

# Lecture \_7

LINUX essentials  
Dr .Sara Mohamed



# Filtering Input

# grep command in Unix/Linux

- The grep command is one of the most useful tools in Linux and Unix systems. It is used to search for specific words, phrases, or patterns inside text files, and shows the matching lines on your screen. **grep** Command is useful when you need to quickly find certain keywords or phrases in logs or documents.
- The grep command is a text filter that will search input and return lines which contain a match to a given pattern.

# Syntax of grep Command in Unix/Linux

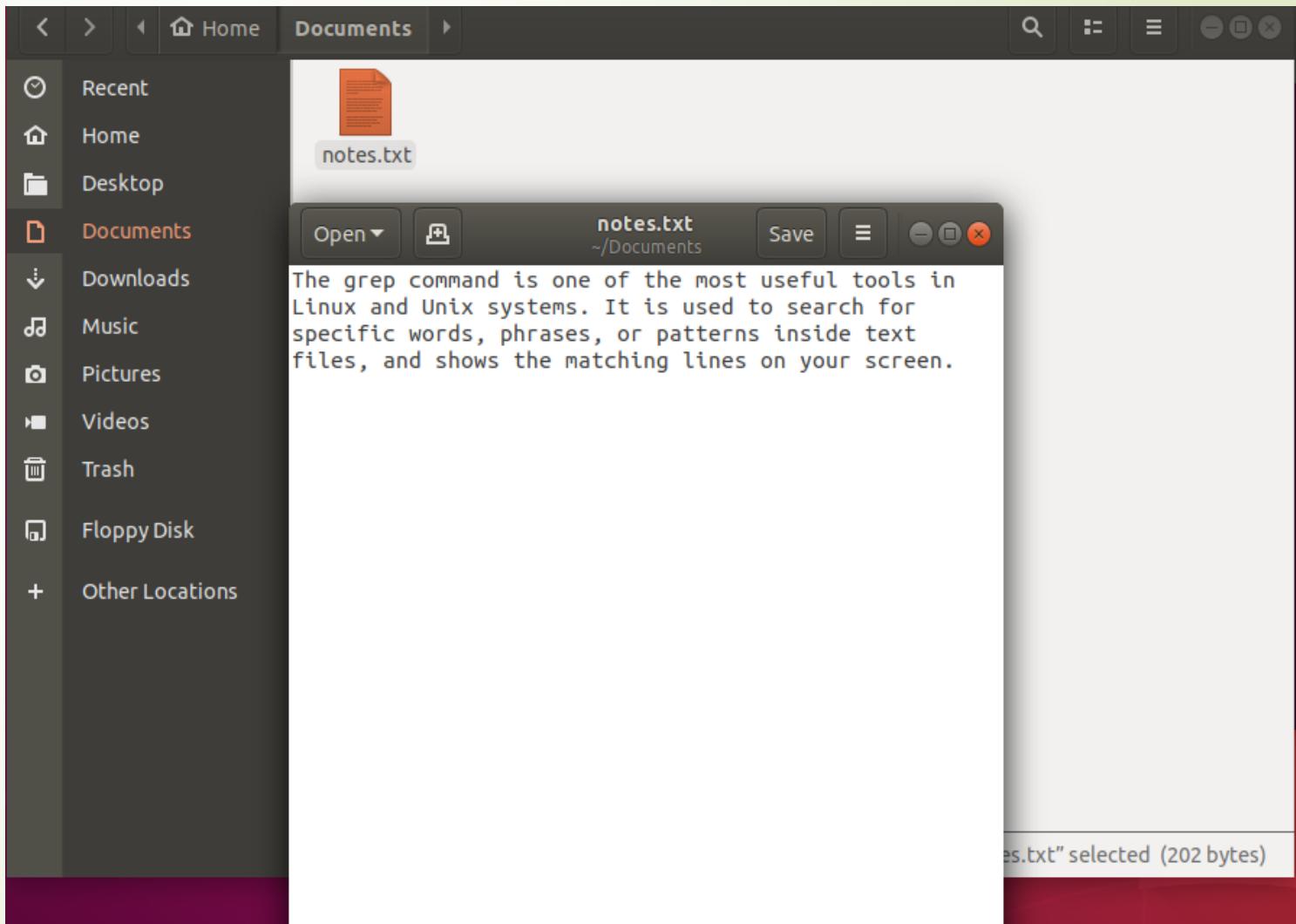
- The basic syntax of the `grep` command is as follows:

```
grep [options] pattern [files]
```

- **options:** These are command-line flags that modify the behaviour of grep.
- **pattern:** This is the regular expression you want to search for.
- **File:** This is the name of the file(s) you want to search within. You can specify multiple files for simultaneous searching.

# grep Command example\_1

→ Suppose we have A file called notes.txt as shown:





you want to find all lines containing the word **command**

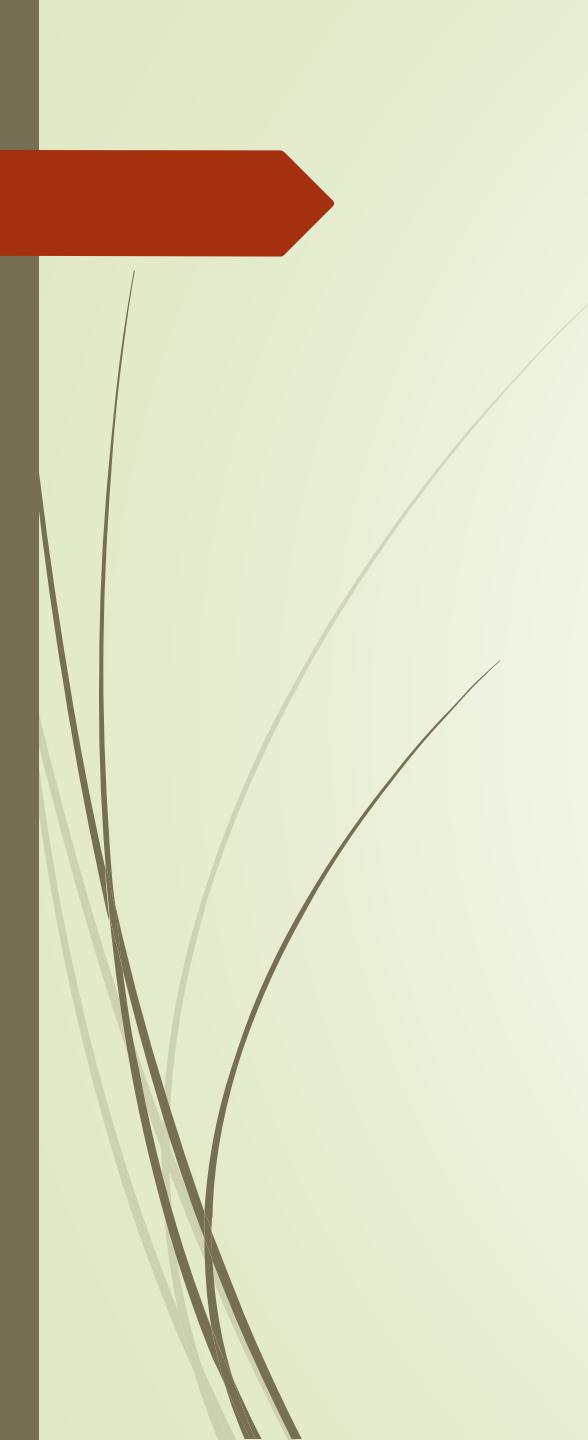
```
ahmed@ubuntu:~/Documents$ cat notes.txt
```

The grep command is one of the most useful tools in Linux and Unix systems. It is used to search for specific words, or patterns inside text files, and shows the matching lines on your screen.

```
ahmed@ubuntu:~/Documents$ grep command notes.txt
```

The grep **command** is one of the most useful tools in Linux and Unix systems. It is used to search for specific words, or patterns inside text files, and shows the matching lines on your screen.

```
ahmed@ubuntu:~/Documents$ |
```



# **Commonly Used grep Options**

# Case insensitive search

- The `-i` option enables to search for a string case insensitively in the given file. It matches the words like "UNIX", "Unix", "unix".

```
ahmed@ubuntu:~/Documents$ grep -i grep notes.txt
```

The **grep** command is one of the most useful tools in Linux and Unix systems. It is used to search inside text files, and shows the matching lines on your screen.

