

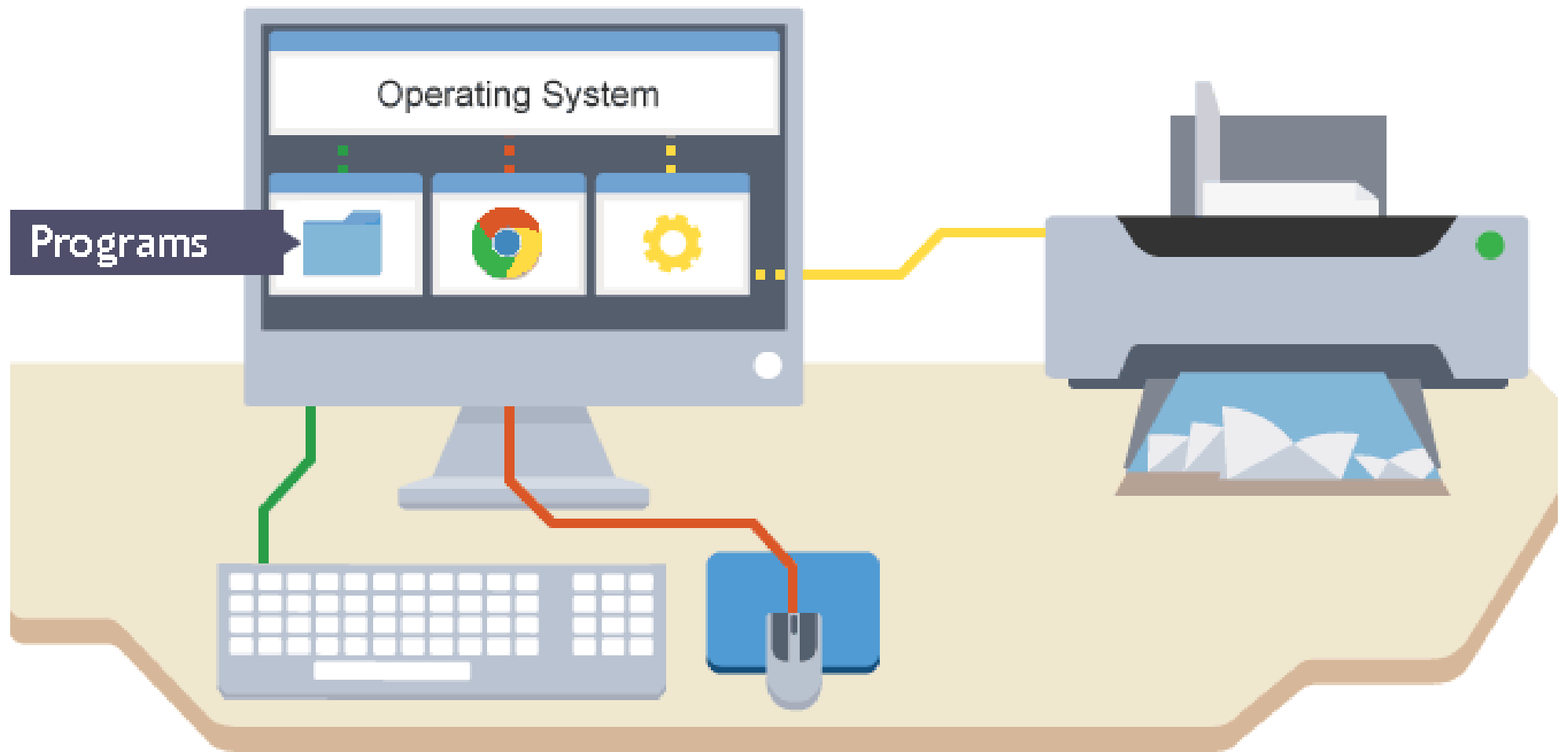
EECS 2031 - Software Tools

Lab 2 Tutorial: Introduction to UNIX/Linux

Tilemachos Pechlivanoglou
(tipech@eecs.yorku.ca)

