

# **LINUX COMMANDS**

**Print Commands in Terminal: echo – printf****echo and printf**

Command	echo [option] [string] / printf [option] [string]
Description	<p>The echo and printf commands are used to generate text output in the terminal. The echo command can be used to print simple text to the screen, while the printf command offers more complex formatting options.</p> <p>The "&gt;" operator is used to print to a file, The "&gt;&gt;" operator is used to append to the end of the current file.</p>

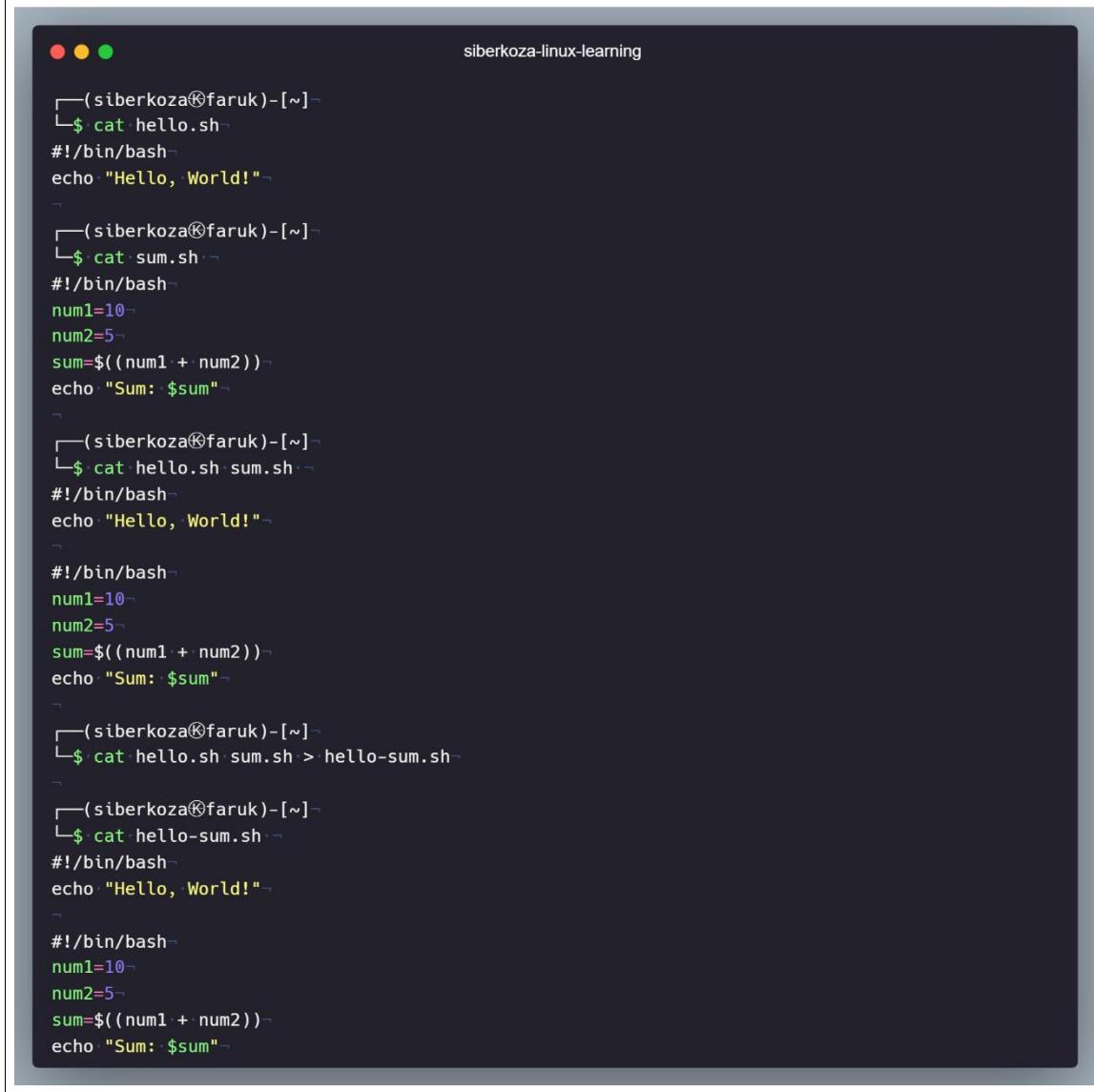


The screenshot shows a terminal window titled "siberkoza-linux-learning". It displays several examples of the echo and printf commands:

```
(siberkoza㉿faruk)~$ echo "Hello World!"  
Hello World!  
  
(siberkoza㉿faruk)~$ printf "Hello World!"  
Hello World!  
  
(siberkoza㉿faruk)~$ echo "Some Text">> test.txt  
  
(siberkoza㉿faruk)~$ cat test.txt  
Some Text  
  
(siberkoza㉿faruk)~$ echo "Some Text - New">>> test.txt  
  
(siberkoza㉿faruk)~$ cat test.txt  
Some Text  
Some Text - New
```

**Commands to View File Contents:** cat - less - more - head - tail**cat**

Command	cat
Usage Methods	cat [options] filename cat file1 file2 cat file1 file2 > newfile
Description	The “cat” command combines the contents of one or more files and prints them on the screen. It is often used to display the contents of short files.



```

siberkoza@faruk:~$ cat hello.sh
#!/bin/bash
echo "Hello, World!"

siberkoza@faruk:~$ cat sum.sh
#!/bin/bash
num1=10
num2=5
sum=$((num1+ num2))
echo "Sum: $sum"

siberkoza@faruk:~$ cat hello.sh sum.sh
#!/bin/bash
echo "Hello, World!"

#!/bin/bash
num1=10
num2=5
sum=$((num1+ num2))
echo "Sum: $sum"

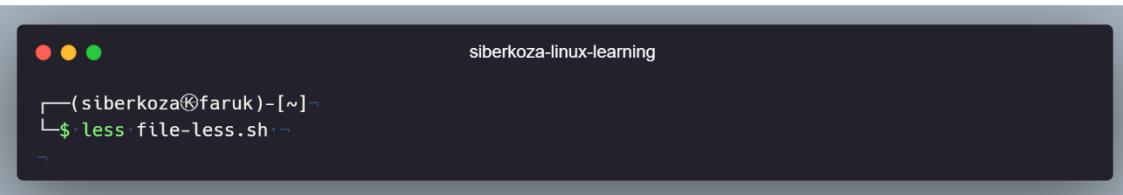
siberkoza@faruk:~$ cat hello.sh sum.sh > hello-sum.sh
siberkoza@faruk:~$ cat hello-sum.sh
#!/bin/bash
echo "Hello, World!"

#!/bin/bash
num1=10
num2=5
sum=$((num1+ num2))
echo "Sum: $sum"

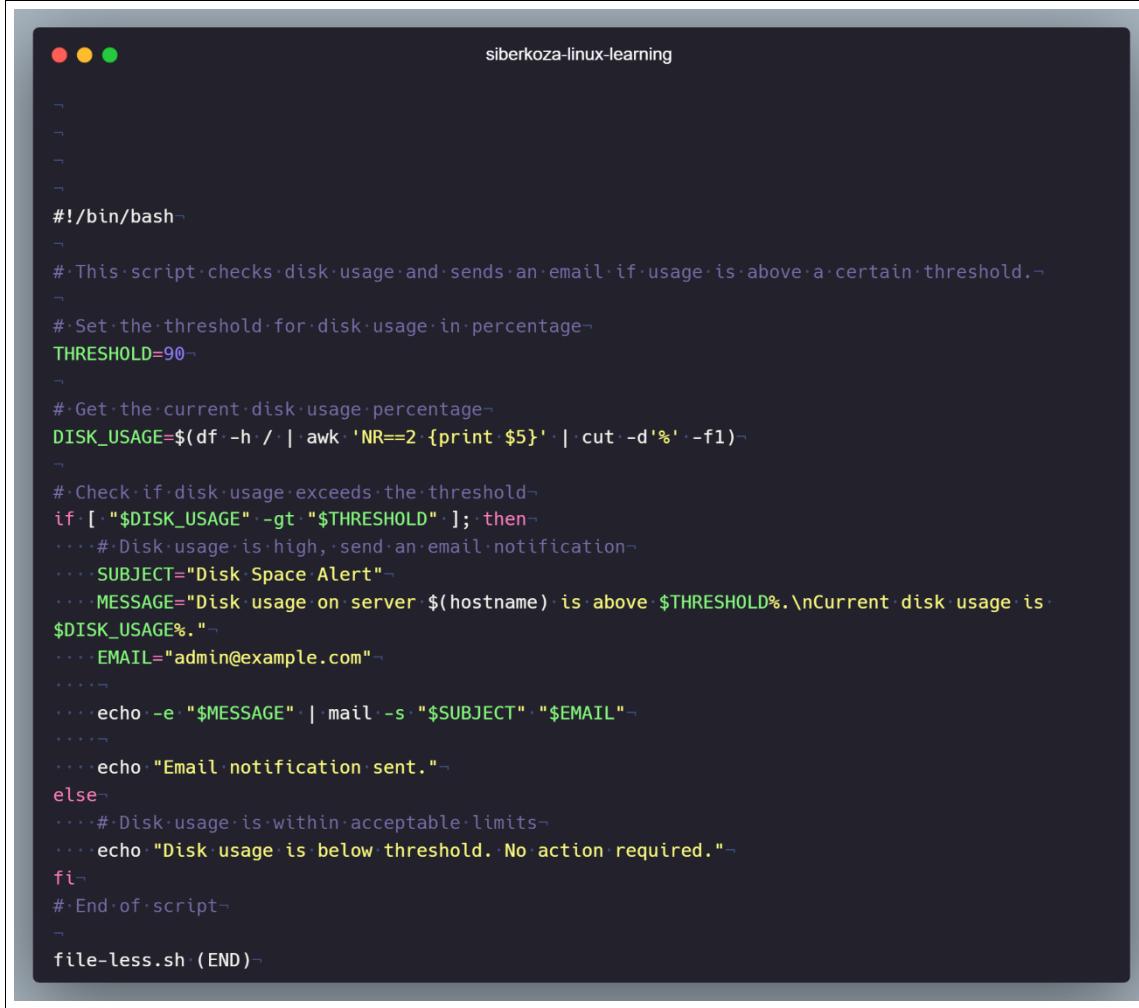
```

**less**

Command	less
Usage Methods	less [options] filename
Description	The “less” command displays the contents of the file page by page to fit the size of the screen. It may be more convenient for analysing long files, because the user can scroll up and down at will.