The **Grep** command is one of the most useful tools in Linux and Unix systems.

The **GREP** command is one of the most useful tools in Linux and Unix systems.

```
ahmed@ubuntu:~/Documents$
```

# Displaying the Count Matches Using grep

- We can find the number of lines that matches the given string/pattern.
- 

```
ahmed@ubuntu:~/Documents$ grep -c grep notes.txt  
1  
ahmed@ubuntu:~/Documents$ █
```

# Checking Whole Words Using grep

By default, grep matches the given string/pattern even if it is found as a substring in a file. The `-w` option to grep makes it match only the whole words. `grep -w` searches for a whole word only, not part of another word.  
It matches only when the pattern is a complete word with boundaries (space, punctuation, line start/end).

```
administrator@GFG19566-LAPTOP:~$ grep -w "unix" geekfile.txt
unix is great os. unix was developed in Bell labs.
uNix is easy to learn.unix is a multiuser os.Learn unix .unix is a powerful.
administrator@GFG19566-LAPTOP:~$ █
```

# Display Matched Pattern Using grep

► By default, grep displays the entire line which has the matched string. We can make the grep to display only the matched string by using the -o option.

The screenshot shows a terminal window and a text editor window side-by-side. The terminal window at the top has a dark background and contains the following text:

```
File Edit View Search Terminal Help  
ahmed@ubuntu:~/Documents$ grep -o files notes.txt  
files  
files  
files  
ahmed@ubuntu:~/Documents$
```

To the left of the terminal is a vertical dock with several icons: a folder, a document with an orange 'A', a question mark, a right arrow, a blue sunburst, a circular icon with an orange 'A', and a notebook.

The text editor window on the right has a light gray background. At the top, it shows the file path: notes.txt ~/Documents. The main area of the editor contains the following text:

The grep command is one of the most useful tools in Linux and Unix systems. It is used to search for specific words, phrases, or patterns inside text files, and shows the matching lines on your screen.

The Grep command is one of the most useful tools in Linux and Unix systems.

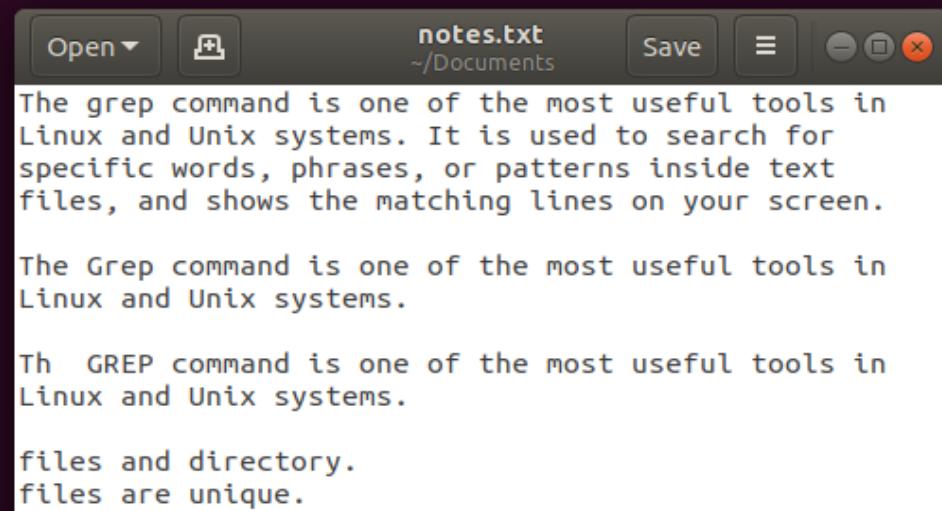
Th GREP command is one of the most useful tools in Linux and Unix systems.

files and directory.  
files are unique.|

At the bottom of the editor window, there are status indicators: Plain Text ▾ Tab Width: 8 ▾ Ln 8, Col 18 ▾ INS.

# Show Line Numbers with grep -n

```
files
files
files
ahmed@ubuntu:~/Documents$ grep -n files notes.txt
1:The grep command is one of the most useful tools in Linux and Unix systems. It is used to search for specific words, phrase
s, or patterns inside text files, and shows the matching lines on your screen.
7:files and directory.
8:files are unique.
ahmed@ubuntu:~/Documents$
```



# Inverting the Pattern Match Using grep

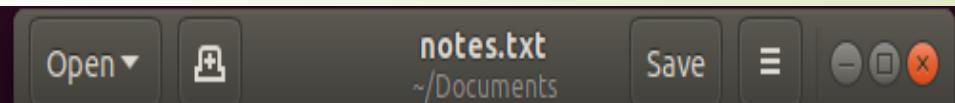
- ▶ You can display the lines that are not matched with the specified search string pattern using the `-v` option.

```
ahmed@ubuntu:~/Documents$ grep -v files notes.txt
```

The Grep command is one of the most useful tools in Linux and Unix systems.

The GREP command is one of the most useful tools in Linux and Unix systems.

```
ahmed@ubuntu:~/Documents$ 
```



The grep command is one of the most useful tools in Linux and Unix systems. It is used to search for specific words, phrases, or patterns inside text files, and shows the matching lines on your screen.

The Grep command is one of the most useful tools in Linux and Unix systems.

The GREP command is one of the most useful tools in Linux and Unix systems.

files and directory.  
files are unique.|

# Regular expressions

- Regular expressions are patterns that only certain commands are able to interpret. Regular expressions can be expanded to match certain sequences of characters in text.
- Regular expressions have two common forms: **basic and extended**. Most commands that use regular expressions can interpret basic regular expressions. However, extended regular expressions are not available for all commands and a command option is typically required for them to work correctly

## Basic Regex Character(s) Meaning

[ ]

Any one specified character

[ ^ ]

Not the one specified character

\*

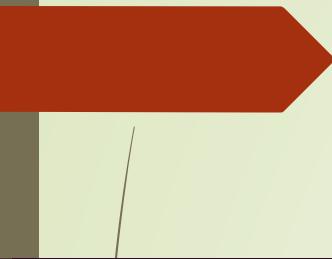
Zero or more of the previous character

^

If first character in the pattern, then pattern must be at beginning of the line to match, otherwise just a literal ^

\$

If last character in the pattern, then pattern must be at the end of the line to match, otherwise just a literal \$



```
ahmed@ubuntu:~/Documents$ grep [i] notes.txt
```

The grep command **is** one of the most useful tools **in** Linux and Unix systems. It **is** used to search for **specific** words, phrases or patterns **inside** text **files**, and shows the matching **lines** on your screen.

The Grep command **is** one of the most useful tools **in** Linux and Unix systems.

The GREP command **is** one of the most useful tools **in** Linux and Unix systems.  
**files** and **directory**.