Material marked with 
will be in your exams

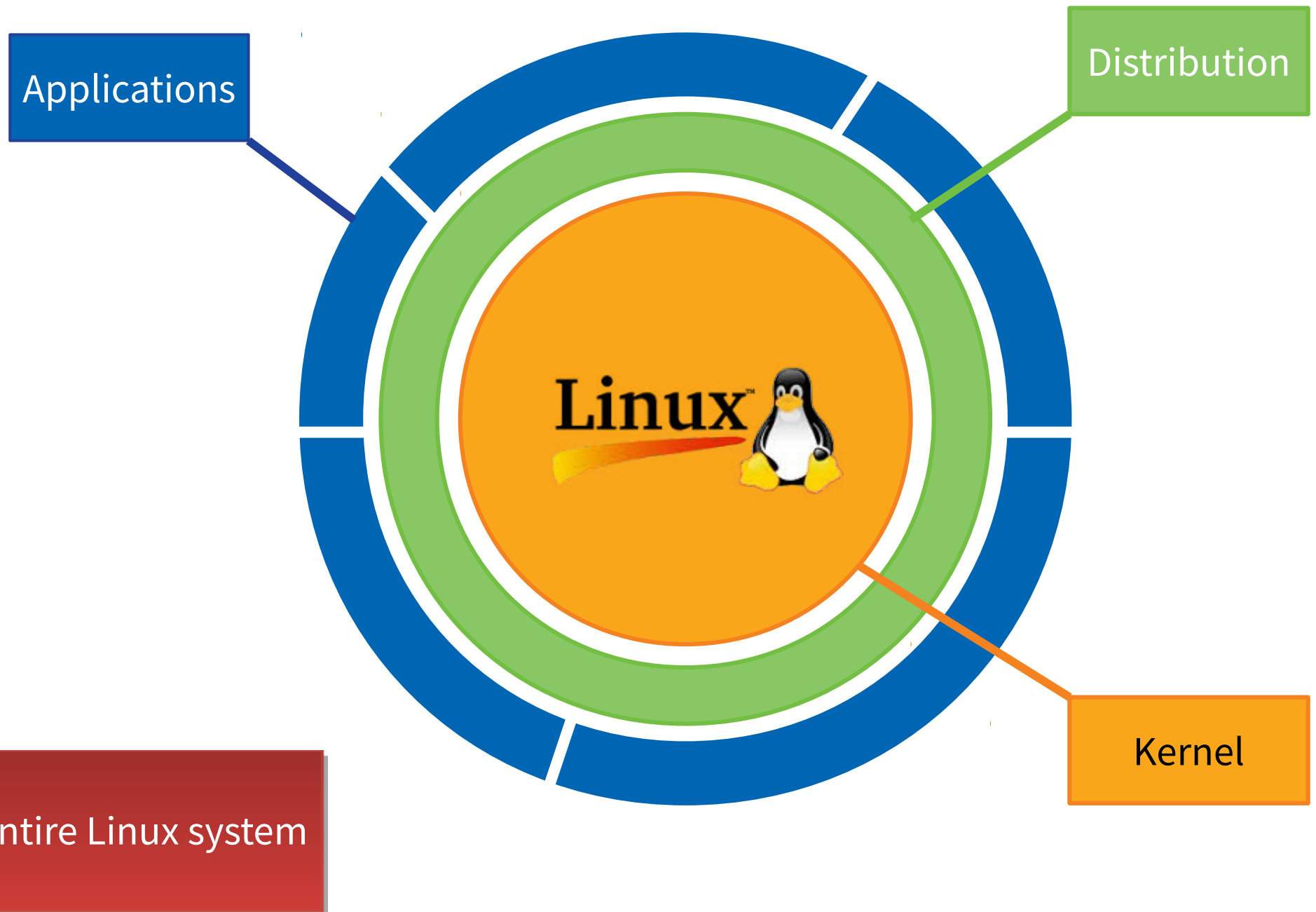
Introduction to Unix

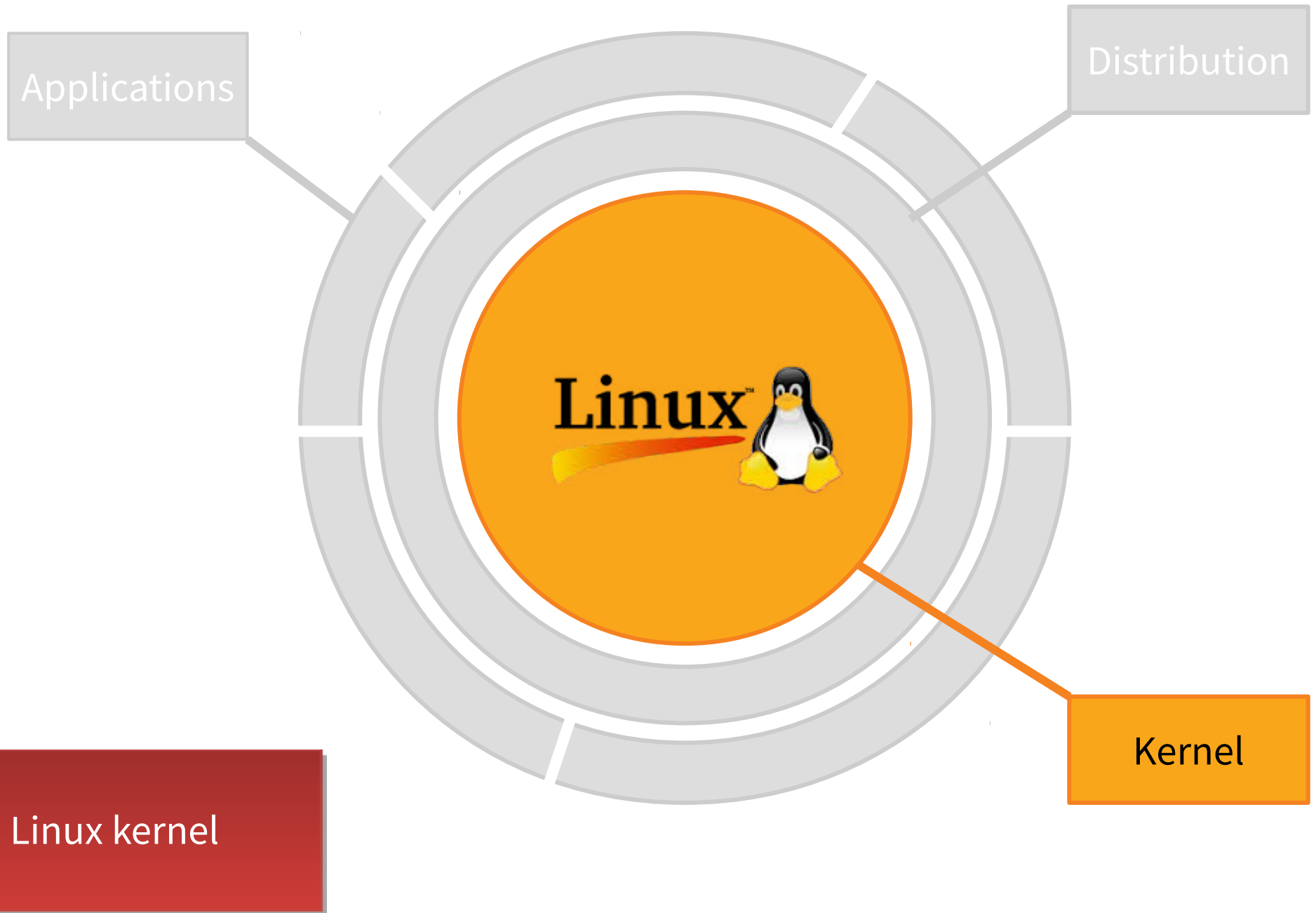


Linux is an operating
system (OS)



