посочва, че е файл.
d	Посочва, че е директория (папка).
l	Посочва, че файла е линкнат.

```
nick@xmachine:~/Test/perm$ ls -lih
total 4,0K
19271321 drwxrwxr-x 2 nick nick 4,0K окт 20 14:22 directory_example
19271319 -rw-rw-r-- 1 nick nick    0 окт 20 14:21 file.example
19271323 lrwxrwxrwx 1 nick nick   30 окт 20 14:22 link-exampl -> directory_example/link.example
```

Цифрен формат за записване на правата (октален):

4	Четене
2	Писане
1	Изпълнение
0	Без правомощия

Файловете имат три типа вид на собственост – потребител, група и всички останали, за това те имат следния формат след преглед със ls -l в всяка ситуация. За да знаете кои права за кого са, ги разделяте по 3. Първата тройка е за потребителя, втората е групата а третата е всички останали.

Команди за промяна на правата:

chmod u+r {filename}	Дава правото на потребителя да чете даденият файл или директория.
chmod u+w {filename}	Дава правото на потребителя да може да пише по дадения файл или директория.
chmod u+x {filename}	Дава правото на потребителя да може да изпълнява даденият файл или директория.

chmod u-r {filename}	Премахва правото на потребителя да чете даденият файл или директория.
chmod u-w {filename}	Премахва правото на потребителя да пише даденият файл или директория.
chmod u-x {filename}	Премахва правото на потребителя да изпълнява даденият файл или директория.

chmod g+r {filename}	Дава правото на групата да може да чете даденият файл или директория.
chmod g+w {filename}	Дава правото на групата да може да пише даденият файл или директория.
chmod g+x {filename}	Дава правото на групата да може да изпълнява даденият файл или директория.

chmod g-r {filename}	Премахва правото на групата да може да чете даденият файл или директория.
chmod g-w {filename}	Премахва правото на групата да може да пише даденият файл или директория.
chmod g-x {filename}	Премахва правото на групата да може да изпълнява даденият файл или директория.

chmod o+r {filename}	Дава правото на всички останали да четат даденият файл или директория.
chmod o+w {filename}	Дава правото на всички останали да пишат даденият файл или директория.
chmod o+x {filename}	Дава правото на всички останали да изпълняват даденият файл или директория.

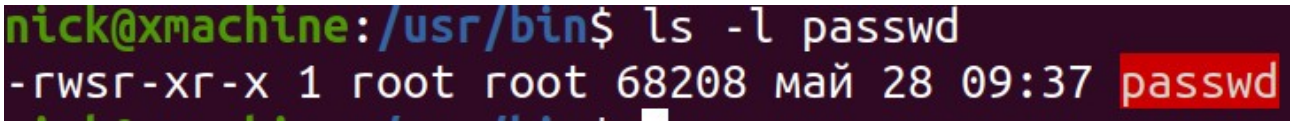
chmod o-r {filename}	Дава правото на всички останали да четат даденият файл или директория.

chmod o-w {filename}	Дава правото на всички останали да пишат даденият файл или директория.
chmod o-x {filename}	Дава правото на всички останали да изпълняват даденият файл или директория.

chown {username} file.extension	Променя потребителя собственик на файл.
chgrp {groupname} file.extension	Променя групата собственик на файл.
addgroup {groupname}	Добавя нова група.
chown :{groupname} file.extension	Променя основната група собственик на файла.
chown {username}:{groupname} file.extension	Променя потребителя и групата собственици на файла

SUID, SGUID и Sticky bit.