**files** are **unique**.

```
ahmed@ubuntu:~/Documents$
```



The first anchor character ^ is used to ensure that a pattern appears at the beginning of the line.

```
ahmed@ubuntu:~/Documents$ grep ^A notes.txt  
A is for Apple sara  
ahmed@ubuntu:~/Documents$ grep ^T notes.txt  
This new idea  
ahmed@ubuntu:~/Documents$
```

A is for Apple sara  
B is for Bear  
C is for Cat  
E is for Elephant  
G is for Grapes  
H is for Happy  
K is for Kangaroo  
O is for Oval  
P is for Pickle sara  
Q is for Quark

This new idea|

The second anchor character \$ can be used to ensure a pattern appears at the end of the line.

```
ahmed@ubuntu:~/Documents$ cat red.txt
```

banana

cola

lemon

tea

mangoa

apple

```
ahmed@ubuntu:~/Documents$ grep a$ red.txt
```

banana

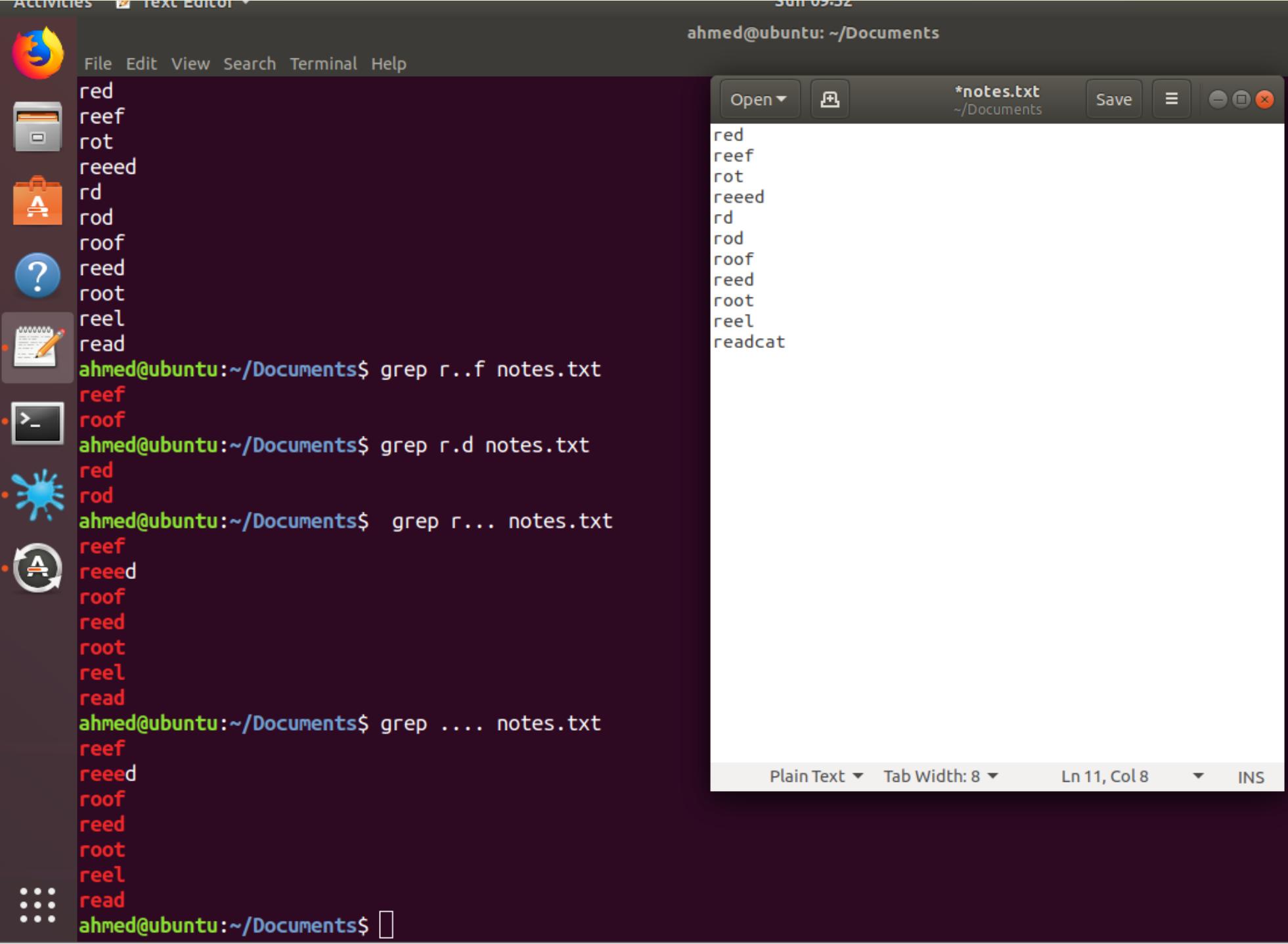
cola

tea

mangoa

```
ahmed@ubuntu:~/Documents$ █
```

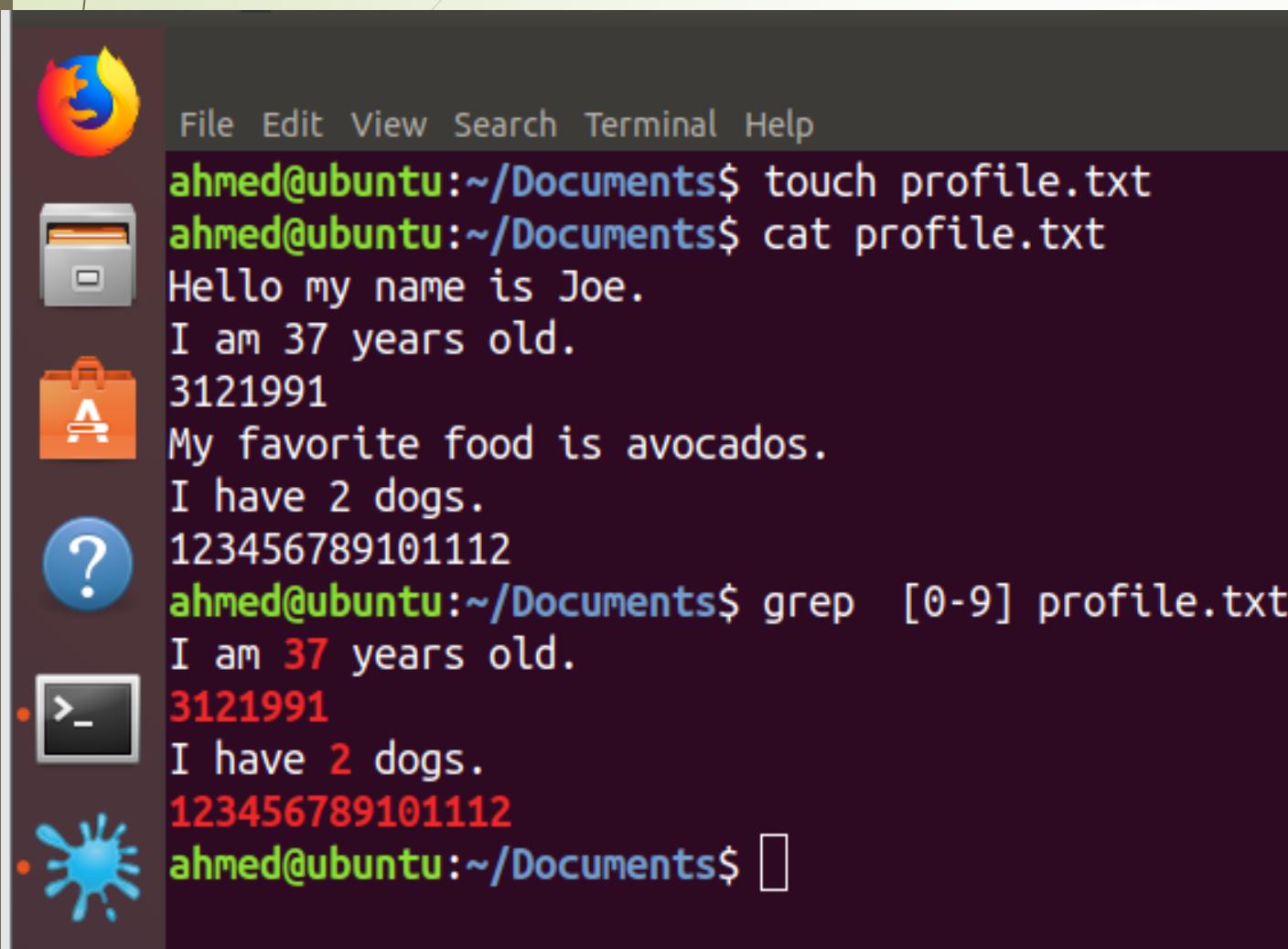
# Match a Single Character With .