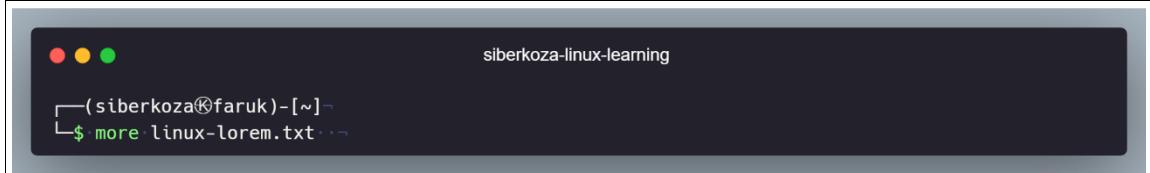
```
(siberkoza@faruk) [~]
$ less file-less.sh
```



```
#!/bin/bash
# This script checks disk usage and sends an email if usage is above a certain threshold.
# Set the threshold for disk usage in percentage
THRESHOLD=90
# Get the current disk usage percentage
DISK_USAGE=$(df -h / | awk 'NR==2 {print $5}' | cut -d '%' -f1)
# Check if disk usage exceeds the threshold
if [ "$DISK_USAGE" -gt "$THRESHOLD" ]; then
    # Disk usage is high, send an email notification
    SUBJECT="Disk Space Alert"
    MESSAGE="Disk usage on server $(hostname) is above $THRESHOLD%. Current disk usage is $DISK_USAGE%."
    EMAIL="admin@example.com"
    echo -e "$MESSAGE" | mail -s "$SUBJECT" "$EMAIL"
    echo "Email notification sent."
else
    # Disk usage is within acceptable limits
    echo "Disk usage is below threshold. No action required."
fi
# End of script
file-less.sh (END)
```

**more**

Command	more
Usage Methods	more [options] [num] [+pattern] [+linenum] [filename]
Description	The “more” command displays the contents of the file page by page to fit the size of the screen. It works like “less” but without the scroll-back feature.



```
siberkoza-linux-learning

(siberkoza㉿faruk)-[~] ~
└─$ more linux-Lorem.txt

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Purus faucibus ornare suspendisse sed nisi lacus sed viverra tellus. Pellentesque massa placerat dui ultricies lacus sed turpis tincidunt id. Eget mauris pharetra et ultrices neque ornare. Nullam eget felis eget nunc lobortis mattis aliquam faucibus purus. Integer vitae justo eget magna fermentum iaculis. Porttitor eget dolor morbi non arcu risus quis varius. At erat pellentesque adipiscing commodo elit at. Nisi lacus sed viverra tellus in hac habitasse platea. In hac habitasse platea dictumst vestibulum rhoncus est pellentesque elit. Aenean pharetra magna ac placerat vestibulum lectus mauris ultrices. Consectetur purus ut faucibus pulvinar elementum integer. Praesent semper feugiat nibh sed pulvinar. Mollis aliquam ut porttitor leo. Eu turpis egestas pretium aenean pharetra magna ac. Diam maecenas sed enim ut sem viverra aliquet eget sit.

Adipiscing at in tellus integer feugiat scelerisque varius. Blandit aliquam etiam erat velit scelerisque in. Risus pretium quam vulputate dignissim suscipit in est ante. Vestibulum lectus mauris ultrices eros in cursus turpis massa tincidunt. At lectus urna duis convallis convallis tellus id interdum. Viverr a orci sagittis eu volutpat odio facilisis mauris sit. Sed velit dignissim sodales ut eu sem. Aliquam nulla facilisi cras fermentum. Fringilla est ullamcorper eget nulla facilisi. Diam maecenas sed enim ut sem viverra aliquet eget sit. Duis ultricies lacus sed turpis. Vel turpis nunc eget lorem dolor. Facilisis mauris sit amet massa. Lacus viverra vitae congue eu consequat ac felis donec. Ornare aenean euismod elementum nisi. Porta nibh venenatis cras sed.

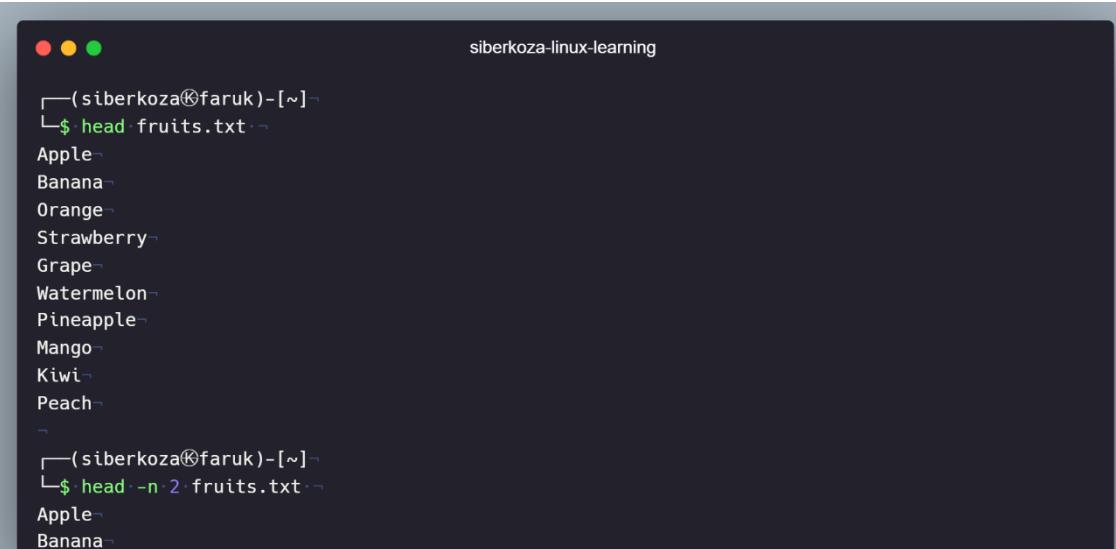
Pellentesque massa placerat dui ultricies lacus sed turpis. Interdum consectetur libero id faucibus. Nibh cras pulvinar mattis nunc sed blandit libero volutpat sed. Tortor posuere ac ut consequat semper. Tellus in metus vulputate eu scelerisque felis imperdiet proin. Amet mattis vulputate enim nulla aliquet posuere. More--(13%)
```

## head

Command	head
Usage Methods	head [options] [filename]
Description	The “head” command displays the head of a file. By default, it displays the first 10 lines.



```
$ ./fruits.txt
Apple
Banana
Orange
Strawberry
Grape
Watermelon
Pineapple
Mango
Kiwi
Peach
Pear
Cherry
Raspberry
Blueberry
Plum
Lemon
Lime
Coconut
Avocado
Papaya
```

```
└─(siberkoza@faruk)~
└$ head fruits.txt
Apple
Banana
Orange
Strawberry
Grape
Watermelon
Pineapple
Mango
Kiwi
Peach
└
└─(siberkoza@faruk)~
└$ head -n 2 fruits.txt
Apple
Banana
```

**tail**

Command	tail
Usage Methods	tail [options] [filename]
Description	The “tail” command displays the last part of a file. By default, it displays the last 10 lines.

```
siberkoza-linux-learning

$ fruits.txt
Apple
Banana
Orange
Strawberry
Grape
Watermelon
Pineapple
Mango
Kiwi
Peach
Pear
Cherry
Raspberry
Blueberry
Plum
Lemon
Lime
Coconut
Avocado
Papaya
```

```
siberkoza-linux-learning

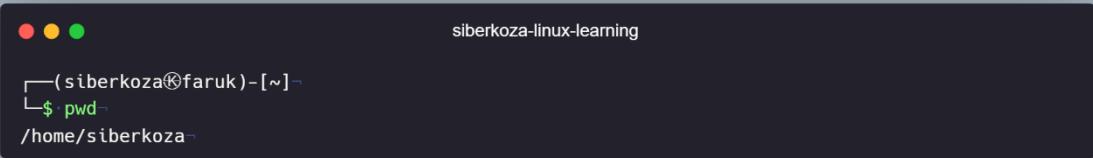
└─(siberkoza@faruk)-[~]─
└─$ tail fruits.txt
Pear
Cherry
Raspberry
Blueberry
Plum
Lemon
Lime
Coconut
Avocado
Papaya
└─

└─(siberkoza@faruk)-[~]─
└─$ tail -n 2 fruits.txt
Avocado
Papaya
```

**Basic Linux Commands for File and Directory Operations:** [pwd](#) - [cd](#) - [ls](#) - [touch](#) - [file](#) - [cat](#) - [less](#) - [cp](#) - [mv](#) - [mkdir](#) - [rmdir](#) - [rm](#) - [find](#) - [cut](#) - [paste](#) - [head](#) - [tail](#) - [expand](#) and [unexpand](#) - [join](#) and [split](#) - [sort](#) - [tr](#) - [uniq](#) (unique) - [wc](#) - [nl](#) - [grep](#) - [vim](#) - [nano](#)

## [pwd](#)

Command	<code>pwd</code>
Description	The “pwd” command shows the full path to the user's current (working) directory.



```
siberkoza-linu~(siberkoza@faruk)-[~]-
$ pwd
/home/siberkoza-
```

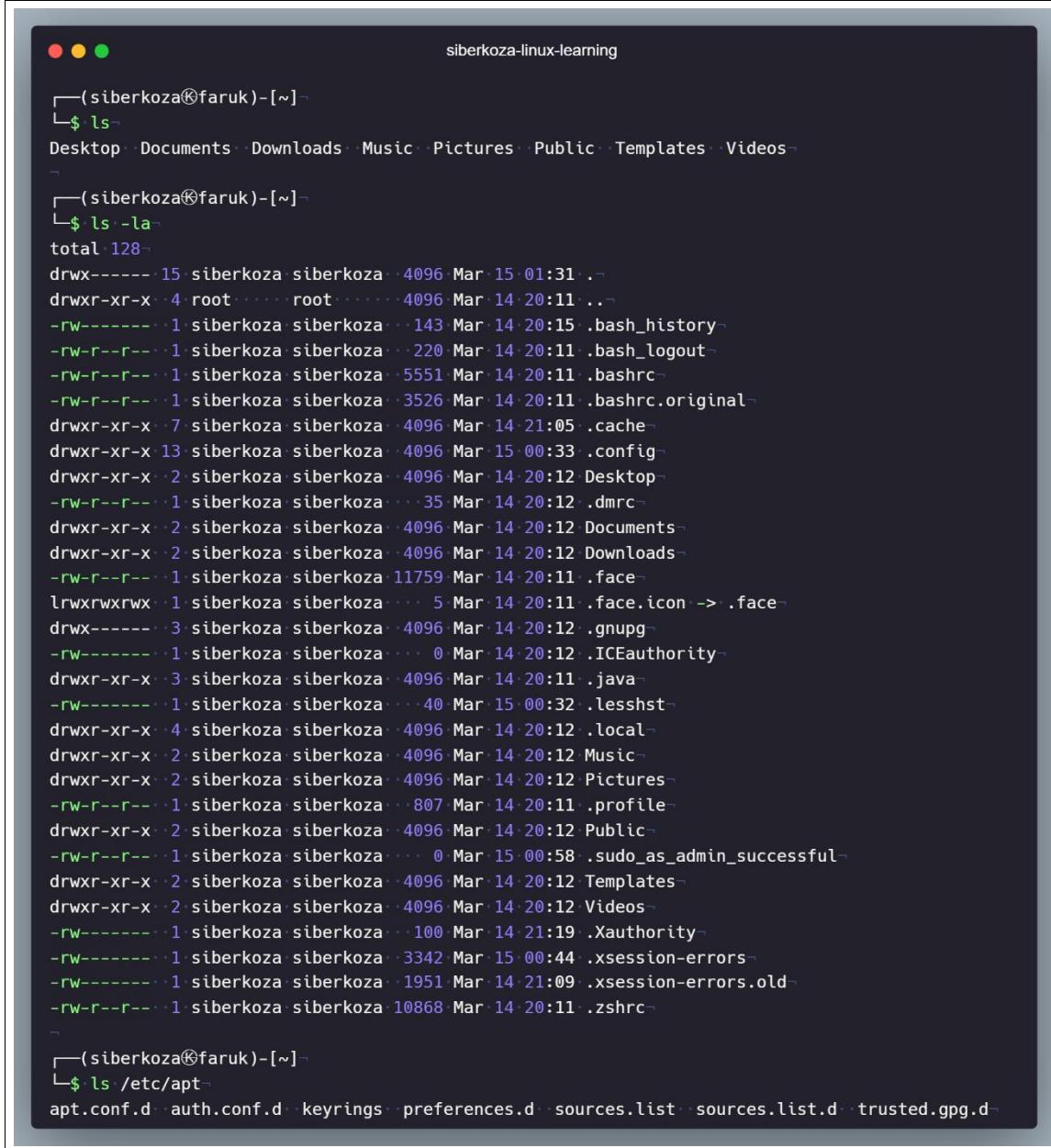
## [cd](#)