SUID:
Прилагане на право на собственика което се бележи със “s” вместо знака за изпълнение “x”. Идеята е ако имате един скрипт който трябва да се ползва от всички потребители да се добави тази привилегия. Веднъж добавена това дава шанса на потребителите да ползват скрипта без да са го направили. Пример за такава програмка е “passwd”.



```
nick@xmachine:~/Test/perm/SUID$ touch file.extension
nick@xmachine:~/Test/perm/SUID$ chmod 760 file.extension
nick@xmachine:~/Test/perm/SUID$ ls -l
total 0
-rwxrw---- 1 nick nick 0 окт 20 17:30 file.extension
nick@xmachine:~/Test/perm/SUID$ chmod 4764 file.extension
nick@xmachine:~/Test/perm/SUID$ ls -l
total 0
-rwsrw-r-- 1 nick nick 0 окт 20 17:30 file.extension
nick@xmachine:~/Test/perm/SUID$ chmod 0764
chmod: missing operand after ‘0764’
Try 'chmod --help' for more information.
nick@xmachine:~/Test/perm/SUID$ chmod 0764 file.extension
nick@xmachine:~/Test/perm/SUID$ ls -l
total 0
-rwxrw-r-- 1 nick nick 0 окт 20 17:30 file.extension
```

SGUID:
Прилагане на право на групата което се бележи със “s” вместо знака за изпълнение “x”. Идеята е ако имате един скрипт или програма, чиято група се съдържа много потребители, те да могат да я изполват. Пример за такава програма е “ssh-agent”.

```
nick@xmachine:/usr/bin$ ls -l ssh-agent
-rwxr-sr-x 1 root ssh 350504 май 29 10:37 ssh-agent
```

```
nick@xmachine:~/Test/perm/SGUID$ ls -l
total 0
-rwxrw-r-- 1 nick nick 0 окт 20 17:32 file.extension
nick@xmachine:~/Test/perm/SGUID$ chmod 0764 file.extension
nick@xmachine:~/Test/perm/SGUID$ chmod 2764 file.extension
nick@xmachine:~/Test/perm/SGUID$ ls -l
total 0
-rwxrwSr-- 1 nick nick 0 окт 20 17:32 file.extension
nick@xmachine:~/Test/perm/SGUID$ chmod 0764 file.extension
nick@xmachine:~/Test/perm/SGUID$ ls -l
total 0
-rwxrw-r-- 1 nick nick 0 окт 20 17:32 file.extension
nick@xmachine:~/Test/perm/SGUID$
```

Sticky bit:
Прилагане на право на което позволява да може само потребителя създател да изтрие даденият файл и никой друг.

```
nick@xmachine:~/Test/perm/Sticky$ ls -l
total 0
-rw-rw-r-- 1 nick nick 0 окт 20 17:36 file.extension
nick@xmachine:~/Test/perm/Sticky$ chmod 1764 file.extension
nick@xmachine:~/Test/perm/Sticky$ ls -l
total 0
-rwxrw-r-T 1 nick nick 0 окт 20 17:36 file.extension
nick@xmachine:~/Test/perm/Sticky$ chmod 0764 file.extension
nick@xmachine:~/Test/perm/Sticky$ ls -l
total 0
-rwxrw-r-- 1 nick nick 0 окт 20 17:36 file.extension
```

su - {username}	Смяна на потребителя.
mkdir {directory name}	Създава папка.
groupadd {groupname}	Добавяне на група.
usermod -aG {group} {username}	Добавете потребителя към допълнителните групи. Винаги с -aG ако добавяте група към

	потребител.
touch {filename or filename{1..-} }	Създава един файл по фаше наименование или може да направи много файлове ако добавите преди extension на файла {1..което и да е чило = ще отговаря на максималния брой файлове}
useradd {username }	Добавяте потребител в някой системи интерактивно в някой не, използвайте внимателно.
passwd {username}	Променя паролата на потребител.
adduser {username}	Добавя нов потребител интерактивно.
chmod -R {permissions} {folder}/	Променяте правата на папка и файловото съдържание.
usermod -G {group} {username}	Сменяте основната група на потребителя.

umask
Показва текущите настройки с които потребителя в действие ще създава папки и файлове.

Обикновенни стойности:

- 0002 за нормални потребители.
- 0022 за root потребителя.

Директорна структура:

• Обърната дървовидна структура с един корен (боаб).

• Разграничението на малки от големи букви (case sensitive).

• Всяка папка или файл, чието име започва с . (точка) е скрита от нормалният „поглед“.

• Просто . В папката (директорията) обозначава текущата директория.