It was based on UNIX





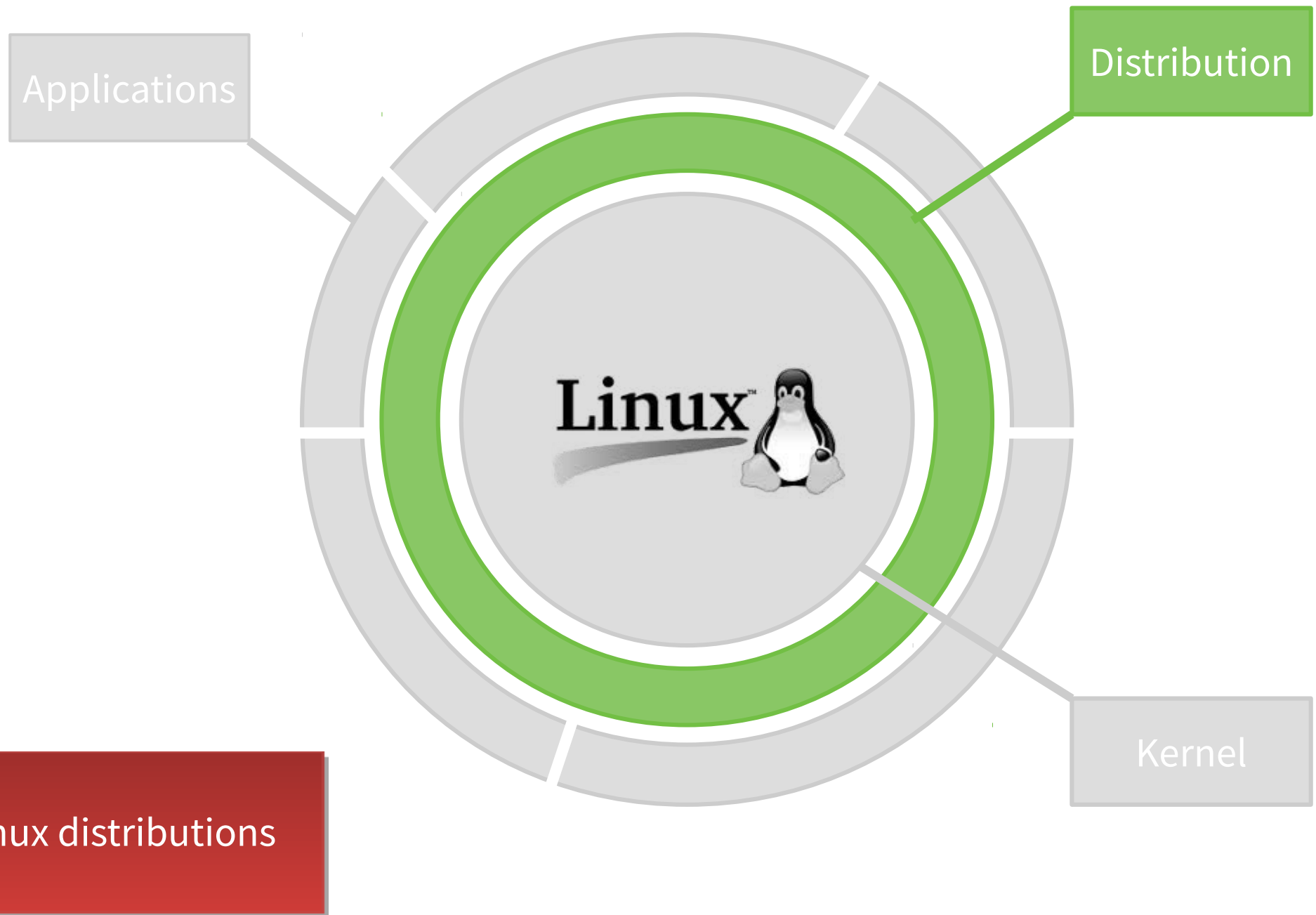
Linux kernel

Written in C by Linus Torvald in 1991

Handles low-level actions like:

- CPU usage
- RAM/memory usage
- Hard disk read/write
- Filesystem management





Linux distributions

Linux Distributions (distros)

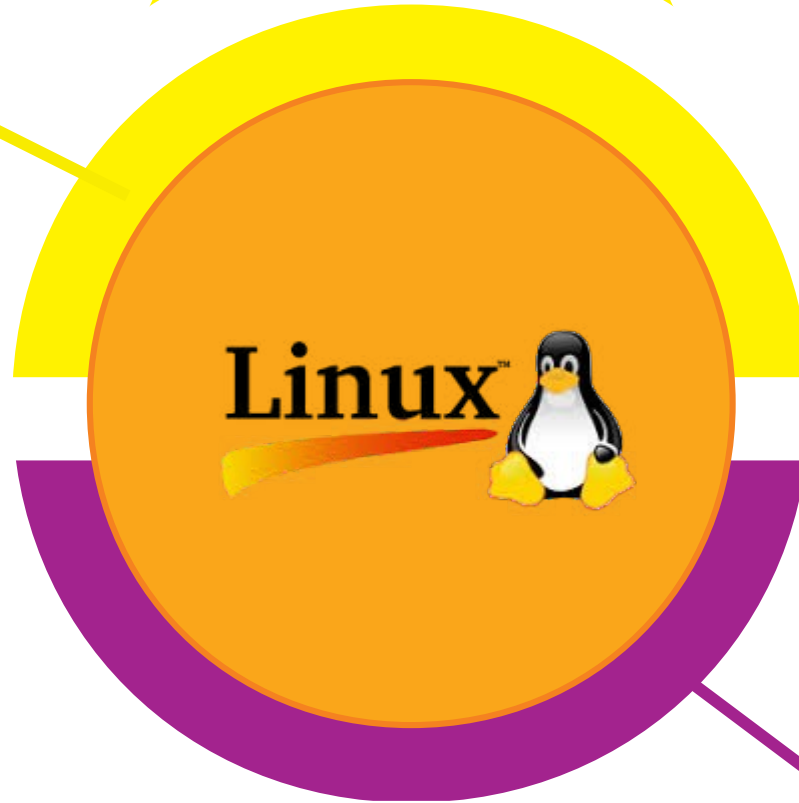
Bundles containing:

- the kernel
- all the tools needed to interact with it
- application/program store and installer
- other tools like device drivers, etc.