Command	<code>cd</code>
Description	<p>The “cd” command is used to change the user's current working directory.</p> <ul style="list-style-type: none"> <li>. (current directory): Shows the current directory.</li> <li>... (parent directory): Switches to a parent directory of the current directory.</li> <li>~ (default directory): Switches to the default directory. "e.g.: home/user"</li> <li>- (old directory): Returns to the previous directory</li> <li>/ (home directory): Switches to the main directory</li> </ul>

Command	<code>cd</code>
Description	<pre>siberkoza-linu~(siberkoza@faruk)-[~]- \$ cd . - (siberkoza@faruk)-[~]- \$ cd .. - (siberkoza@faruk)-[~]- \$ cd - / - (siberkoza@faruk)-[~]- \$ cd ./home/siberkoza/Desktop/ - (siberkoza@faruk)-[~/Desktop]- \$ cd ... - (siberkoza@faruk)-[~]- \$ .</pre>

**ls**

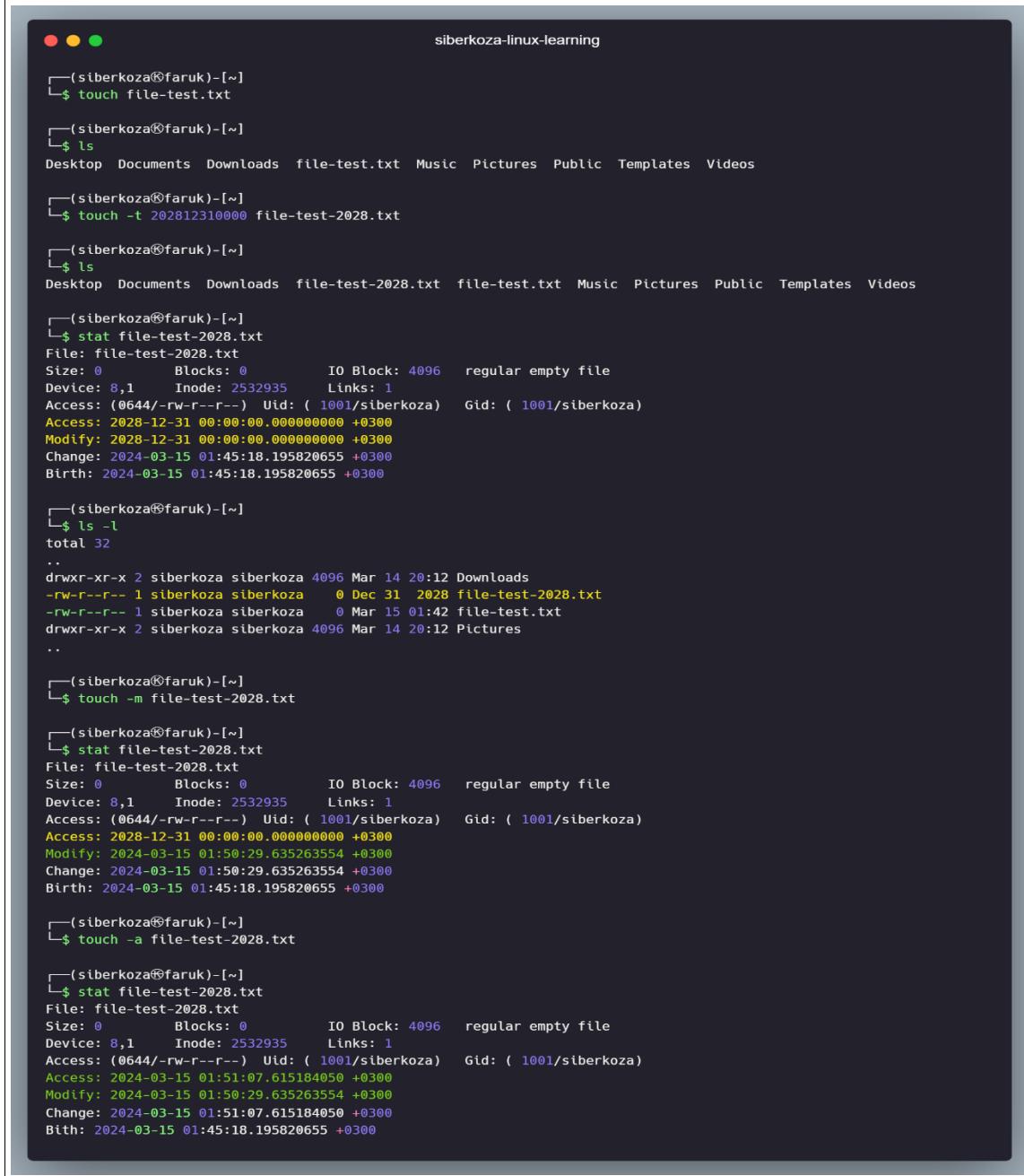
<b>Command</b>	ls
<b>Description</b>	<p>The “cd” command is used to change the user's current working directory.</p> <p>ls -R: Lists the contents of the directory in a branched manner.</p> <p>ls -r: Lists in reverse order when sorting.</p> <p>ls -t: Sorts by modification time, newest first.</p> <p>ls -a: Also shows hidden files.</p> <p>ls -l: Provides long format output.</p>



```
siberkoza@faruk:[~]
$ ls
Desktop Documents Downloads Music Pictures Public Templates Videos
-
(siberkoza@faruk)[~]
$ ls -la
total 128
drwx----- 15 siberkoza siberkoza 4096 Mar 15 01:31 .
drwxr-xr-x  4 root      root      4096 Mar 14 20:11 ..
-rw-----  1 siberkoza siberkoza  143 Mar 14 20:15 .bash_history
-rw-r--r--  1 siberkoza siberkoza 220 Mar 14 20:11 .bash_logout
-rw-r--r--  1 siberkoza siberkoza 5551 Mar 14 20:11 .bashrc
-rw-r--r--  1 siberkoza siberkoza 3526 Mar 14 20:11 .bashrc.original
drwxr-xr-x  7 siberkoza siberkoza 4096 Mar 14 21:05 .cache
drwxr-xr-x 13 siberkoza siberkoza 4096 Mar 15 00:33 .config
drwxr-xr-x  2 siberkoza siberkoza 4096 Mar 14 20:12 Desktop
-rw-r--r--  1 siberkoza siberkoza  35 Mar 14 20:12 .dmrc
drwxr-xr-x  2 siberkoza siberkoza 4096 Mar 14 20:12 Documents
drwxr-xr-x  2 siberkoza siberkoza 4096 Mar 14 20:12 Downloads
-rw-r--r--  1 siberkoza siberkoza 11759 Mar 14 20:11 .face
lrwxrwxrwx  1 siberkoza siberkoza    5 Mar 14 20:11 .face.icon -> .face
drwx-----  3 siberkoza siberkoza 4096 Mar 14 20:12 .gnupg
-rw-----  1 siberkoza siberkoza    0 Mar 14 20:12 .ICEauthority
drwxr-xr-x  3 siberkoza siberkoza 4096 Mar 14 20:11 .java
-rw-----  1 siberkoza siberkoza   40 Mar 15 00:32 .lessht
drwxr-xr-x  4 siberkoza siberkoza 4096 Mar 14 20:12 .local
drwxr-xr-x  2 siberkoza siberkoza 4096 Mar 14 20:12 Music
drwxr-xr-x  2 siberkoza siberkoza 4096 Mar 14 20:12 Pictures
-rw-r--r--  1 siberkoza siberkoza  807 Mar 14 20:11 .profile
drwxr-xr-x  2 siberkoza siberkoza 4096 Mar 14 20:12 Public
-rw-r--r--  1 siberkoza siberkoza    0 Mar 15 00:58 .sudo_as_admin_successful
drwxr-xr-x  2 siberkoza siberkoza 4096 Mar 14 20:12 Templates
drwxr-xr-x  2 siberkoza siberkoza 4096 Mar 14 20:12 Videos
-rw-----  1 siberkoza siberkoza 100 Mar 14 21:19 .Xauthority
-rw-----  1 siberkoza siberkoza 3342 Mar 15 00:44 .xsession-errors
-rw-----  1 siberkoza siberkoza 1951 Mar 14 21:09 .xsession-errors.old
-rw-r--r--  1 siberkoza siberkoza 10868 Mar 14 20:11 .zshrc
-
(siberkoza@faruk)[~]
$ ls /etc/apt
apt.conf.d auth.conf.d keyrings preferences.d sources.list sources.list.d trusted.gpg.d
```

**touch**

Kullanım	<code>touch</code>
Usage Methods	<code>touch [options] filename</code> <code>touch file1 file2</code>
Açıklama	The “touch” command is used to create a file and update the modification time of the file.



```

siberkoza@faruk:[~]
└$ touch file-test.txt

siberkoza@faruk:[~]
└$ ls
Desktop Documents Downloads file-test.txt Music Pictures Public Templates Videos

siberkoza@faruk:[~]
└$ touch -t 202812310000 file-test-2028.txt

siberkoza@faruk:[~]
└$ ls
Desktop Documents Downloads file-test-2028.txt file-test.txt Music Pictures Public Templates Videos

siberkoza@faruk:[~]
└$ stat file-test-2028.txt
File: file-test-2028.txt
Size: 0          Blocks: 0          IO Block: 4096   regular empty file
Device: 8,1      Inode: 2532935    Links: 1
Access: (0644/-rw-r--r--)  Uid: ( 1001/siberkoza)  Gid: ( 1001/siberkoza)
Access: 2028-12-31 00:00:00.000000000 +0300
Modify: 2028-12-31 00:00:00.000000000 +0300
Change: 2024-03-15 01:45:18.195820655 +0300
Birth: 2024-03-15 01:45:18.195820655 +0300

siberkoza@faruk:[~]
└$ ls -l
total 32
..
drwxr-xr-x 2 siberkoza siberkoza 4096 Mar 14 20:12 Downloads
-rw-r--r-- 1 siberkoza siberkoza    0 Dec 31  2028 file-test-2028.txt
-rw-r--r-- 1 siberkoza siberkoza    0 Mar 15 01:42 file-test.txt
drwxr-xr-x 2 siberkoza siberkoza 4096 Mar 14 20:12 Pictures
..

siberkoza@faruk:[~]
└$ touch -m file-test-2028.txt

siberkoza@faruk:[~]
└$ stat file-test-2028.txt
File: file-test-2028.txt
Size: 0          Blocks: 0          IO Block: 4096   regular empty file
Device: 8,1      Inode: 2532935    Links: 1
Access: (0644/-rw-r--r--)  Uid: ( 1001/siberkoza)  Gid: ( 1001/siberkoza)
Access: 2028-12-31 00:00:00.000000000 +0300
Modify: 2024-03-15 01:50:29.635263554 +0300
Change: 2024-03-15 01:50:29.635263554 +0300
Birth: 2024-03-15 01:45:18.195820655 +0300

siberkoza@faruk:[~]
└$ touch -a file-test-2028.txt

siberkoza@faruk:[~]
└$ stat file-test-2028.txt
File: file-test-2028.txt
Size: 0          Blocks: 0          IO Block: 4096   regular empty file
Device: 8,1      Inode: 2532935    Links: 1
Access: (0644/-rw-r--r--)  Uid: ( 1001/siberkoza)  Gid: ( 1001/siberkoza)
Access: 2024-03-15 01:51:07.615184050 +0300
Modify: 2024-03-15 01:50:29.635263554 +0300
Change: 2024-03-15 01:51:07.615184050 +0300
Birth: 2024-03-15 01:45:18.195820655 +0300

```

**file**

Command	file
Usage Methods	file [options] filename file *
Description	The “file” command analyses the contents of the file to determine its type and provides this information to the user.