The image shows a screenshot of an Ubuntu desktop environment. On the left, there is a dock with several icons: a browser, a file manager, a terminal, a help icon, a notes icon, a terminal icon, a splash screen icon, and a system settings icon. The terminal window at the bottom left shows the command line interface with the following text:  
File Edit View Search Terminal Help  
red  
reef  
rot  
reed  
rd  
rod  
roof  
reed  
root  
reel  
read  
ahmed@ubuntu:~/Documents\$ grep r..f notes.txt  
reef  
roof  
ahmed@ubuntu:~/Documents\$ grep r.d notes.txt  
red  
rod  
ahmed@ubuntu:~/Documents\$ grep r... notes.txt  
reef  
reed  
roof  
reed  
root  
reel  
read  
ahmed@ubuntu:~/Documents\$ grep .... notes.txt  
reef  
reed  
roof  
reed  
root  
reel  
read  
ahmed@ubuntu:~/Documents\$

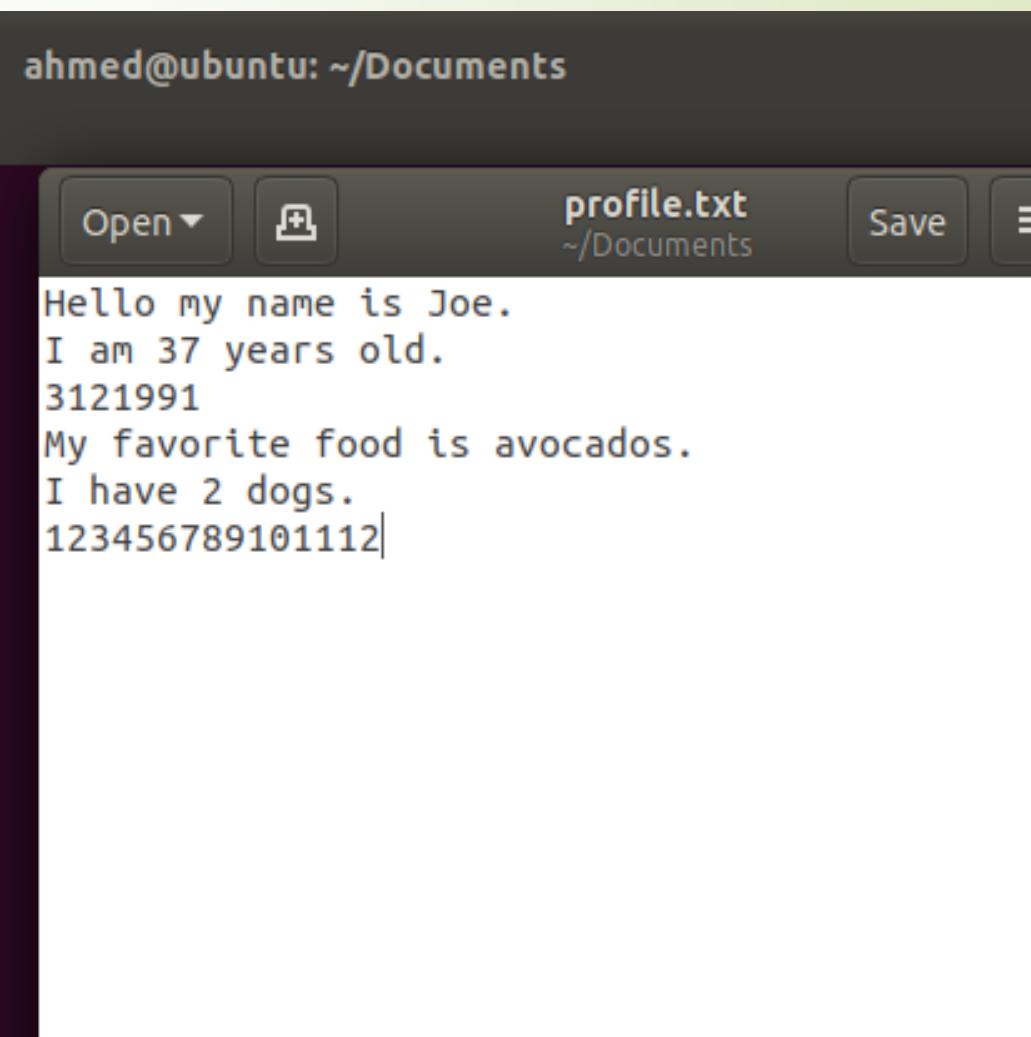
The text editor window on the right displays the contents of the 'notes.txt' file, which contains the same list of words as the terminal output. The status bar at the bottom of the text editor shows 'Plain Text', 'Tab Width: 8', 'Ln 11, Col 8', and 'INS'.

To find all the lines in the profile.txt which have a number in them, use the pattern [0123456789] or [0-9]:



A screenshot of a Linux desktop environment, likely Ubuntu, showing a terminal window. The terminal window has a dark background and contains the following text:

```
ahmed@ubuntu:~/Documents$ touch profile.txt
ahmed@ubuntu:~/Documents$ cat profile.txt
Hello my name is Joe.
I am 37 years old.
3121991
My favorite food is avocados.
I have 2 dogs.
123456789101112
ahmed@ubuntu:~/Documents$ grep [0-9] profile.txt
I am 37 years old.
3121991
I have 2 dogs.
123456789101112
ahmed@ubuntu:~/Documents$
```



A screenshot of a Linux desktop environment, likely Ubuntu, showing a text editor window. The window title is "profile.txt" and it shows the following text:

```
ahmed@ubuntu: ~/Documents
Open ▾
profile.txt
~/Documents
Save
Hello my name is Joe.
I am 37 years old.
3121991
My favorite food is avocados.
I have 2 dogs.
123456789101112
```

On the other hand, to find all the lines which contain any non-numeric characters, insert a ^ as the first character inside the brackets.



```
123456789101112
ahmed@ubuntu:~/Documents$ grep [^0-9] profile.txt
Hello my name is Joe.
I am 37 years old.
My favorite food is avocados.
I have 2 dogs.
ahmed@ubuntu:~/Documents$
```

The . normally matches any one character, but placed inside the square brackets, then it will just match itself.

```
ahmed@ubuntu:~/Documents$ cat profile.txt
```

Hello my name is Joe.

I am 37 years old.

3121991

My favorite food is avocados.

I have 2 dogs.

123456789101112

```
ahmed@ubuntu:~/Documents$ grep [a] profile.txt
```

Hello my name is Joe.

I am 37 years old.

My favorite food is avocados.

I have 2 dogs.

```
ahmed@ubuntu:~/Documents$ grep [.] profile.txt
```

Hello my name is Joe.

I am 37 years old.

My favorite food is avocados.

I have 2 dogs.

profile.txt  
~/Documents

Hello my name is Joe.

I am 37 years old.

3121991

My favorite food is avocados.