Most popular Linux distros:

- Ubuntu
- CentOS
- RedHat
- Fedora
- RaspBian
- openSUSE

Shell



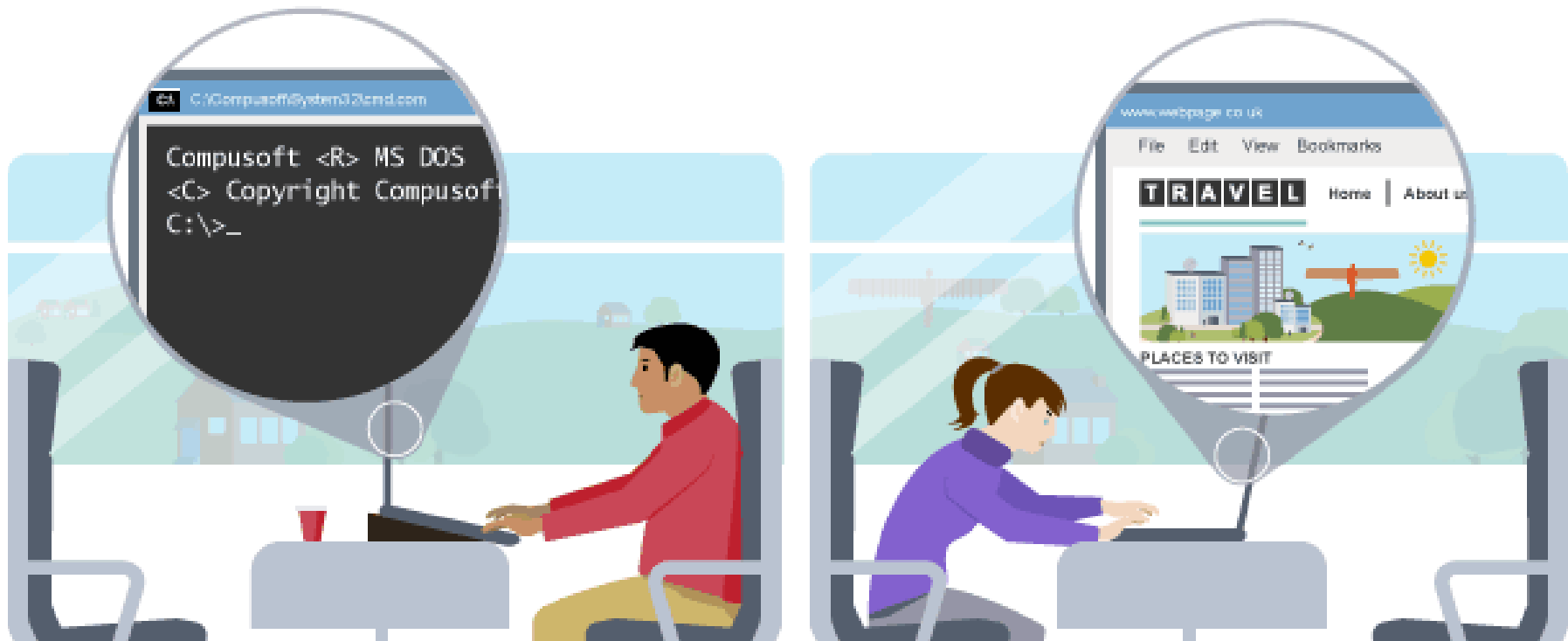
Graphical shell

Linux shell

Linux shell

Shell:

- interface between kernel & outside world (user)
- can be through a command line or graphical



Linux graphical shell

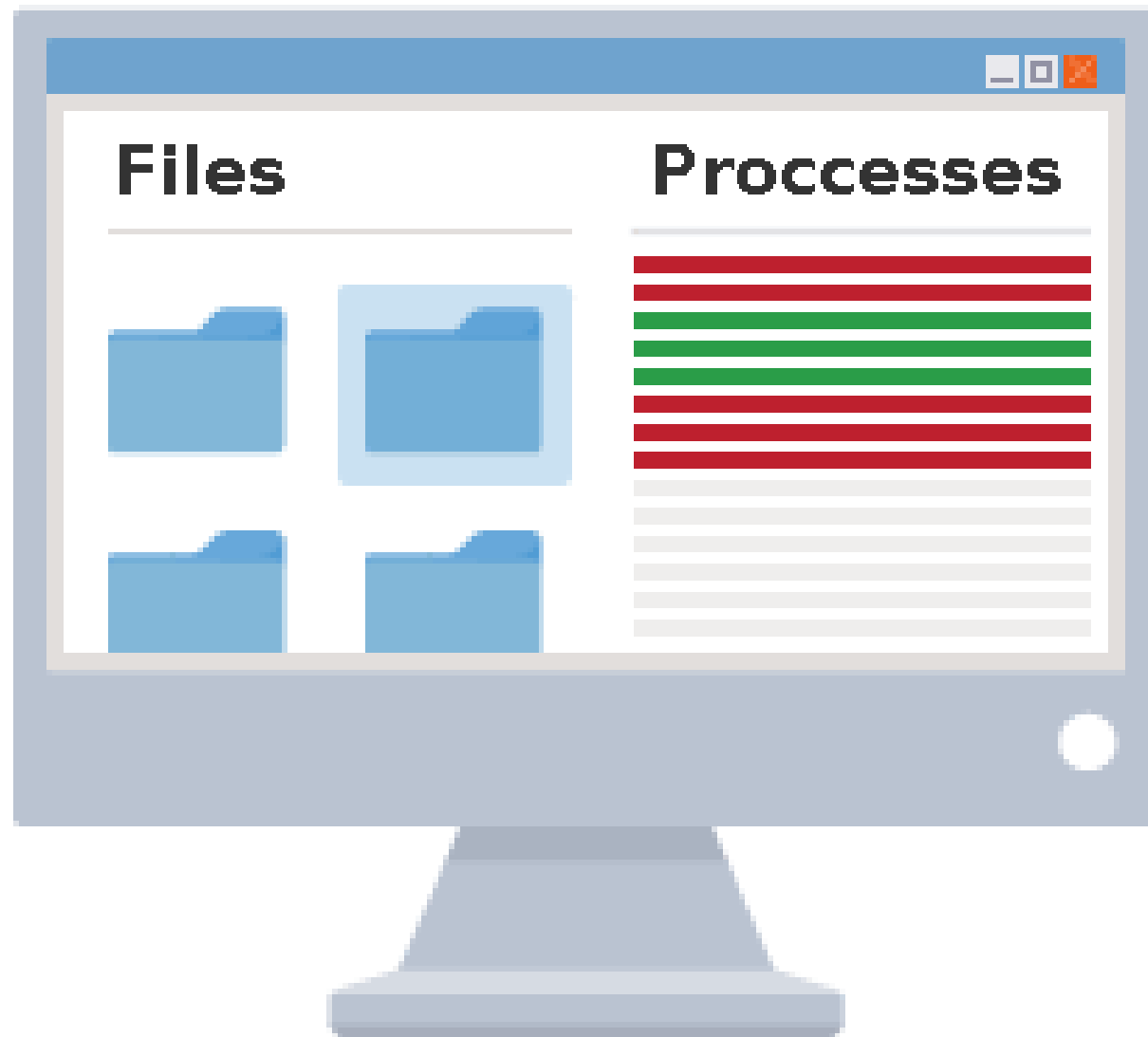
The graphical shell:

- is much more **user friendly**
- web browsers, image editors, etc. use it
- is similar to Windows or MacOS user interfaces
- has many variations, with GNOME most popular
- provides access to the command line interface (CLI) through the “**Terminal**” program

Linux basic shell

The basic command line shell (or just shell) :

- is much more **powerful** and **versatile** in Linux
- many programs only use this instead of a UI
- has many variations, **tcsh** and **bash** are very popular
- usually comes with helpful features, like:
 - **filename completion** with **Tab** key
 - **command history** with **Up** or **Down** arrow keys

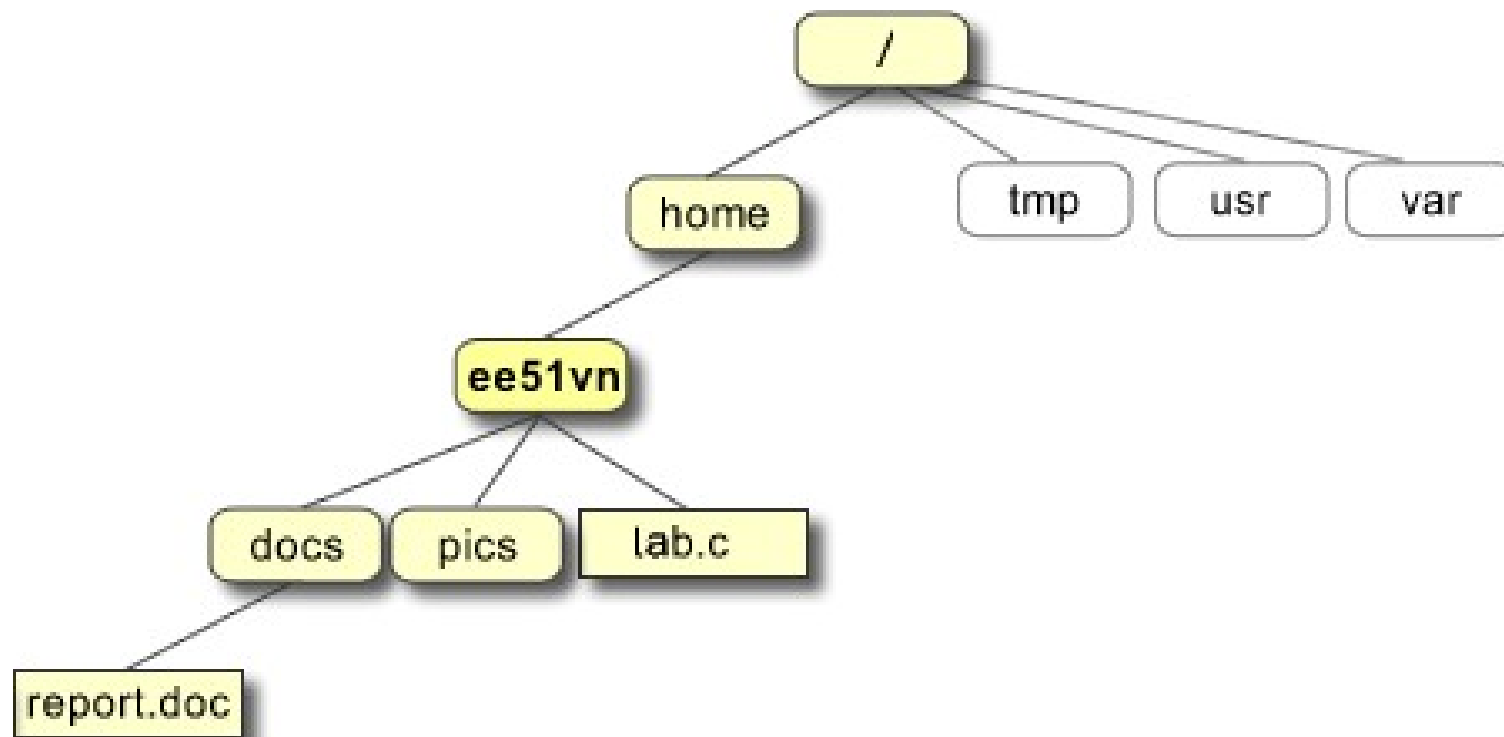


Files and processes

Files and processes

Everything in Linux is either a file or a process

- Process:
 - executing program
 - has a unique **PID** (process identifier)
- File:
 - a collection of data
 - created by the system or the user through programs
 - its **path** is its **location** in the system plus its **name**
 - files starting with “.” are **hidden**



Directory structure

Directory structure (1/2)

In Linux, files are organised in groups called **directories** (folders).