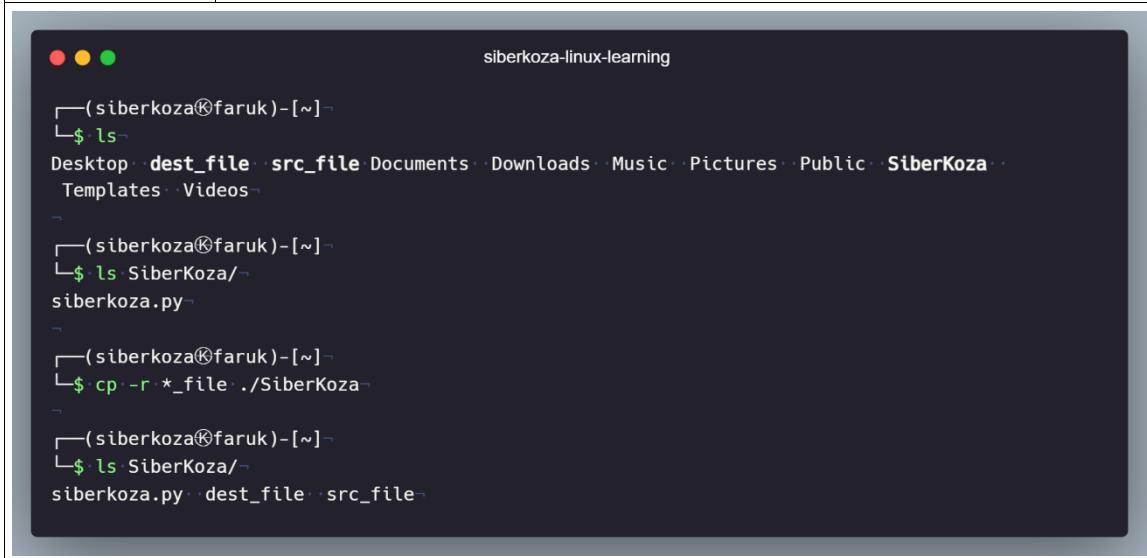


```
siberkoza-linuX-learning
(siberkoza@faruk)-[~/Downloads]
$ file turkey-flag.gif
turkey-flag.gif: GIF image data, version 89a, 461 x 458

(siberkoza@faruk)-[~/Downloads]
$ file *
main.py: ..... Python script, ASCII text executable
Turkey_flag_300.png: PNG image data, 450 x 300, 8-bit/color RGBA, non-interlaced
turkey-flag.gif: .... GIF image data, version 89a, 461 x 458
```

**cp**

Command	cp
Usage Methods	cp [options] [source_file] [destination]
Description	The “cp” command is a command used to copy files and directories.



```
siberkoza-linuX-learning
(siberkoza@faruk)-[~]
$ ls
Desktop dest_file src_file Documents Downloads Music Pictures Public SiberKoza Templates Videos

(siberkoza@faruk)-[~]
$ ls SiberKoza/
siberkoza.py

(siberkoza@faruk)-[~]
$ cp -r *_file ./SiberKoza/
(siberkoza@faruk)-[~]
$ ls SiberKoza/
siberkoza.py dest_file src_file
```

**mv**

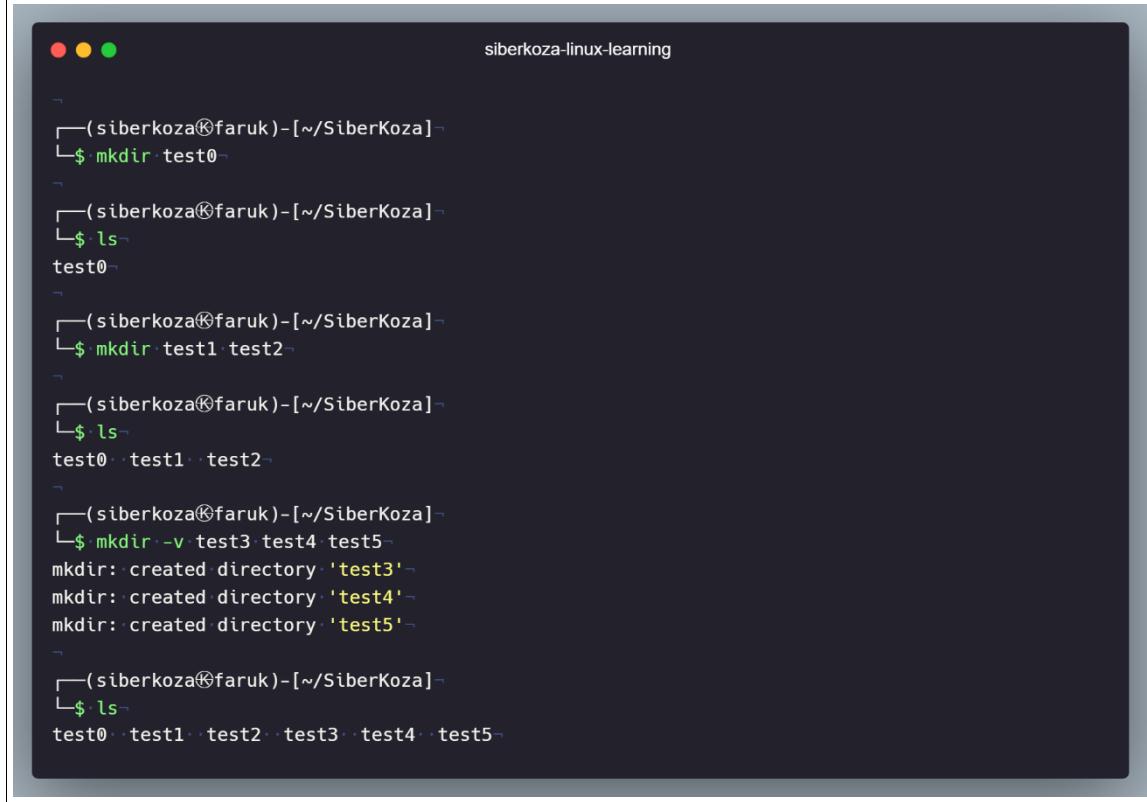
Command	mv
Usage Methods	mv [options] [source_file] [destination_file]
Description	The “mv” command is used to move or rename files and directories.

```
siberkoza@faruk:~$ ls -l
total 0
siberkoza@faruk:~$ mv src_file dest_file
siberkoza@faruk:~$ ls -l
total 0
siberkoza@faruk:~$ mv dest_file src_file
siberkoza@faruk:~$ ls -l
total 0
siberkoza@faruk:~$ mv *_file ./SiberKoza
siberkoza@faruk:~$ ls SiberKoza/
siberkoza.py
siberkoza@faruk:~$ ls
total 0
```

```
siberkoza@faruk:~$ ls -l
total 0
siberkoza@faruk:~$ mv old-test.txt new-file.txt
siberkoza@faruk:~$ ls
total 0
```

## **mkdir**

Command	<code>mkdir</code>
Usage Methods	<code>mkdir [options] [directory_name]</code>
Description	The “mkdir” command is used to create new directories.



```
siberkoza-linux-learning

(siberkoza@faruk) [~/SiberKoza]
$ mkdir test0

(siberkoza@faruk) [~/SiberKoza]
$ ls
test0

(siberkoza@faruk) [~/SiberKoza]
$ mkdir test1 test2
(siberkoza@faruk) [~/SiberKoza]
$ ls
test0 test1 test2

(siberkoza@faruk) [~/SiberKoza]
$ mkdir -v test3 test4 test5
mkdir: created directory 'test3'
mkdir: created directory 'test4'
mkdir: created directory 'test5'

(siberkoza@faruk) [~/SiberKoza]
$ ls
test0 test1 test2 test3 test4 test5
```

## rmdir

Command	rmdir
Usage Methods	rmdir [options] [directory]
Description	The “rmdir” command is used to delete an empty directory. It cannot delete non-empty directories.

```
siberkoza-linux-learning

└──(siberkoza㉿faruk)-[~/SiberKoza] ~
└─$ rmdir test0

└──(siberkoza㉿faruk)-[~/SiberKoza] ~
└─$ ls
test1 test2 test3 test4 test5

└──(siberkoza㉿faruk)-[~/SiberKoza] ~
└─$ rmdir test1 test2

└──(siberkoza㉿faruk)-[~/SiberKoza] ~
└─$ ls
test3 test4 test5

└──(siberkoza㉿faruk)-[~/SiberKoza] ~
└─$ rmdir -v test*
rmdir: removing directory, 'test3'
rmdir: removing directory, 'test4'
rmdir: removing directory, 'test5'

└──(siberkoza㉿faruk)-[~/SiberKoza] ~
└─$ ls
..... #####
##### EMPTY #####
.....
```

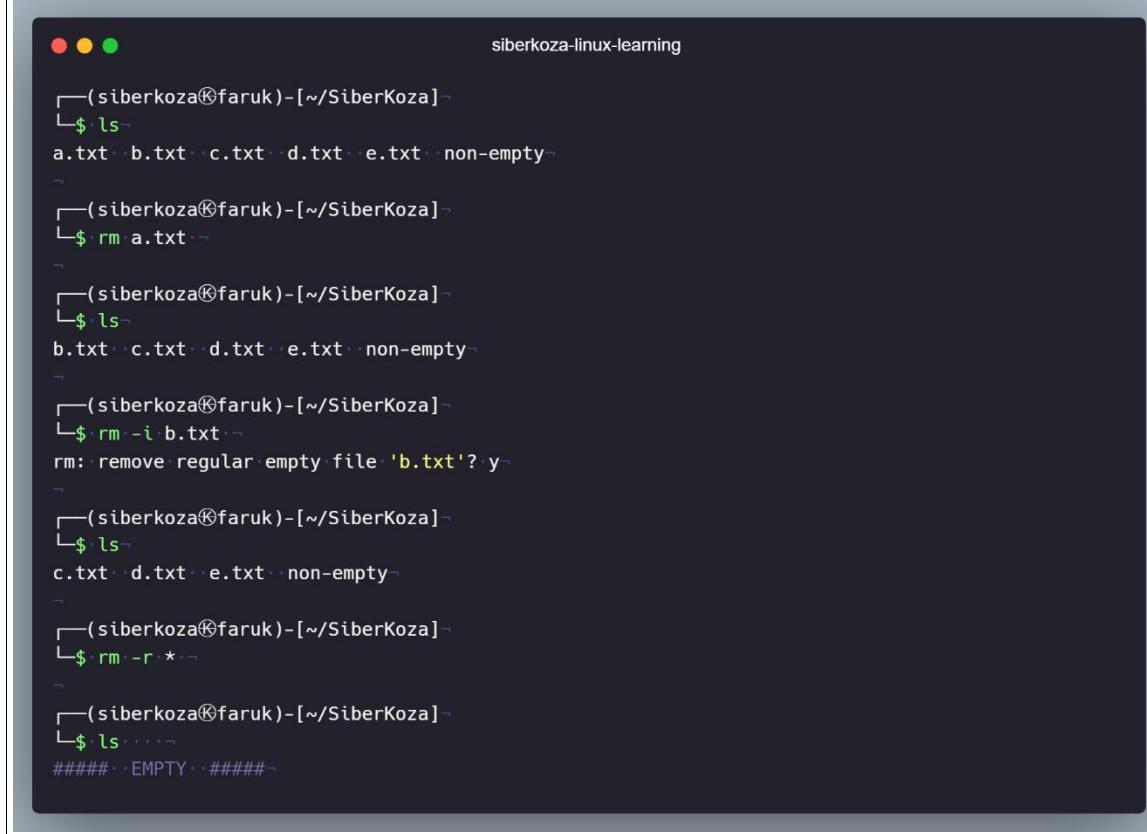
```
siberkoza-linux-learning

└──(siberkoza㉿faruk)-[~/SiberKoza] ~
└─$ ls
non-empty

└──(siberkoza㉿faruk)-[~/SiberKoza] ~
└─$ cd ..
└──(siberkoza㉿faruk)-[~] ~
└─$ rmdir SiberKoza
rmdir: failed to remove 'SiberKoza/': Directory not empty
```