• Две .. Са предходната (една папка назад или нагоре към първият клон).

Команди с цел откриване и локализиране:

• locate : комадна която търси в локаната база данни за папки и файлове които съвпадат определен критерии.
• updatedb : опреснява локаната база с данни която locate ползва.

• whereis : това е команда която локализира бинарките, източници и наръчници за команда.
Командата find:

find .	Намира всичко в директорията и под нея

find {directory name}	Намирате всички директории свързани с нея.
find . –type d	Ще намери всички директории без файлове.
find . –type f	Търси само файлове.
find . –type f –name {filename}	Търси файлове по дадено име.
sudo find -type f -name “{filrename}*”	Търи файла глобирайки го (wildcard)
sudo find . -type f -iname “{filename*}”	Търси файла и премахва case sensitive случаите.
find -type f -iname "*.txt"	Може и с глобиране.
find . -type f -mmin -10	Търси последно модифицираните файлове примера е с 10 минути.
find -mmtime	Опцията която търси време дни за модификацийте.
find -size option find . -size +5M	Търси по зададен размер.
find . –empty	Търси само празните файлове – без съдържание.
find –perm 777 или 666	Търси по зададени права.
find webin -exec chown ivan:sudo {} +	Тръси даден файл и му прилага права или команда. Тук е с промяна на групата.
find webin -type f -exec chmod 660 {} +	Тръси даден файл и му прилага права или команда. Тук е с промяна на правата.

Регулярни изрази:

Регулярните изрази позволяват фина спецификация на търсенията и се поддържат от множество популярни команди и езици за програмиране.

• grep g..m /etc/passwd .

• Точката обозначава един знак.

• ^ търси в началото на линията.

• $ търси в края на линията.

• Може определени комбинации.

• grep –i прави grep не чуствителен към знаците.

• grep ^[Aa].[Aa] /etc/passwd .

• man 7 regex .

Wildcards (file globbing) - предлагат лесен начин за подаване на множество имена на файлове като аргументи на команди (програми)

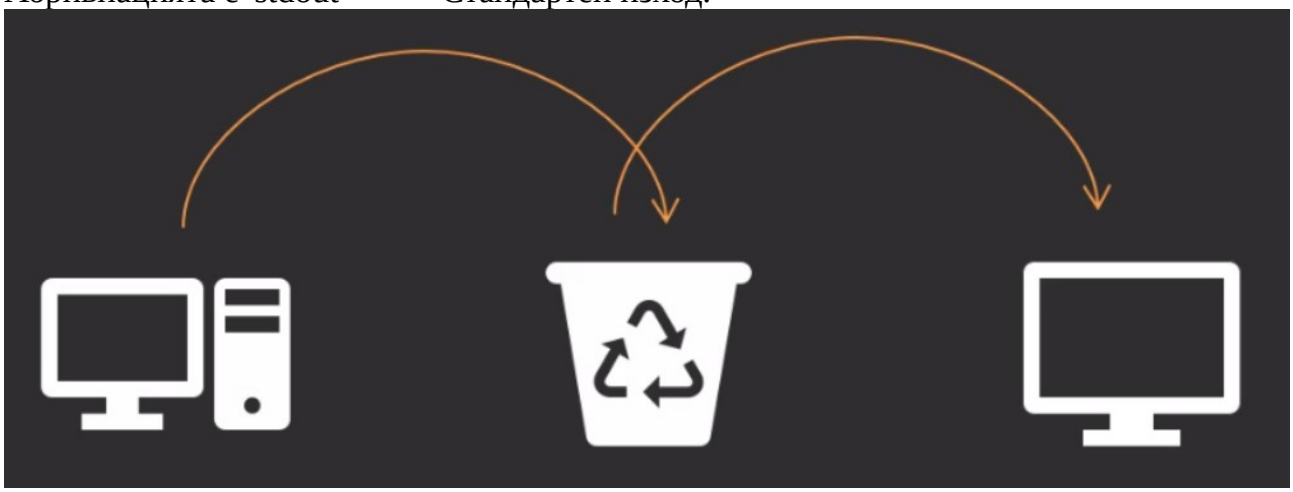
• Wildcards се указват със специални символи:

• ? – единичен символ.