I have 2 dogs.

123456789101112

Plain Text ▾ Tab Width: 8 ▾

Ln 6, Col 16 ▾

INS



File Edit View Search Terminal Help

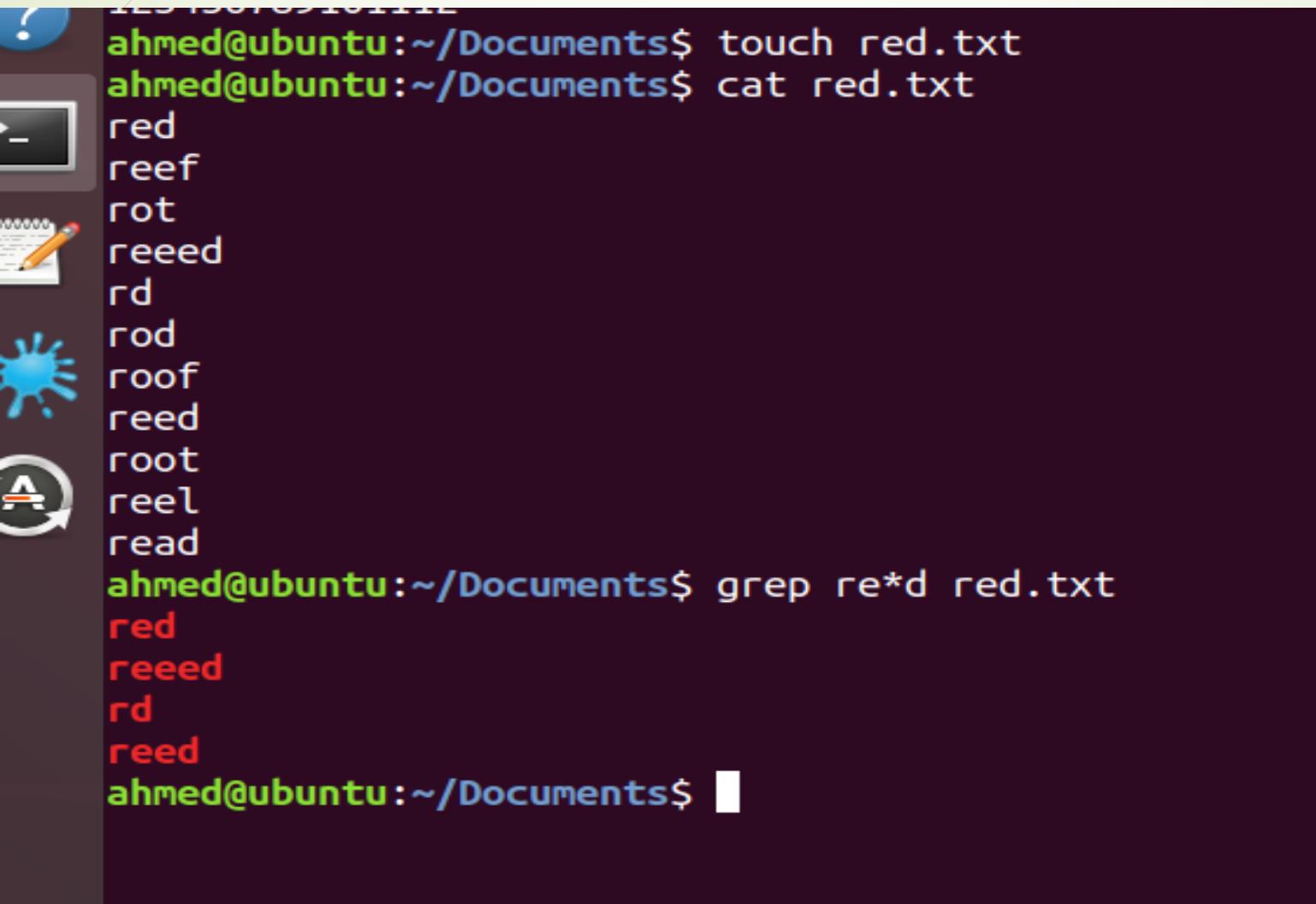
ahmed@ubuntu:~/Documents\$ grep [3] profile.txt  
I am 37 years old.

3121991

123456789101112

ahmed@ubuntu:~/Documents\$

Match a Repeated Character Or Patterns With \* .The regular expression character \* is used to match **zero or more occurrences** of a character or pattern preceding it

A screenshot of a Linux desktop environment, specifically Ubuntu, showing a terminal window. The terminal window has a dark background and light-colored text. It displays a command-line session where the user creates a file named 'red.txt' and then lists its contents. Finally, the user runs a grep command to search for the pattern 're\*d' in the file.

```
ahmed@ubuntu:~/Documents$ touch red.txt
ahmed@ubuntu:~/Documents$ cat red.txt
red
reef
rot
reeed
rd
rod
roof
reed
root
reel
read
ahmed@ubuntu:~/Documents$ grep re*d red.txt
red
reeed
rd
reed
ahmed@ubuntu:~/Documents$
```

Activities Text Editor ▾ Sun 10:28

ahmed@ubuntu: ~/Documents

File Edit View Search Terminal Help

```
ahmed@ubuntu:~/Documents$ grep 'e*' red.txt
red
reef
rot
reed
rd
rod
roof
reed
root
reel
read
ahmed@ubuntu:~/Documents$
```

Open ▾ red.txt ~/Documents Save ⌂

red  
reef  
rot  
reed  
rd  
rod  
roof  
reed  
root  
reel  
read|

# Displaying Directory Structure in a Tree Format in Linux

- The tree command in Linux displays the directory structure in a hierarchical, tree-like format, providing a clear visual representation of files and subdirectories.
- Tree command helps visualize the organization of the filesystem.
- It can include hidden files and limit the depth of display.
- Useful for quickly analysing complex directory structures.



File Edit View Search Terminal Help

```
ahmed@ubuntu:~/Documents$ cd  
ahmed@ubuntu:~$ tree
```

Command 'tree' not found, but can be installed with:

```
sudo snap install tree  
sudo apt install tree
```

## Installing 'tree' Command in Linux

See 'snap info tree' for additional versions.



```
ahmed@ubuntu:~$ sudo apt install tree  
[sudo] password for ahmed:  
Reading package lists... Done  
Building dependency tree  
Reading state information... Done  
The following NEW packages will be installed:  
  tree
```

0 upgraded, 1 newly installed, 0 to remove and 732 not upgraded.

Need to get 40.7 kB of archives.

After this operation, 105 kB of additional disk space will be used.

```
Get:1 http://us.archive.ubuntu.com/ubuntu bionic/universe amd64 tree amd64 1.7.0-5 [40.7 kB]
```

Fetched 40.7 kB in 1s (50.1 kB/s)

Selecting previously unselected package tree.

(Reading database ... 112964 files and directories currently installed.)

Preparing to unpack .../tree\_1.7.0-5\_amd64.deb ...

Unpacking tree (1.7.0-5) ...

Setting up tree (1.7.0-5) ...