**rm**

Command	rm
Usage Methods	rm [options] [directory]
Description	<p>The “rm” command is used to delete files and directories.</p> <ul style="list-style-type: none"><li>• -i, --interactive: To ask for confirmation before deletion.</li><li>• -f, --force: Makes deletion more difficult, does not ask for confirmation.</li><li>• -r, -R, --recursive: Deletes directories and their contents.</li><li>• -v, --verbose: Show detailed output during the operation.</li></ul>



A screenshot of a terminal window titled "siberkoza-linux-learning". The terminal shows a user named "siberkoza" at a prompt "(siberkoza㉿faruk)". Inside the terminal, the user runs several commands:

- \$ ls - lists files: a.txt, b.txt, c.txt, d.txt, e.txt, non-empty
- \$ rm a.txt - removes file "a.txt"
- \$ ls - lists files: b.txt, c.txt, d.txt, e.txt, non-empty
- \$ rm -i b.txt - removes file "b.txt" with confirmation ("rm: remove regular empty file 'b.txt'? y")
- \$ ls - lists files: c.txt, d.txt, e.txt, non-empty
- \$ rm -r \* - removes all files in the directory (including "non-empty")
- \$ ls - lists files: EMPTY

## find

Command	find
Usage Methods	find [path] [options] [expression]
Description	The find command is used to search for files and directories according to certain criteria.

```
siberkoza@faruk:[~]
└$ find /etc/apt -type f
/etc/apt/sources.list
/etc/apt/apt.conf.d/01autoremove
/etc/apt/apt.conf.d/02autoremove-postgresql
/etc/apt/apt.conf.d/80debtags
/etc/apt/apt.conf.d/99needrestart
/etc/apt/apt.conf.d/50command-not-found
/etc/apt/apt.conf.d/70debconf
/etc/apt/apt.conf.d/50kali
/etc/apt/apt.conf.d/50apt-file.conf
/etc/apt/trusted.gpg.d/debian-archive-buster-security-automatic.asc
/etc/apt/trusted.gpg.d/debian-archive-buster-stable.asc
/etc/apt/trusted.gpg.d/debian-archive-bullseye-automatic.asc
/etc/apt/trusted.gpg.d/debian-archive-bullseye-security-automatic.asc
/etc/apt/trusted.gpg.d/debian-archive-bookworm-security-automatic.asc
/etc/apt/trusted.gpg.d/debian-archive-bullseye-stable.asc
/etc/apt/trusted.gpg.d/debian-archive-buster-automatic.asc
/etc/apt/trusted.gpg.d/debian-archive-bookworm-automatic.asc
/etc/apt/trusted.gpg.d/debian-archive-bookworm-stable.asc
```

```
siberkoza@faruk:[~]
└$ ./SiberKoza/
./SiberKoza/
./SiberKoza/find-user.sh
./SiberKoza/user-01.txt
./SiberKoza/user-01.png
.

(siberkoza@faruk)[~]
└$ find ./SiberKoza -name "*.sh"
./SiberKoza/find-user.sh
```

**cut**

Command	cut
Usage Methods	cut [options] [file]
Description	The “cut” command is used to cut specific sections from lines.

```
siberkoza-linu...x-learning

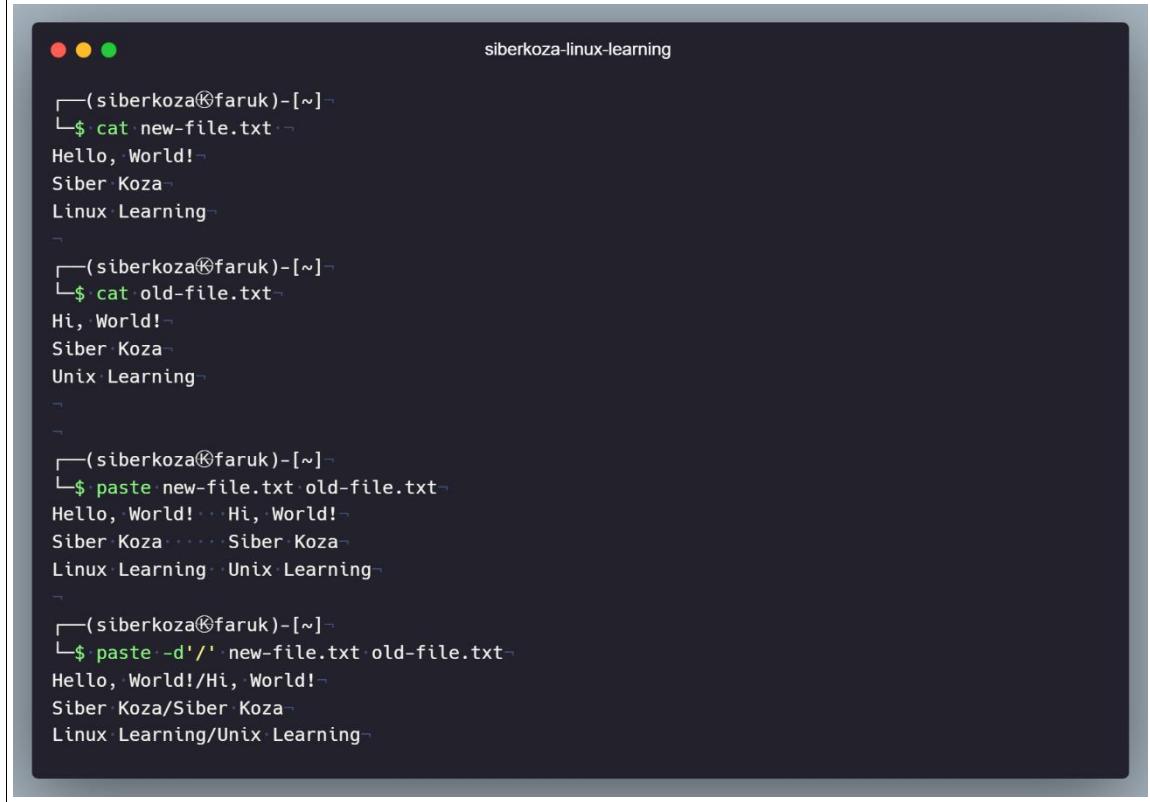
└─(siberkoza@faruk)─[~]─
└$ cut -c 1-5 fruits.txt
Apple
Banan
Orang
Straw
Grape
Water
Pinea
Mango
Kiwi
Peach
Pear
Cherr
Raspb
Blueb
Plum
Lemon
Lime
Cocon
Avoca
Papay
```

```
siberkoza-linu...x-learning

└─(siberkoza@faruk)─[~]─
└$ cat new-file.txt
Hello, World!
Siber-Koza
Linux Learning
-
└─(siberkoza@faruk)─[~]─
└$ cut -d' ' -f1 new-file.txt
Hello,
Siber
Linux
-
```

**paste**

Command	<code>paste</code>
Usage Methods	<code>paste [options] [file1] [file2]</code>
Description	The <code>paste</code> command is used to merge the contents of files.



The screenshot shows a terminal window titled "siberkoza-linux-learning". It displays a sequence of commands and their outputs:

```

(siberkoza㉿faruk)~
└$ cat new-file.txt
Hello, World!
Siber Koza
Linux Learning

(siberkoza㉿faruk)~
└$ cat old-file.txt
Hi, World!
Siber Koza
Unix Learning

(siberkoza㉿faruk)~
└$ paste new-file.txt old-file.txt
Hello, World!...Hi, World!
Siber Koza...Siber Koza
Linux Learning...Unix Learning

(siberkoza㉿faruk)~
└$ paste -d'/' new-file.txt old-file.txt
Hello, World!/Hi, World!
Siber Koza/Siber Koza
Linux Learning/Unix Learning

```

**expand and unexpand**

Command	<code>expand / unexpand</code>
Usage Methods	<code>expand [filename]</code> <code>unexpand [filename]</code>
Description	The "expand" command is used to convert tab characters in a file to spaces. The "unexpand" is used to convert consecutive spaces in a file to tab characters.

**join and split**

Command	join / split
Usage Methods	join [file1] [file2] split [options] [filename]
Description	The “join” command merges two files according to their common areas. The “split” command splits a file according to a certain size or a certain number of lines.



The screenshot shows a terminal window titled "siberkoza-linux-learning". The terminal session demonstrates the use of the "join" command to merge two files based on common fields. It starts by displaying the contents of "file1.txt" and "file2.txt" separately, then uses the "join" command to merge them into a single output where each line contains a pair of items from both files.

```
(siberkoza@faruk)~$ cat file1.txt
1·Apple
2·Banana
3·Orange
-
(siberkoza@faruk)~$ cat file2.txt
1·Red
2·Yellow
4·Green
-
(siberkoza@faruk)~$ join file1.txt file2.txt
1·Apple Red
2·Banana Yellow
-
```

```
siberkoza@faruk:[~/SiberKoza] $ ls -l
total 4
-rw-r--r-- 1 siberkoza siberkoza 146 Mar 15 03:55 fruits.txt

siberkoza@faruk:[~/SiberKoza] $ split -l 2 fruits.txt chunk-
siberkoza@faruk:[~/SiberKoza] $ ls -l
total 100
-rw-r--r-- 1 siberkoza siberkoza 13 Mar 15 03:51 chunk-aa
-rw-r--r-- 1 siberkoza siberkoza 18 Mar 15 03:51 chunk-ab
-rw-r--r-- 1 siberkoza siberkoza 17 Mar 15 03:51 chunk-ac
-rw-r--r-- 1 siberkoza siberkoza 16 Mar 15 03:51 chunk-ad
-rw-r--r-- 1 siberkoza siberkoza 11 Mar 15 03:51 chunk-ae
-rw-r--r-- 1 siberkoza siberkoza 12 Mar 15 03:51 chunk-af
-rw-r--r-- 1 siberkoza siberkoza 20 Mar 15 03:51 chunk-ag
-rw-r--r-- 1 siberkoza siberkoza 11 Mar 15 03:51 chunk-ah
-rw-r--r-- 1 siberkoza siberkoza 13 Mar 15 03:51 chunk-ai
-rw-r--r-- 1 siberkoza siberkoza 15 Mar 15 03:51 chunk-aj
-rw-r--r-- 1 siberkoza siberkoza 146 Mar 15 03:55 fruits.txt

siberkoza@faruk:[~/SiberKoza] $ cat chunk-aa
Apple
Banana

siberkoza@faruk:[~/SiberKoza] $ split -l 5 fruits.txt chf-
siberkoza@faruk:[~/SiberKoza] $ ls -l
total 116
-rw-r--r-- 1 siberkoza siberkoza 37 Mar 15 03:57 chf-aa
-rw-r--r-- 1 siberkoza siberkoza 38 Mar 15 03:57 chf-ab
-rw-r--r-- 1 siberkoza siberkoza 37 Mar 15 03:57 chf-ac
-rw-r--r-- 1 siberkoza siberkoza 34 Mar 15 03:57 chf-ad
-rw-r--r-- 1 siberkoza siberkoza 13 Mar 15 03:51 chunk-aa
-rw-r--r-- 1 siberkoza siberkoza 18 Mar 15 03:51 chunk-ab
-rw-r--r-- 1 siberkoza siberkoza 17 Mar 15 03:51 chunk-ac
-rw-r--r-- 1 siberkoza siberkoza 16 Mar 15 03:51 chunk-ad
-rw-r--r-- 1 siberkoza siberkoza 11 Mar 15 03:51 chunk-ae
-rw-r--r-- 1 siberkoza siberkoza 12 Mar 15 03:51 chunk-af
-rw-r--r-- 1 siberkoza siberkoza 20 Mar 15 03:51 chunk-ag
-rw-r--r-- 1 siberkoza siberkoza 11 Mar 15 03:51 chunk-ah
-rw-r--r-- 1 siberkoza siberkoza 13 Mar 15 03:51 chunk-ai
-rw-r--r-- 1 siberkoza siberkoza 15 Mar 15 03:51 chunk-aj
-rw-r--r-- 1 siberkoza siberkoza 146 Mar 15 03:55 fruits.txt

siberkoza@faruk:[~] $ cat chf-aa
Apple
Banana
Orange
Strawberry
Grape
```