• * - нула или повече последователни символа.

• [...] – класове и обхвати {...} – списък символни низове.

Във всички UNIX базирани системи имат нещо като “кофа” за всичкият изход от команди. “Кофата” се нарича още standard output.
Абривиацията е 'stdout' ===> Стандартен изход.



Пример:
Това което правя оттдолу е прескачане на “кофата” и прехвърлянето на информацията от ls -lah командата във файл с име bucket.

```
nick@xmachine:~/Test/bucket$ ls -lah
total 8,0K
drwxrwxr-x  2 nick nick 4,0K окт 20 18:33 .
drwxrwxr-x 10 nick nick 4,0K окт 20 18:33 ..
nick@xmachine:~/Test/bucket$ ls -lah > bucket
nick@xmachine:~/Test/bucket$ cat bucket
total 8,0K
drwxrwxr-x  2 nick nick 4,0K окт 20 18:33 .
drwxrwxr-x 10 nick nick 4,0K окт 20 18:33 ..
-rw-rw-r--  1 nick nick    0 окт 20 18:33 bucket
nick@xmachine:~/Test/bucket$
```

Също можем да извършваме във една комада действието с piping:

Piping – позволява STDOUT (изход) на една програма (лявата на pipe) да стане STDIN (вход) на
друга (дясната на pipe).

Символът за pipe е “|”.

tee – позволява едновременно запис към STDOUT и към файл.

Стандартен вход:

Стандартиният вход (standard input) е най често зададен от нас използвайки клавиатурата – пишейки командите.
Но може да бъде и от приложение/deamon. Файлове и стандартен вход могат да се ползват в стандартният вход на други команди:

• със <, |

• Абривиация – ‘stdin’

• Примерно wc < test.sh.

• cat /etc/passwd | less

Стандартна Грешка:

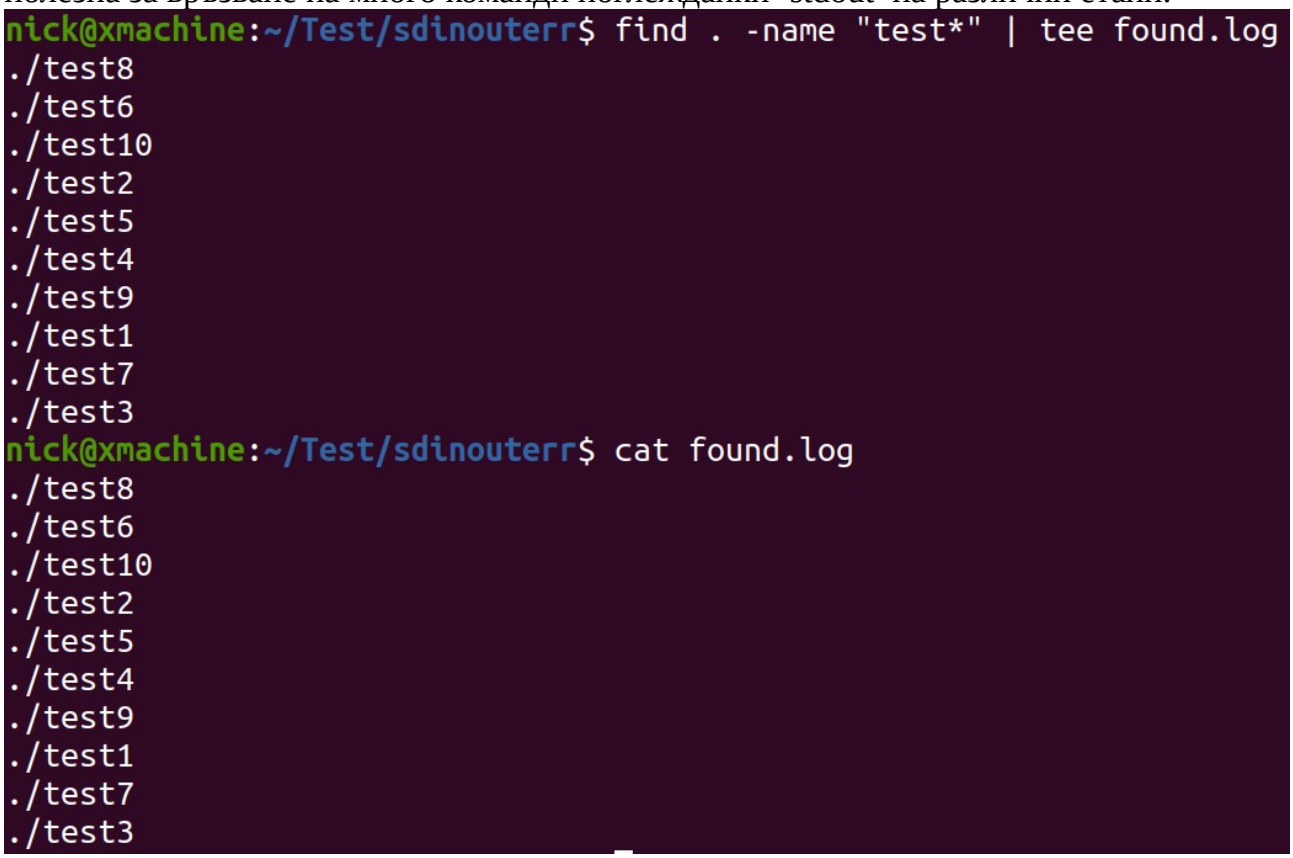
Индфикационнен номер ако нещата не са направени както трявбва.