Processing triggers for man-db (2.8.3-2) ...

```
ahmed@ubuntu:~$
```

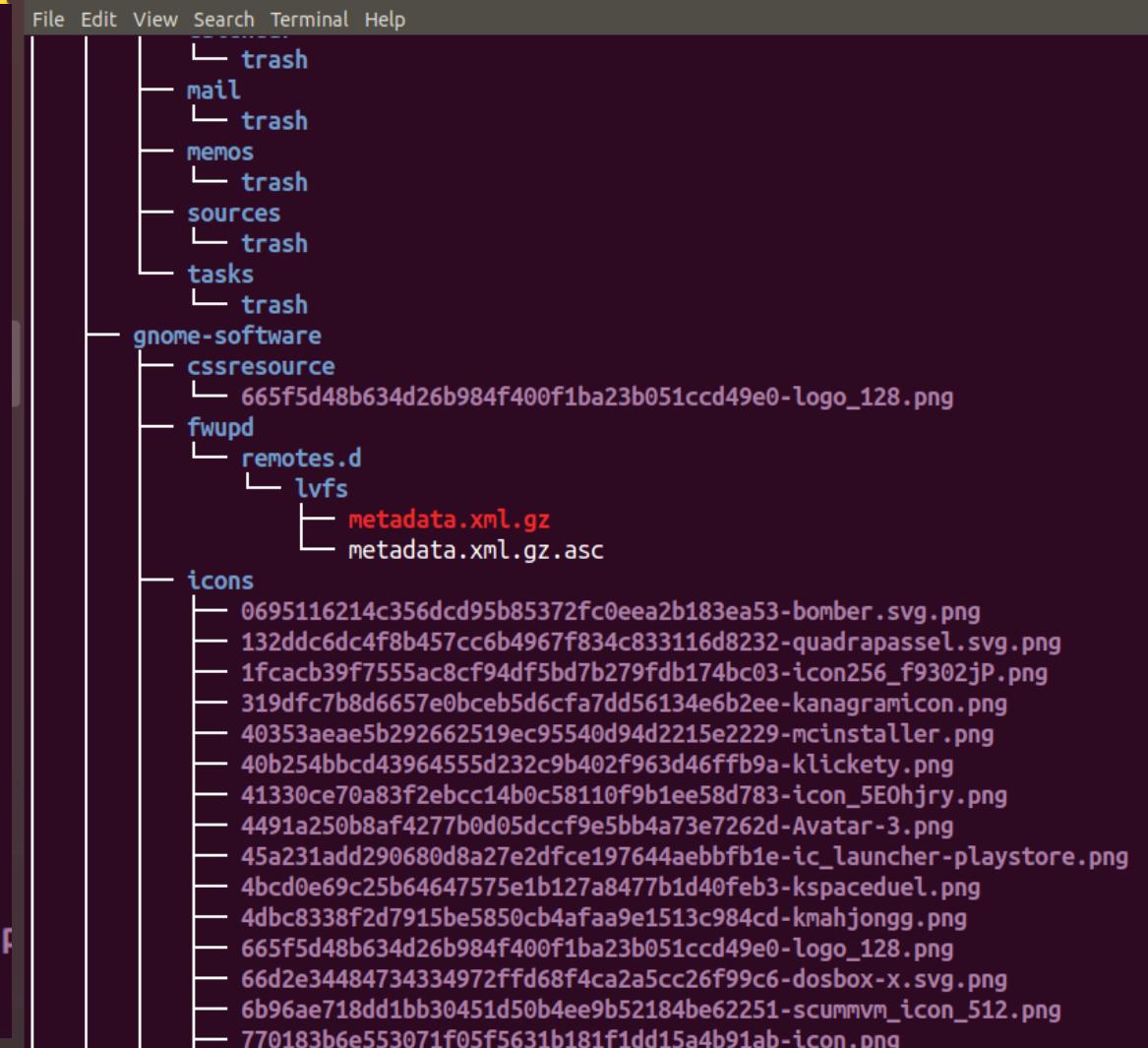


```
PROCESSING TRIGGERS FOR MANUDD (2.0.0-2) ...
ahmed@ubuntu:~$ tree
.
├── Desktop
├── Documents
│   ├── notes.txt
│   └── profile.txt
├── Downloads
├── Music
├── Pictures
├── Public
├── Templates
└── Videos
```

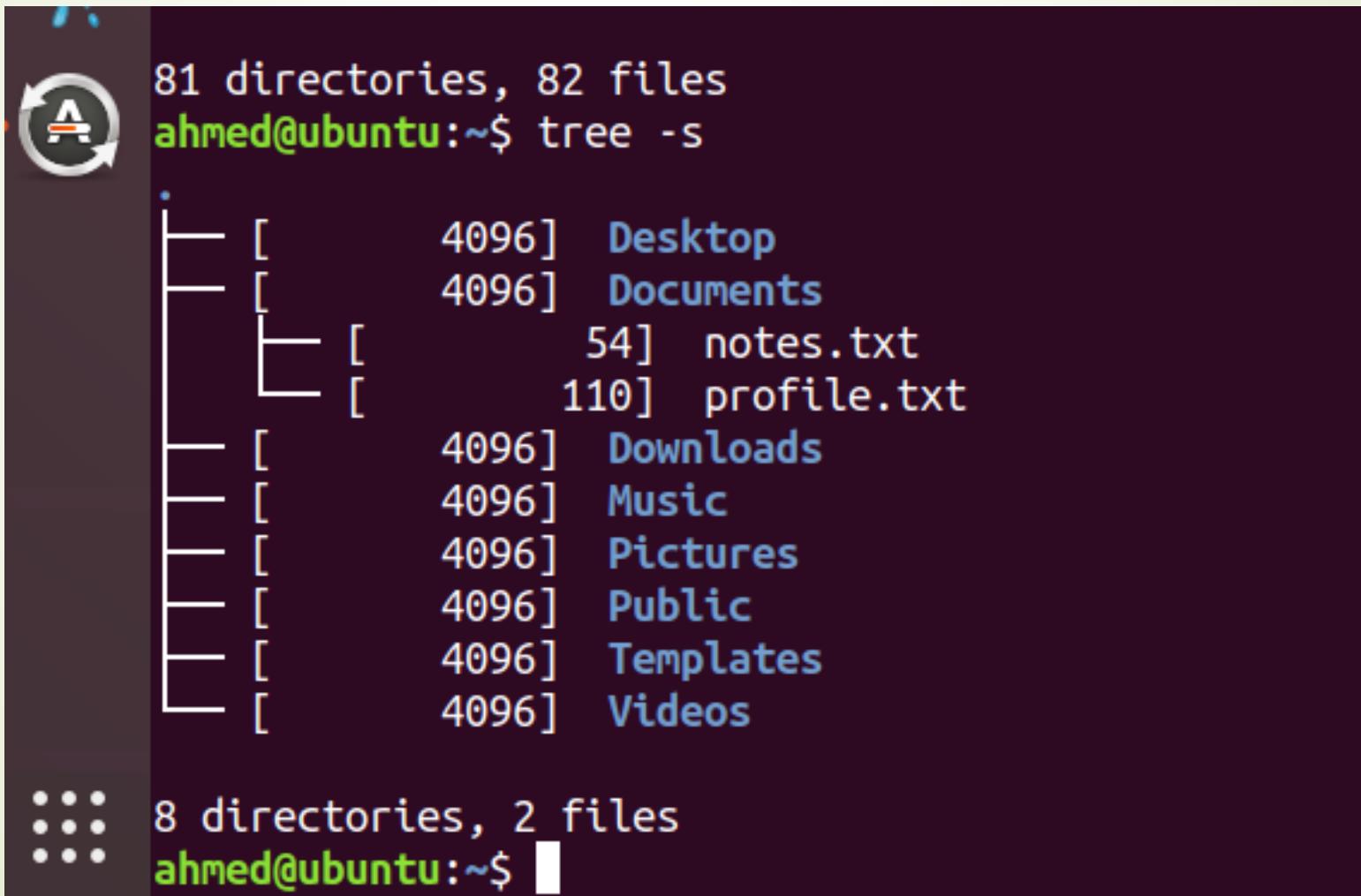
# tree -a will display hidden files along with directories and files.

```
8 directories, 2 files
ahmed@ubuntu:~$ tree -a
```

```
.
├── .bash_logout
├── .bashrc
├── .cache
│   ├── evolution
│   │   ├── addressbook
│   │   │   └── trash
│   │   ├── calendar
│   │   │   └── trash
│   │   ├── mail
│   │   │   └── trash
│   │   ├── memos
│   │   │   └── trash
│   │   ├── sources
│   │   │   └── trash
│   │   ├── tasks
│   │   │   └── trash
│   └── gnome-software
│       ├── cssresource
│       │   └── 665f5d48b634d26b984f400f1ba23b051ccd49e0-logo_128.p
│       ├── fwupd
│       └── remotes.d
```



tree -s will display the file size as shown below. While using this option, it prints out the size of the files along with the file names.



```
81 directories, 82 files
ahmed@ubuntu:~$ tree -s
.
├── [ 4096] Desktop
├── [ 4096] Documents
│   ├── [ 54] notes.txt
│   └── [ 110] profile.txt
├── [ 4096] Downloads
├── [ 4096] Music
├── [ 4096] Pictures
├── [ 4096] Public
├── [ 4096] Templates
└── [ 4096] Videos

8 directories, 2 files
ahmed@ubuntu:~$
```