**sort**

Command	sort
Usage Methods	sort [options] [filename]
Description	The “sort” command is used to sort the lines of a file.

```
siberkoza-linux-learning

└──(siberkoza㉿faruk)-[~]─
└─$ cat alphabetic.txt ─
banana
apple
orange
grape
cherry
─
└──(siberkoza㉿faruk)-[~]─
└─$ sort alphabetic.txt ─
apple
banana
cherry
grape
orange
```

```
siberkoza-linux-learning

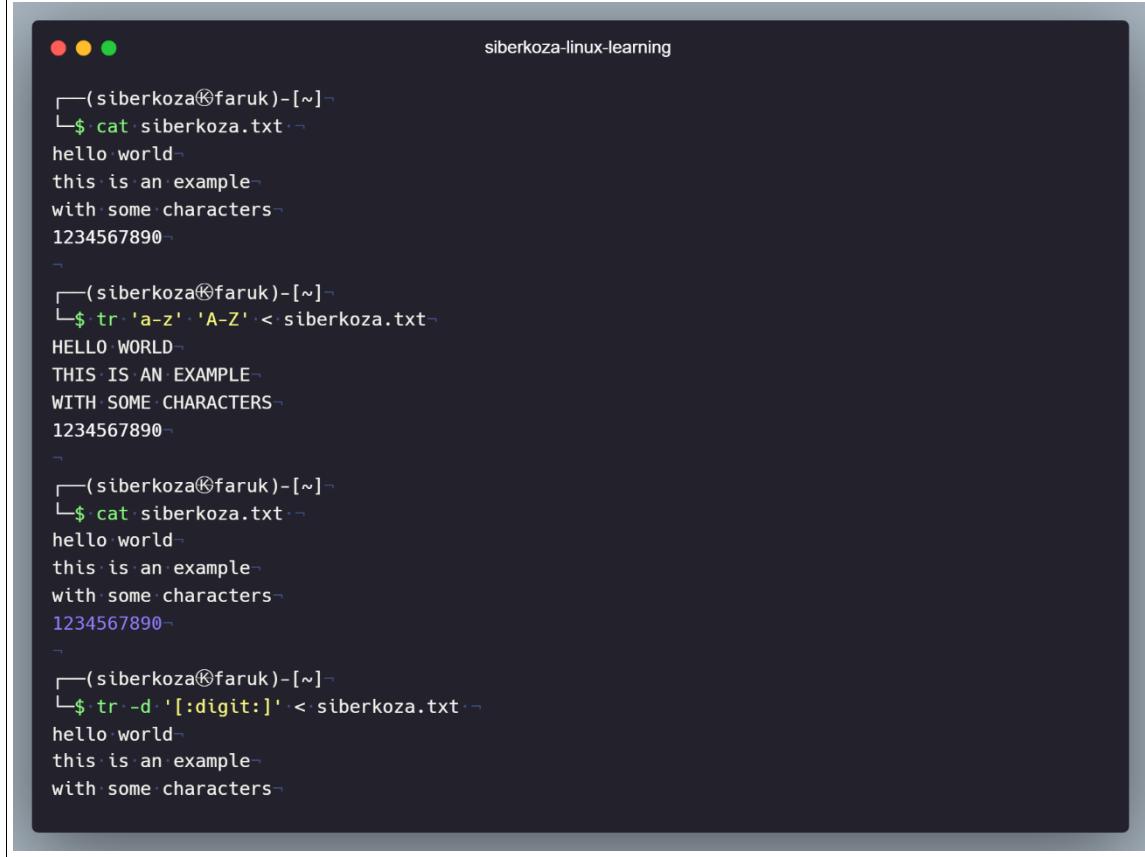
└──(siberkoza㉿faruk)-[~]─
└─$ cat numeric.txt ─
5
20
3
10
1
15
─
└──(siberkoza㉿faruk)-[~]─
└─$ sort -n numeric.txt ─
1
3
5
10
15
20
```

---

**tr**

---

Command	tr
Usage Methods	tr [option] set1 [set2]
Description	The “tr” command is used to convert or delete characters in a text stream.

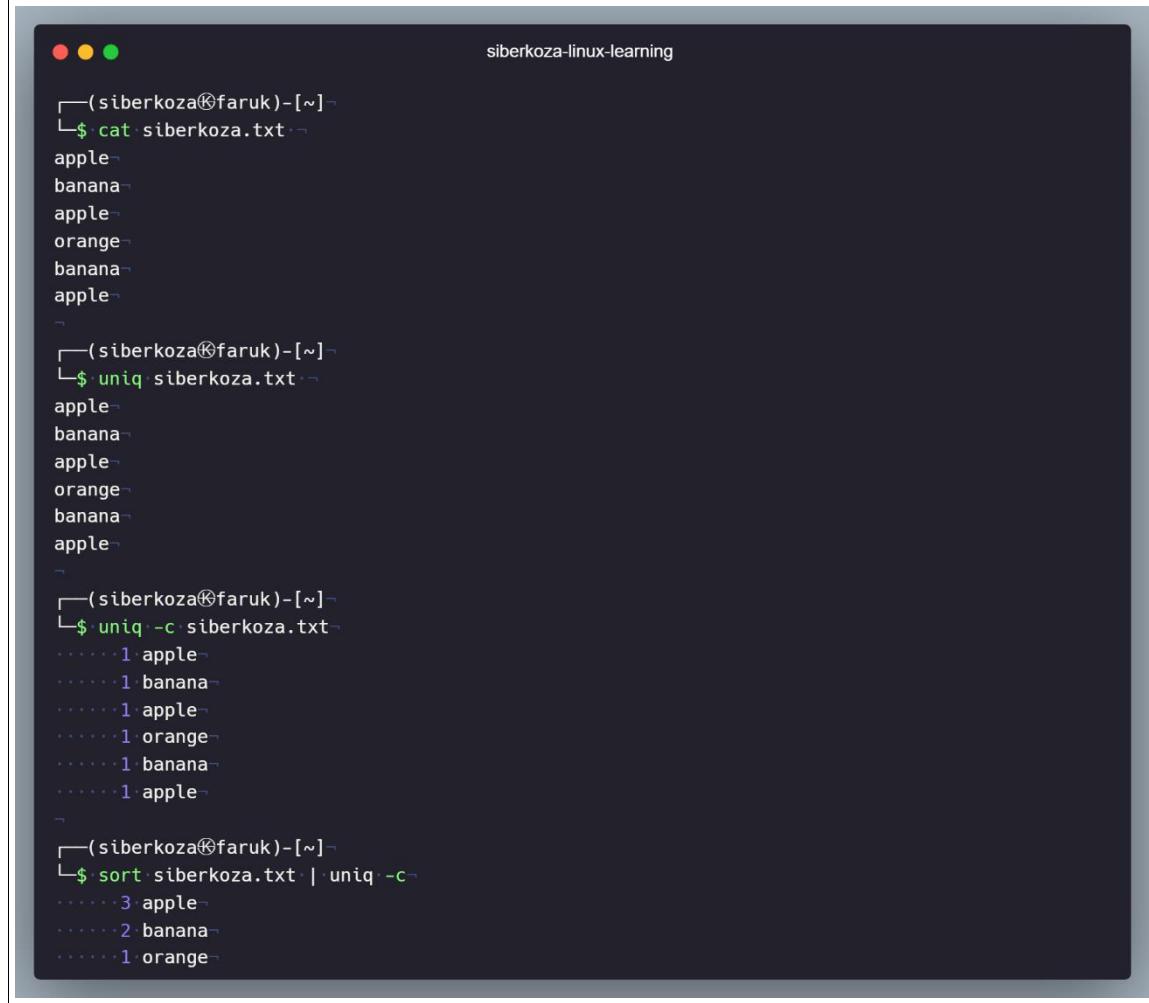


A screenshot of a terminal window titled "siberkoza-linux-learning". The terminal shows four examples of the "tr" command:

- Example 1: Converts lowercase to uppercase. Command: \$ cat siberkoza.txt | tr 'a-z' 'A-Z'. Output: HELLO WORLD
- Example 2: Converts uppercase to lowercase. Command: \$ cat siberkoza.txt | tr 'A-Z' 'a-z'. Output: hello world
- Example 3: Deletes digits from the file. Command: \$ cat siberkoza.txt | tr -d [:digit:']. Output: this is an example with some characters
- Example 4: Deletes punctuation from the file. Command: \$ cat siberkoza.txt | tr -d [:punct:']. Output: this is an example with some characters

**uniq**

Command	uniq
Usage Methods	uniq [options] [filename]
Description	The “uniq” command is used to remove consecutive repeats in a file.