- Абривиация ‘stderr’
- Попринцип е излиза на екрана като стандартен изход
- Има ‘файлова дръжка’ – цифра с която е свързана – 2
- stdin е 0
- stdout е 1
- ./script.sh == виждаме грешката директно на екрана
- ./script.sh 2> error.log пращаме грешката във файл
- ./script.sh 2>&1 | less == отваряме грешката дирекнто на екрана с less

```
nick@xmachine:~/Test/sdinouterr$ ls -lah
total 8,0K
drwxrwxr-x  2 nick nick 4,0K окт 20 18:42 .
drwxrwxr-x 11 nick nick 4,0K окт 20 18:42 ..
nick@xmachine:~/Test/sdinouterr$ echo $?
0
nick@xmachine:~/Test/sdinouterr$ ls -lah something
ls: cannot access 'something': No such file or directory
nick@xmachine:~/Test/sdinouterr$ echo $?
2
nick@xmachine:~/Test/sdinouterr$ ls -lah something 2> error.log
nick@xmachine:~/Test/sdinouterr$ ls -l
total 4
-rw-rw-r-- 1 nick nick 57 окт 20 18:43 error.log
nick@xmachine:~/Test/sdinouterr$ cat error.log
ls: cannot access 'something': No such file or directory
nick@xmachine:~/Test/sdinouterr$ ls -lah something 2>&1 error.log
ls: cannot access 'something': No such file or directory
-rw-rw-r-- 1 nick nick 57 окт 20 18:43 error.log
```

Tee командата:

Командата ‘tee’ чете данните от ‘stdin’ и записва данните в ‘stdout’. Командата е много полезна за връзване на много команди поглеждайки ‘stdout’ на различни етапи.



```
nick@xmachine:~/Test/sdinouterr$ find . -name 'test*' | tee found.log | sort -n | tee sorted.log
./test1
./test10
./test2
./test3
./test4
./test5
./test6
./test7
./test8
./test9
nick@xmachine:~/Test/sdinouterr$ cat sorted.log
./test1
./test10
./test2
./test3
./test4
./test5
./test6
./test7
./test8
./test9
```

tr командата:

Конвертира, извлича и изтрива символи и знаци.