# Command History in Linux | history Command

► The **history command** in **Linux** displays a list of commands that were previously entered in the terminal. By default, it shows the last **1000 commands**, but this can be configured. This feature allows users to recall, reuse, and modify commands without having to retype them.

## Display a Limited Number of Commands History



```
ahmed@ubuntu:~$ history
 1 cd Documents
 2 touch notes.txt
 3 grep "Linux" notes.txt
 4 ls -l notes.txt
 5 grep lines notes.txt
 6 cat notes.txt
 7 grep command notes.txt
 8 grep -i grep notes.txt
 9 grep -c grep notes.txt
10 grep -w grep notes.txt
11 clear
12 grep -o files notes.txt
13 grep -n files notes.txt
14 grep -v files notes.txt
15 grep [i] notes.txt
16 grep ^grep notes.txt
17 cat notes.txt
18 clear
19 grep ^root notes.txt
20 cat notes.txt
21 grep ^c notes.txt
22 grep 'y$' notes.txt
23 grep y$ notes.txt
24 grep ^root notes.txt
25 -----
```

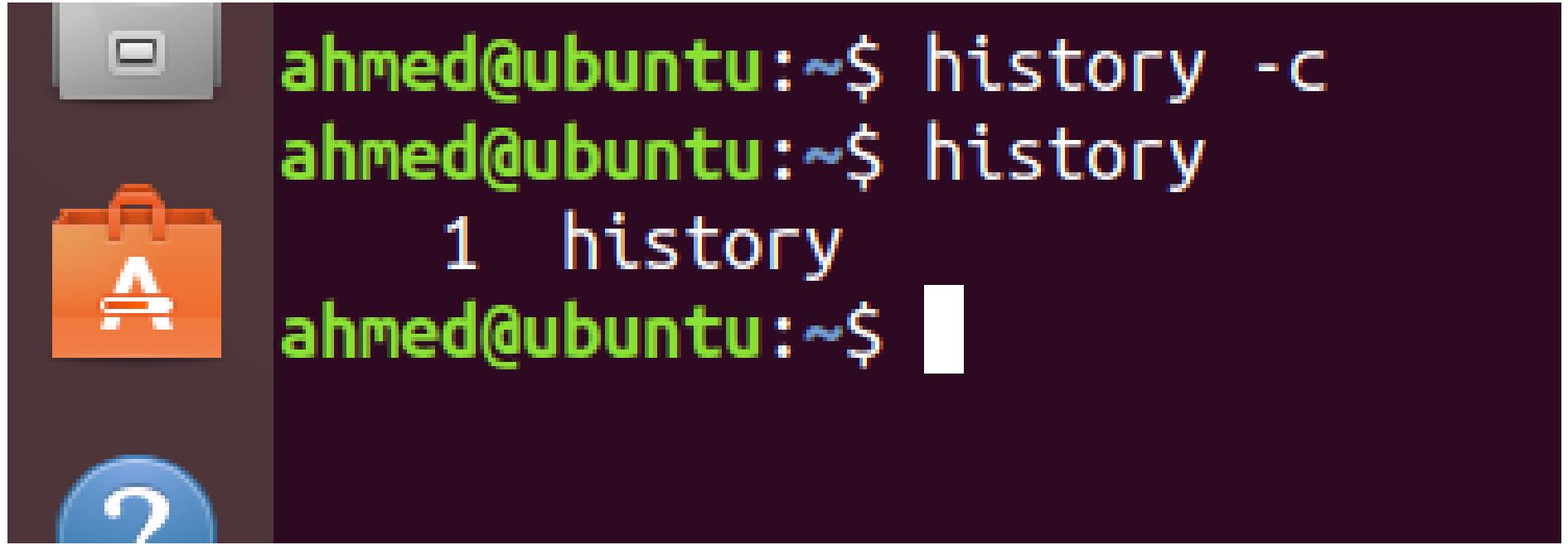
# Display a Limited Number of Commands History

```
61 history
ahmed@ubuntu:~$ history 10
 53 touch red.txt
 54 cat red.txt
 55 grep re*d red.txt
 56 grep r[oe]*d red.txt
 57 clear
 58 grep 'e*' red.txt
 59 cd
 60 histroty
 61 history
 62 history 10
ahmed@ubuntu:~$ █
```

# View the Last 10 Commands History



```
fg19181@GFGNDDLLT448:~$ history | tail
 67 cat capital.txt
 68 wc state.txt
 69 wc state.txt capital.txt
 70 wc -c state.txt
 71 wc -c state.txt capital.txt
 72 wc -m state.txt
 73 wc -L state.txt
 74 cd reactnew
 75 npm start
 76 history | tail
fg19181@GFGNDDLLT448:~$ 
```

A screenshot of a Ubuntu desktop environment. On the left, there's a dock with three icons: a grey window icon, an orange shopping bag icon with a white letter 'A' on it, and a blue circle icon with a white question mark. To the right of the dock is a terminal window with a dark background and light-colored text. The terminal shows the following session:

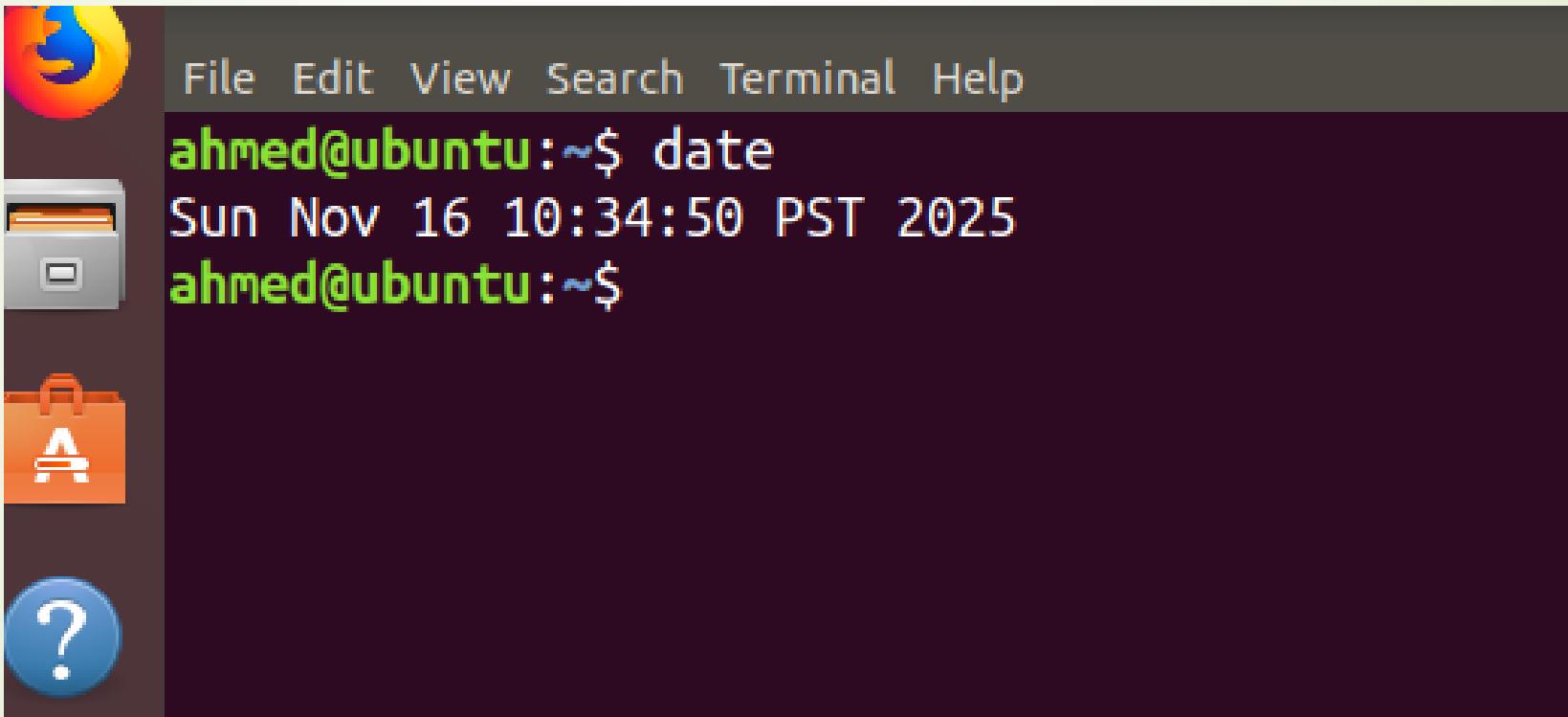
```
ahmed@ubuntu:~$ history -c
ahmed@ubuntu:~$ history
1 history
ahmed@ubuntu:~$
```

The terminal window has a white cursor at the end of the last command line.

## Clear Entire Command History

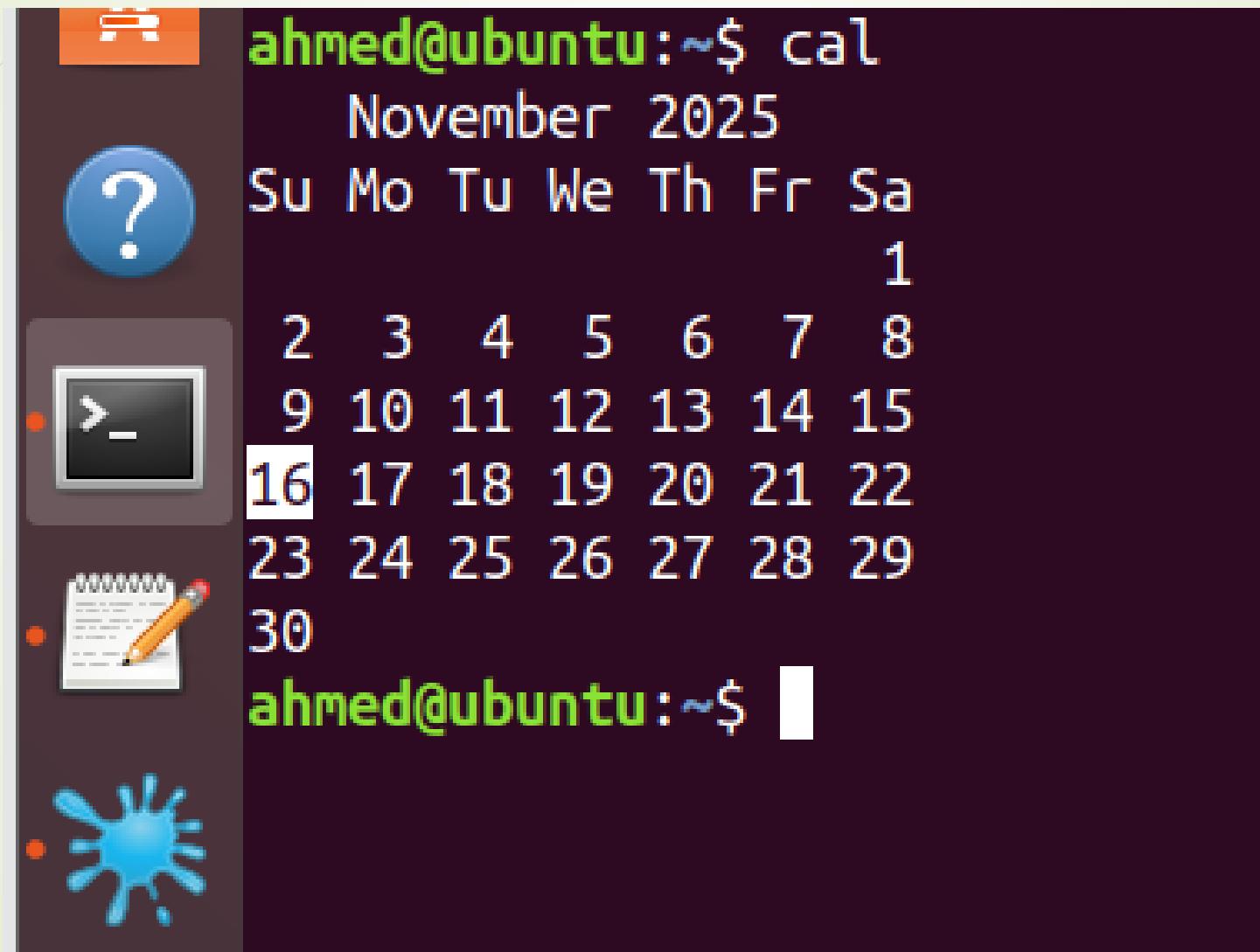
- ▶ Clear the entire command history using the `history -c` option.

# date command to display the date and time

A screenshot of a Linux desktop environment, specifically Ubuntu, showing a terminal window. The terminal window has a dark background and a light-colored menu bar at the top with options: File, Edit, View, Search, Terminal, and Help. The user's session is identified as ahmed@ubuntu:~\$ . Below the menu, the terminal displays the command 'date' followed by the current date and time: 'Sun Nov 16 10:34:50 PST 2025'. The prompt 'ahmed@ubuntu:~\$' appears again at the bottom. To the left of the terminal window, there is a vertical dock containing several icons: a blue circular icon with a question mark, an orange folder icon, a grey document icon, and a red folder icon with a white letter 'A' on it. The overall background of the desktop is a light beige color with some faint, curved lines.

```
File Edit View Search Terminal Help
ahmed@ubuntu:~$ date
Sun Nov 16 10:34:50 PST 2025
ahmed@ubuntu:~$
```

# show calendar



```
ahmed@ubuntu:~$ cal
November 2025
Su Mo Tu We Th Fr Sa
                    1
 2  3  4  5  6  7  8
 9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30
ahmed@ubuntu:~$
```

# References

- ▶ Ramses van Zon," Securing File Access Permissions on Linux ", SciNet HPC, University of Toronto ,27 October 2022.
- ▶ <https://www.geeksforgeeks.org/linux-unix/grep-command-in-unixlinux/>
- ▶ <https://www.geeksforgeeks.org/linux-unix/tree-command-unixlinux/>
- ▶ <https://www.geeksforgeeks.org/linux-unix/history-command-in-linux-with-examples/>