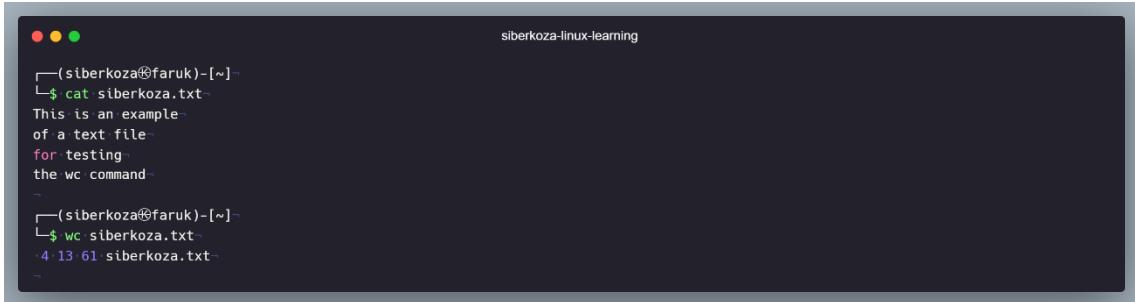


The screenshot shows a terminal window with the title "siberkoza-linux-learning". The terminal displays the following session:

```
(siberkoza@faruk)~$ cat siberkoza.txt
apple
banana
apple
orange
banana
apple
-
(siberkoza@faruk)~$ uniq siberkoza.txt
apple
banana
apple
orange
banana
apple
-
(siberkoza@faruk)~$ uniq -c siberkoza.txt
... 1 apple
... 1 banana
... 1 apple
... 1 orange
... 1 banana
... 1 apple
-
(siberkoza@faruk)~$ sort siberkoza.txt | uniq -c
... 3 apple
... 2 banana
... 1 orange
```

**wc**

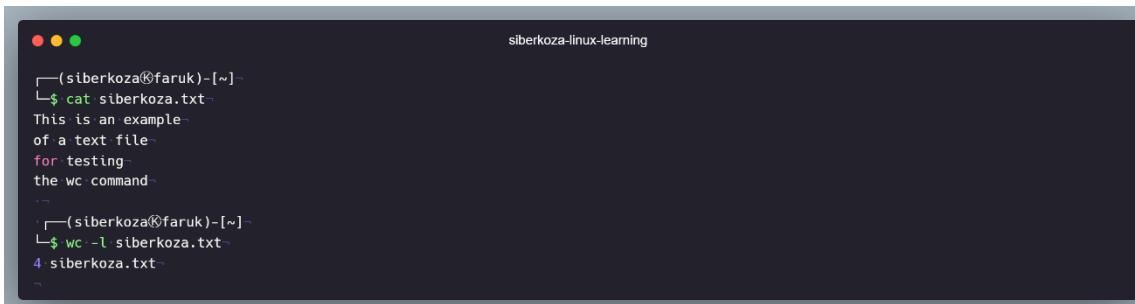
Command	wc
Usage Methods	wc [options] [filename]
Description	The “wc” command is used to count the number of lines, words and bytes of a file.



```
siberkoza@faruk:~$ cat siberkoza.txt
This is an example
of a text file
for testing
the wc command

siberkoza@faruk:~$ wc siberkoza.txt
4 13 61 siberkoza.txt
```

This output shows the number of lines, words and bytes respectively. The file has a total of 4 lines, 13 words and 61 bytes.



```
siberkoza@faruk:~$ cat siberkoza.txt
This is an example
of a text file
for testing
the wc command

siberkoza@faruk:~$ wc -l siberkoza.txt
4 siberkoza.txt
```

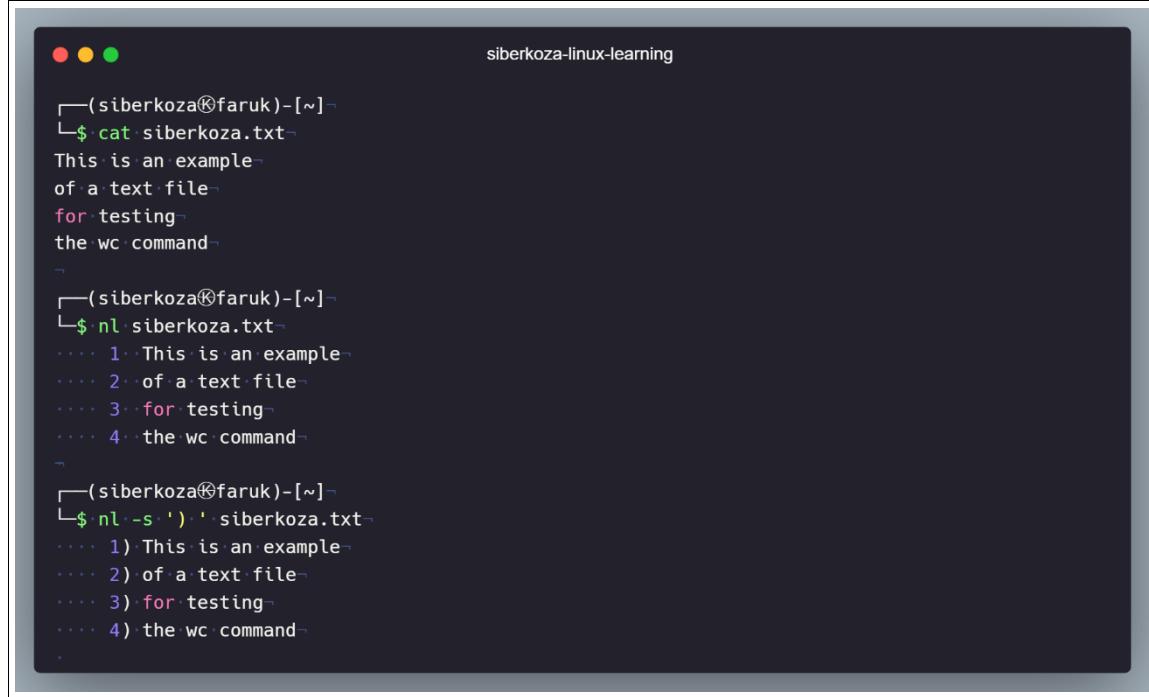
This command displays the total number of lines (4) in the file. The -l option is used to get only the number of lines. The other options are for querying different properties such as -w (word count) and -c (byte count).

---

**nl**

---

Command	
Usage Methods	<code>nl [options] [filename]</code>
Description	The “nl” command is used to number lines in a file.



A screenshot of a Linux terminal window titled "siberkoza-linux-learning". The terminal shows three examples of the "nl" command. In the first example, the user runs "cat siberkoza.txt" which displays a text file containing four lines: "This is an example", "of a text file", "for testing", and "the wc command". In the second example, the user runs "nl siberkoza.txt" which adds line numbers (1, 2, 3, 4) before each line of the same text. In the third example, the user runs "nl -s ')' siberkoza.txt" which adds line numbers (1, 2, 3, 4) and encloses each line in parentheses. The terminal window has a dark background and light-colored text.

---

**grep**

---

Command	grep
Usage Methods	grep [options] pattern [files]
Description	“grep” searches for lines in a file or standard input stream that match a specific pattern and prints these lines to the screen. The pattern can be defined using regular text or regular expressions (regex).



The screenshot shows a terminal window with the title "siberkoza-linux-learning". The terminal history is as follows:

```
(siberkoza@faruk)~$ cat unix.txt
UNIX is a family of multitasking, multiuser computer operating systems that derive from the original AT&T Unix, developed in the 1970s at the Bell Labs research center by Ken Thompson, Dennis Ritchie, and others. Unix systems are characterized by a modular design that is sometimes called the "Unix philosophy", meaning that the operating system provides a set of simple tools that each performs a limited, well-defined function, with a unified filesystem as the main means of communication and a shell scripting and command language to combine the tools to perform complex workflows.

(siberkoza@faruk)~$ grep "Unix" unix.txt
UNIX is a family of multitasking, multiuser computer operating systems that derive from the original AT&T Unix, developed in the 1970s at the Bell Labs research center by Ken Thompson, Dennis Ritchie, and others. Unix systems are characterized by a modular design that is sometimes called the "Unix philosophy", meaning that the operating system provides a set of simple tools that each performs a limited, well-defined function, with a unified filesystem as the main means of communication and a shell scripting and command language to combine the tools to perform complex workflows.

(siberkoza@faruk)~$ grep -o "Unix" unix.txt
Unix
Unix
Unix
```

**vim**

<b>Command</b>	vim
<b>Usage Methods</b>	vim [options] [filename]
<b>Description</b>	“vim” is a text editor that is highly configurable and powerful. It allows users to create, edit, and view text files.

**nano**

<b>Command</b>	nano
<b>Usage Methods</b>	nano [options] [filename]
<b>Description</b>	“nano” is a simple and easy-to-use text editor for Unix-like operating systems. “nano” allows users to create, edit, and view text files directly from the command line.

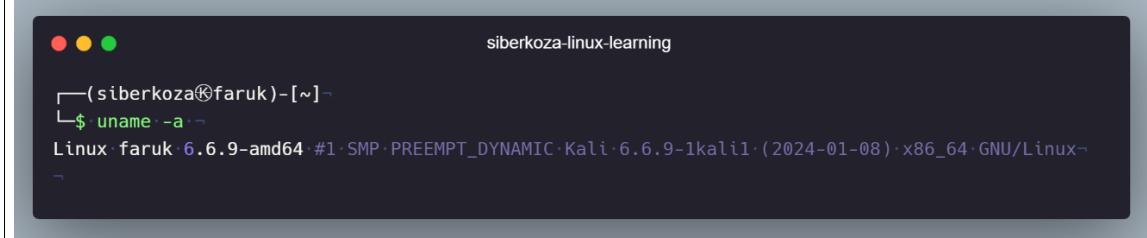
**Linux Commands for Getting Help: --help – man – alias**

- **command --help:** Used to get a short help text on how to use a command.
- **man command:** Used to display a comprehensive manual page for a command.
- **alias:** Used to create and use shortcuts for frequently used commands.

**System Information Commands:** uname - hostname - lscpu - lsb\_release  
- uptime - free - df - du - ifconfig/ip - netstat

### uname

Command	uname -a
Description	“Uname” displays operating system information and information about the system hardware. This command is often used to get general information about the system.

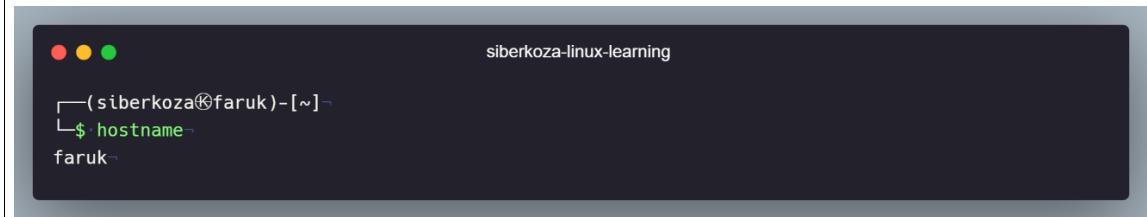


The terminal window shows the output of the 'uname -a' command. The output includes the kernel version (6.6.9), architecture (amd64), SMP, PREEMPT\_DYNAMIC, and the date (2024-01-08). The window title is 'siberkoza-linux-learning'.

```
siberkoza@faruk:[~] $ uname -a
Linux faruk 6.6.9-060609-generic #1 SMP PREEMPT_DYNAMIC Kali-6.6.9-1kali1 (2024-01-08) x86_64 GNU/Linux
```

### hostname

Command	hostname
Description	The “hostname” command displays or sets the host name of the system.



The terminal window shows the output of the 'hostname' command. The output shows the current host name is 'faruk'. The window title is 'siberkoza-linux-learning'.

```
siberkoza@faruk:[~] $ hostname
faruk
```

**lscpu**

<b>Command</b>	lscpu
<b>Description</b>	The “lscpu” command provides information about the CPU (Central Processing Unit) in the system. This command displays the CPU's specifications, architecture, speed, number of cores, number of threads, and other relevant information.