Примера показва трансформация от малки символи към големи.

```
nick@xmachine:~/Test/tr$ ls -l
total 4
-rw-rw-r-- 1 nick nick 52 окт 20 18:57 fime_example.txt
nick@xmachine:~/Test/tr$ cat fime_example.txt
Welcome to this test file.
Hello there.
What is up.
nick@xmachine:~/Test/tr$ cat fime_example.txt | tr "[a-z]" "[A-Z]"
WELCOME TO THIS TEST FILE.
HELLO THERE.
WHAT IS UP.
```

Същия пример – друг метод на изписване:

```
nick@xmachine:~/Test/tr$ cat fime_example.txt | tr "[:lower:]" "[:upper:]"
WELCOME TO THIS TEST FILE.
HELLO THERE.
WHAT IS UP.
```

Пример в който превръщам space в табулация:

```
nick@xmachine:~/Test/tr$ echo "WELCOME TO THE TEST FILE" | tr [:space:] '\t'
WELCOME TO      THE     TEST    FILE    nick@xmachine:~/Test/tr$
```

Трансформация и записването и в нов файл.

```
nick@xmachine:~/Test/tr$ echo "WELCOME TO THE TEST FILE" | tr [:space:] '\t' > file
nick@xmachine:~/Test/tr$ cat fi
file             fime_example.txt
nick@xmachine:~/Test/tr$ cat file
WELCOME TO      THE     TEST    FILE    nick@xmachine:~/Test/tr$
```

Без опции - конвертиране от един набор символи към друг:

Sort командата:

Сортира по азбучен ред думи и др.

```
nick@xmachine:~/Test/sort$ ls -l
total 0
-rw-rw-r-- 1 nick nick 0 окт 20 19:10 file
nick@xmachine:~/Test/sort$ cat file
nick@xmachine:~/Test/sort$ vim file
nick@xmachine:~/Test/sort$ cat file
Fish
Food
Chips
Chrisps
Potato
Banana
nick@xmachine:~/Test/sort$ sort file
Banana
Chips
Chrisps
Fish
Food
Potato
```

Пример с прехвърляне на изхода:

```
nick@xmachine:~/Test/sort$ sort file > file2
nick@xmachine:~/Test/sort$ cat file2
Banana
Chips
Chrisps
Fish
Food
Potato
nick@xmachine:~/Test/sort$ cat file
Fish
Food
Chips
Chrisps
Potato
Banana
```

Сортиране в обърнат ред:

```
nick@xmachine:~/Test/sort$ sort file > file2
nick@xmachine:~/Test/sort$ cat file2
Banana
Chips
Chrisps
Fish
Food
Potato
nick@xmachine:~/Test/sort$ cat file
Fish
Food
Chips
Chrisps
Potato
Banana
```

Случаен принцип:

```
nick@xmachine:~/Test/sort$ sort -R file
Food
Chips
Banana
Potato
Chrisps
Fish
```

Сортиране на определени от нас колони:

```
nick@xmachine:~/Test/sort$ cat file
Fish Movies
Food Cine
Chips Kino
Chrisps Cinematic
Potato Movie
Banana Film
nick@xmachine:~/Test/sort$ sort -k 1 file
Banana Film
Chips Kino
Chrisps Cinematic
Fish Movies
Food Cine
Potato Movie
nick@xmachine:~/Test/sort$ sort -k 2 file
Food Cine
Chrisps Cinematic
Banana Film
Chips Kino
Potato Movie
Fish Movies
```

cut:
Извлича избрани полета от редове текст (използва табулация за разделител по подразбиране). Работи най-добре със структуриран текст (текст с колони).

```
nick@xmachine:~/Test/cut$ cut -d: -f 1,3 passwd
```

```
sync:4
games:5
man:6
lp:7
mail:8
news:9
uucp:10
proxy:13
www-data:33
backup:34
list:38
irc:39
gnats:41
nobody:65534
systemd-network:100
systemd-resolve:101
systemd-timesync:102
messagebus:103
syslog:104
_apt:105
tss:106
uuidd:107
tcpdump:108
avahi-autoipd:109
usbmux:110
rtkit:111
dnsmasq:112
cups-pk-helper:113
speech-dispatcher:114
avahi:115
```