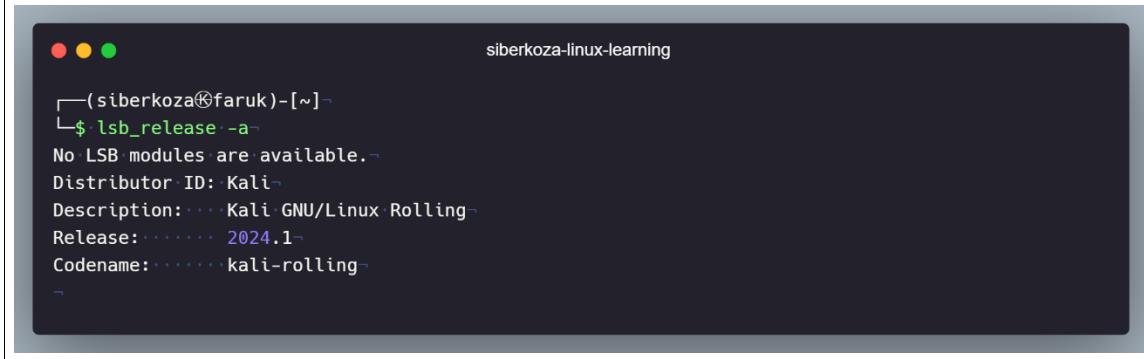
```

siberkoza@faruk:[~] ~
└─$ lscpu
Architecture:          x86_64
CPU op-mode(s):        32-bit, 64-bit
Address sizes:         40 bits physical, 48 bits virtual
Byte Order:            Little Endian
CPU(s):                4
On-line CPU(s) list:  0-3
Vendor ID:             GenuineIntel
Model name:            Intel(R) Core(TM) i5-7300HQ CPU @ 2.50GHz
CPU family:             6
Model:                 158
Thread(s) per core:    1
Core(s) per socket:    2
Socket(s):              2
Stepping:               9
BogoMIPS:              4992.00
Flags:                 fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 ss ht syscall nx rdtscp lm consta nt_tsc arch_perfmon nopl tsc_reliable nonstop_tsc cpuid tsc_known_freq pni pclmulqdq ssse3 fma cx16 sse4_1 sse4_2 movbe popcnt aes xsave avx hypervisor lahf_lm 3dnowprefetch pt1 arat
Virtualization features: ~
Hypervisor vendor:      VMware
Virtualization type:    full
Caches (sum of all):   ~
L1d:                   128 KiB (4 instances)
L1i:                   128 KiB (4 instances)
L2:                     1 MiB (4 instances)
L3:                     12 MiB (2 instances)
NUMA:                  ~
NUMA node(s):           1
NUMA node0 CPU(s):     0-3
Vulnerabilities:       ~
Gather data sampling:  Unknown: Dependent on hypervisor status
Itlb multihit:          KVM: Mitigation: VMX unsupported
Litf:                   Mitigation: PTE Inversion
Mds:                     Vulnerable: Clear CPU buffers attempted, no microcode; SMT Host state unknown
Meltdown:                Mitigation: PTI
Mmio stale data:        Vulnerable: Clear CPU buffers attempted, no microcode; SMT Host state unknown
Retbleed:                Vulnerable
Spec rstack overflow:  Not affected
Spec store bypass:      Vulnerable
Spectre v1:              Mitigation: usercopy/swaps barriers and __user pointer sanitization
Spectre v2:              Mitigation: Retpolines, STIBP disabled, RSB filling, PBRSB-eIBRS Not affected
Srbds:                  Not affected
Tsx async abort:        Not affected

```

## lsb\_release

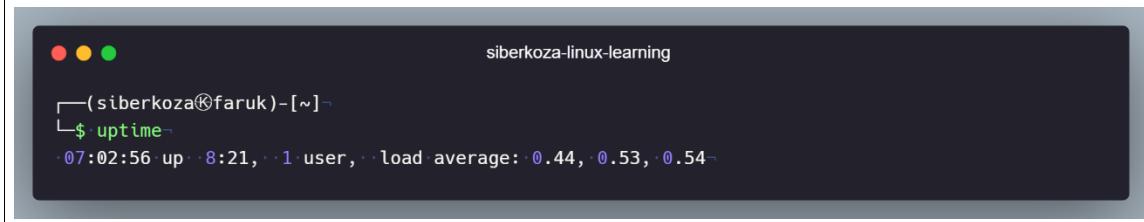
Command	lsb_release -a
Description	This command provides the version number, distribution name, release name, and other relevant information for the Linux distribution.



```
(siberkoza㉿faruk) ~
$ lsb_release -a
No LSB modules are available.
Distributor ID: Kali
Description: Kali GNU/Linux Rolling
Release: 2024.1
Codename: kali-rolling
```

## uptime

Command	uptime
Description	The "uptime" command is a command that displays the uptime and system load on a Linux system. System load indicates how intensive the CPU and other resources are, and usually refers to the average load over a given period of time.



```
(siberkoza㉿faruk) ~
$ uptime
07:02:56 up 1 day 3 hours 45 minutes, 1 user, load average: 0.44, 0.53, 0.54
```

In this output, the uptime in the system is indicated as 1 day 3 hours 45 minutes. The total number of users on the system (user) and the average load (load average) are also displayed. The average load refers to the average load of the CPU in the last 1, 5 and 15 minutes. In the example above, the average load is 0.32, 0.27 and 0.25 respectively. Low values indicate that the load on the system is light, while high values indicate a heavy workload on the system.

**free**

Command	free -m / free -g
Description	The “free” command provides information about system memory (RAM) usage. This command shows the total, used, idle, and cached memory on the system.

```
siberkoza@faruk:[~]
$ free
              total        used        free      shared  buff/cache   available
Mem:       4010124     1578504     1094004      26656     1653560     2431620
Swap:      1048572          0     1048572

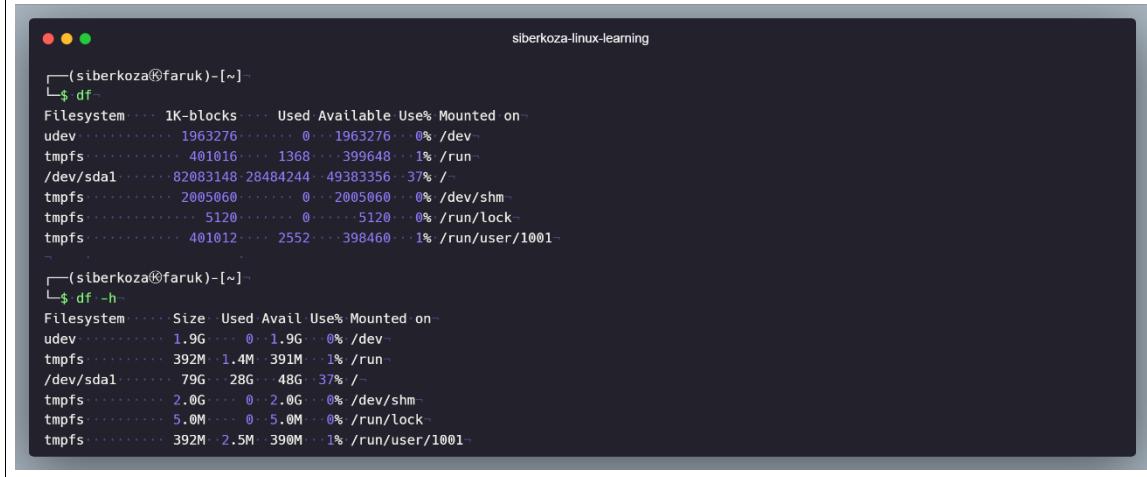
(siberkoza@faruk)[~]
$ free -m
              total        used        free      shared  buff/cache   available
Mem:        3916       1532       1077        26       1614       2383
Swap:      1023          0       1023

(siberkoza@faruk)[~]
$ free -g
              total        used        free      shared  buff/cache   available
Mem:          3          1          1          0          1          2
Swap:          0          0          0
```

The -m option displays results in megabytes (MB), while the -g option displays results in gigabytes (GB).

**df**

<b>Command</b>	df
<b>Description</b>	The “df” command displays the name of the available file systems, the device to which the file systems are mounted, the total space of the file systems, the used space, the free space and the percentage of usage.

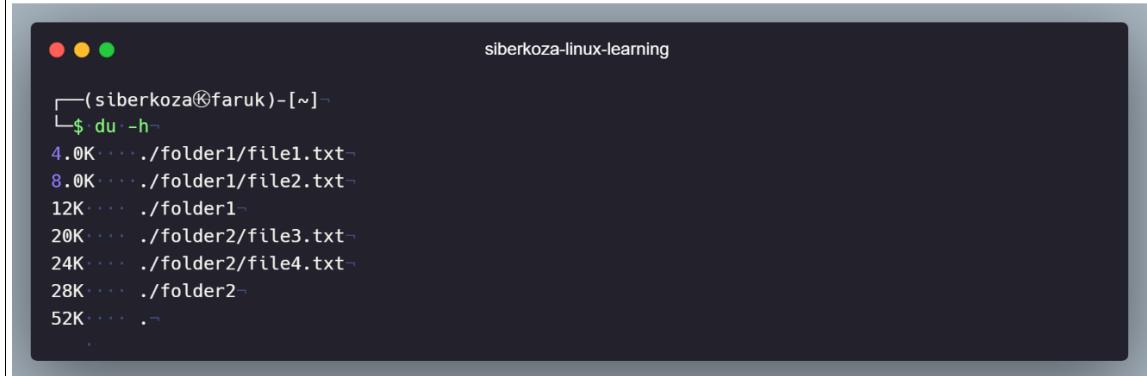


```
(siberkoza@faruk) [~]
$ df
Filesystem 1K-blocks Used Available Use% Mounted on
udev 1963276 0 1963276 0% /dev
tmpfs 401016 1368 399648 1% /run
/dev/sda1 82083148 28484244 49383356 37% /
tmpfs 2005060 0 2005060 0% /dev/shm
tmpfs 5120 0 5120 0% /run/lock
tmpfs 401012 2552 398460 1% /run/user/1001

(siberkoza@faruk) [~]
$ df -h
Filesystem Size Used Avail Use% Mounted on
udev 1.9G 0 1.9G 0% /dev
tmpfs 392M 1.4M 391M 1% /run
/dev/sda1 796M 28G 48G 37%
tmpfs 2.0G 0 2.0G 0% /dev/shm
tmpfs 5.0M 0 5.0M 0% /run/lock
tmpfs 392M 2.5M 390M 1% /run/user/1001
```

**du**

<b>Command</b>	du -h
<b>Description</b>	The du command is a command that shows disc usage. It is short for "Disk Usage". This command shows the disk space occupied by a particular directory or file and the total disk usage of subdirectories.



```
(siberkoza@faruk) [~]
$ du -h
4.0K ./folder1/file1.txt
8.0K ./folder1/file2.txt
12K ./folder1
20K ./folder2/file3.txt
24K ./folder2/file4.txt
28K ./folder2
52K .
```

**ifconfig / ip**

Command	<code>ifconfig / ip addr show</code>
Description	The “ifconfig” and “ip” commands are used to configure network interfaces and manage network connections. Both commands display the IP addresses, operational status, MAC addresses, and other details of network interfaces.

```
(siberkoza@faruk) [~] $ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
      inet 192.168.1.100 brd 192.168.1.255 netmask 255.255.255.0
            broadcast 192.168.1.255
            ether 08:00:27:8e:dd:e4 txqueuelen 1000 (Ethernet)
            RX packets 4513 bytes 342685 (342.6 KB)
            RX errors 0 dropped 0 overruns 0 frame 0
            TX packets 3018 bytes 322400 (322.4 KB)
            TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

```
(siberkoza@faruk) [~] $ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
   inet: 127.0.0.1/8 brd 127.0.0.1 scope host lo
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
   inet: 192.168.1.100/24 brd 192.168.1.255 scope global dynamic eth0
        valid_lft 86240sec preferred_lft 86240sec
        ether 08:00:27:8e:dd:e4 txqueuelen 1000 (Ethernet)
        valid_lft forever preferred_lft forever
```

## netstat

Command	<code>netstat -tuln</code>
Description	The “netstat” command is used to monitor and manage network connections and display information about the network.

```
(siberkoza@faruk) [~] $ netstat -tuln
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp       0      0 127.0.0.1:46793          0.0.0.0:*
```