- Directories can be nested (sub-folders)
- The result is a **tree** structure
 - Every file and folder is located on that tree
- The top of the tree is always the **root** directory(“/”)
- A folder is symbolized with “/” after its name
 - e.g. Documents/
- A file or folder’s **path** is its **location** plus its **name**
 - e.g. /home/tipech/Documents/report.doc

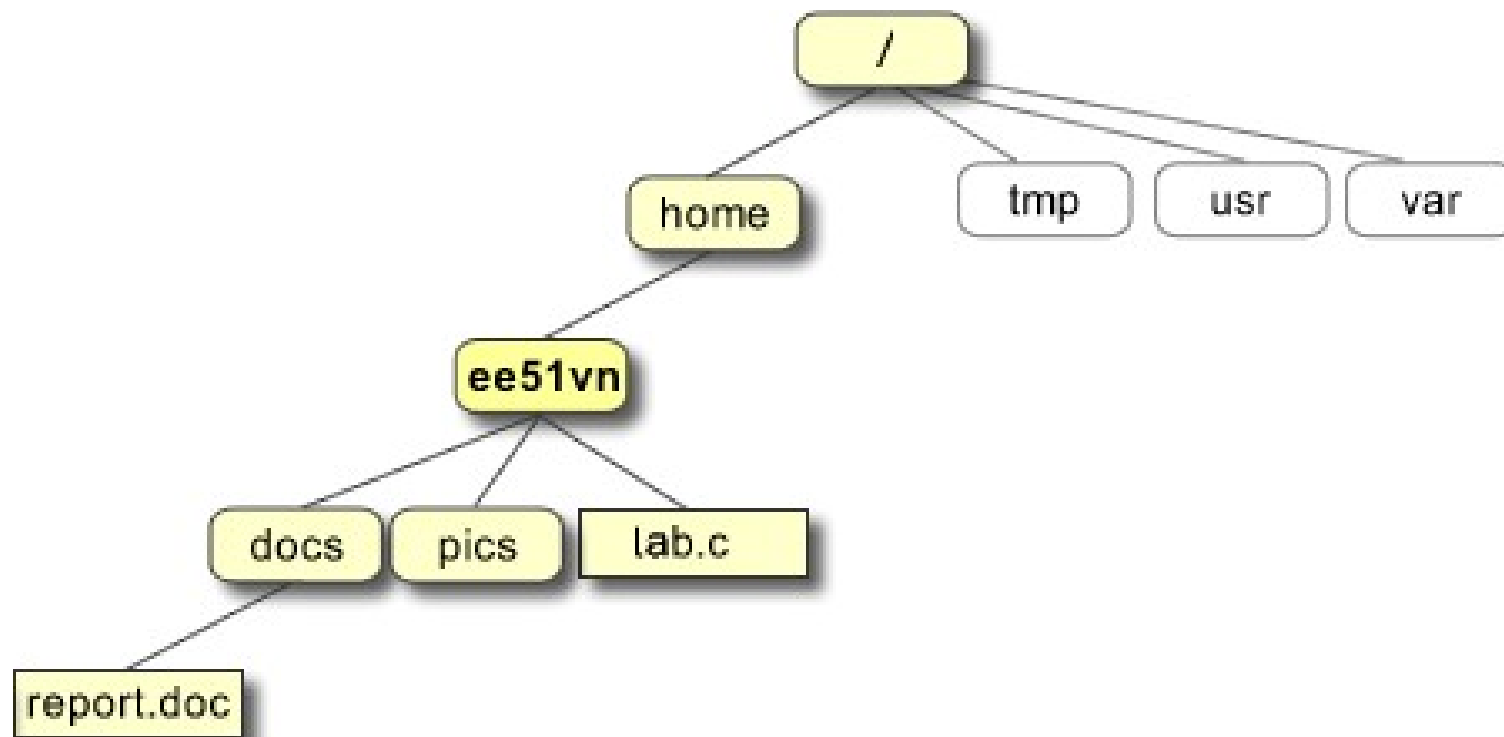
Directory structure (2/2)

Root directory:

- has various system folders (usr, bin, etc, ...)
- has a folder called “home”
- has inside “home” a folder for every PC user
 - lab PCs have “virtual” users, only your username appears
 - this is your **home** directory

Home directory:

- contains only your stuff (e.g. Documents, Pictures)
- has the alias “~”, short for “/home/username/”



Directory structure

Root directory:

User home directory here:

Contains 2 folders:

Home contains 1 file:

docs folder contains 1 file:

File's full path:

/

ee51vn

docs, pics

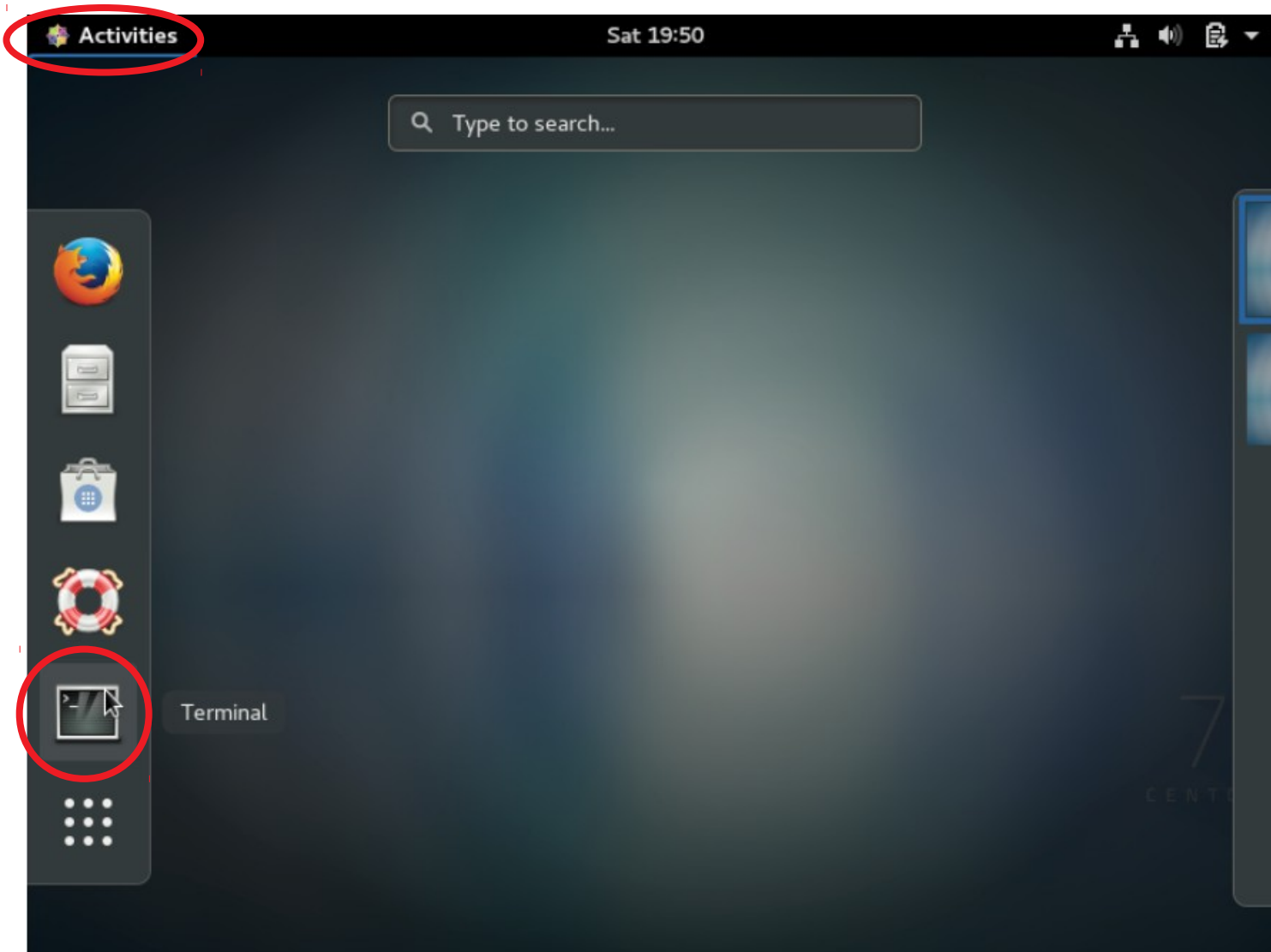
lab.c

report.doc

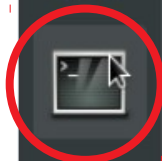
/home/ee51vn/docs/report.doc

Working with directories

1



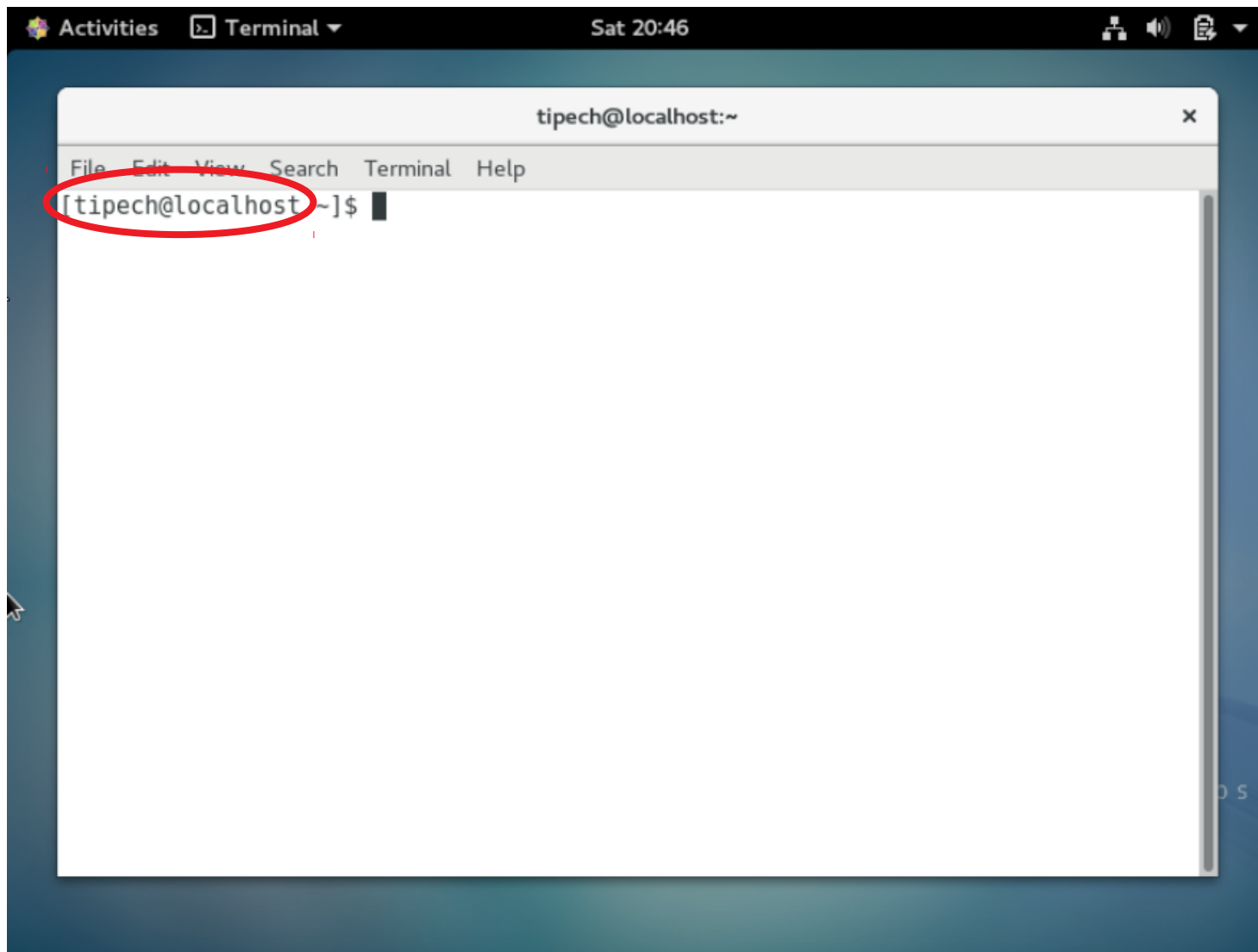
2



Terminal

Launching a terminal

**user@
computer**



Terminal window

Working directory



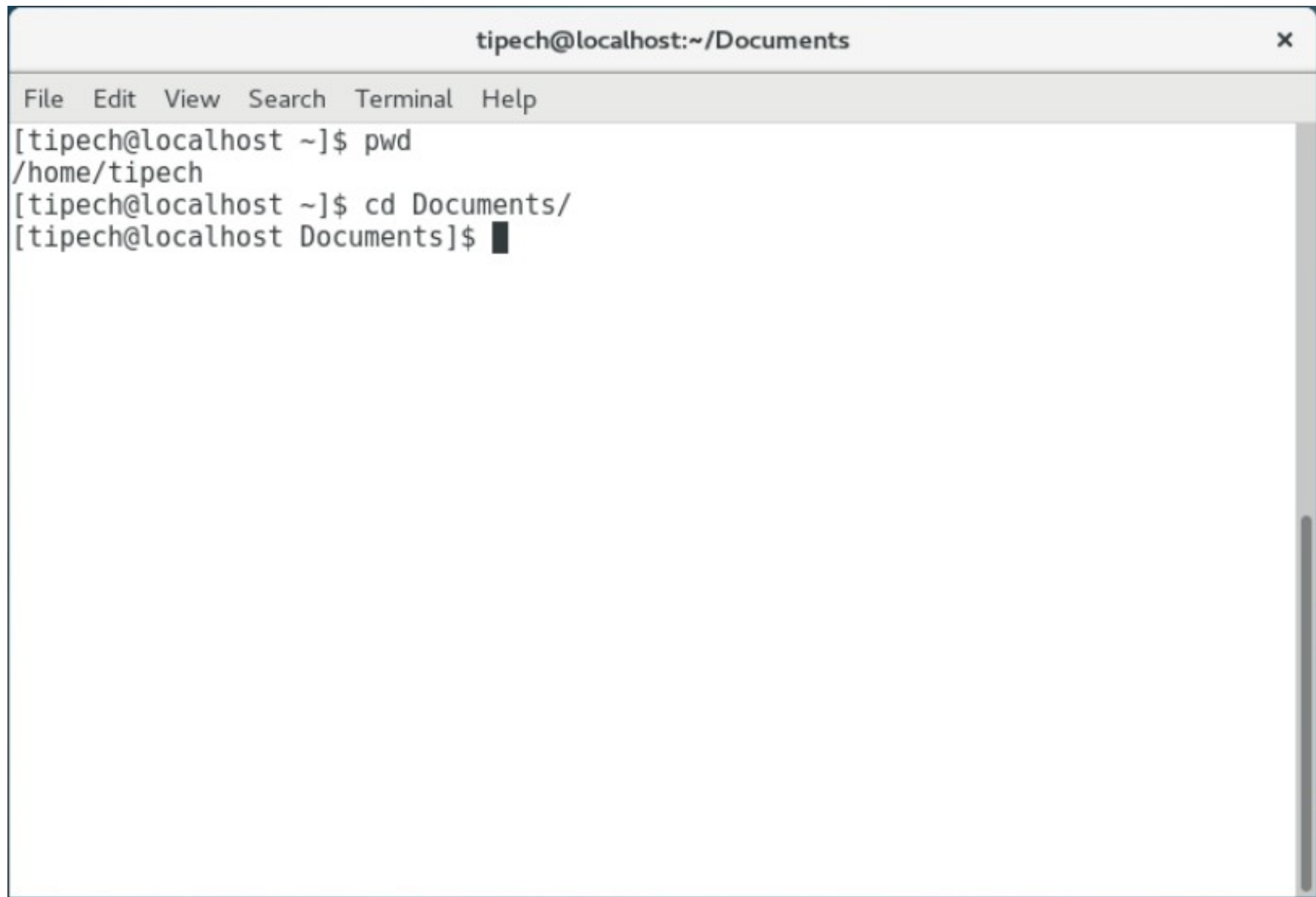
The terminal “points” at any time to a directory

- Just as a web browser “points” at a website or sub-page
- We can change directory to navigate the file system
- The current directory is symbolized with “.”
- The parent directory (one level up) is symbolized with “..”

Use `pwd` to print the working directory

To change directory, use `cd`

- e.g. `cd Documents`

A terminal window titled "tipech@localhost:~/Documents" with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the following commands and output:

```
[tipech@localhost ~]$ pwd
/home/tipech
[tipech@localhost ~]$ cd Documents/
[tipech@localhost Documents]$
```

pwd & cd

List, clear



To list files/folders in a directory, use **ls** (list)

- e.g. `ls`, `ls /home/tipech`
- you can also use **ll** to list files with details

You can pass command arguments with **-<arg>**

- e.g. `ls -a` (prints everything, including hidden files)

To clear the terminal screen, use **clear**

```
tipech@localhost:~  
File Edit View Search Terminal Help  
[tipech@localhost ~]$ ls  
Desktop Documents Downloads Music Pictures Public Templates Videos  
[tipech@localhost ~]$ ls -a  
.  
..  
.bash_history .bash_profile Desktop .ICEauthority Public  
.bash_logout .bashrc Documents .local Templates  
.cache Downloads Music Videos  
.config .esd_auth Pictures  
[tipech@localhost ~]$ ll  
total 0  
drwxr-xr-x. 2 tipech tipech 6 Sep 23 19:37 Desktop  
drwxr-xr-x. 2 tipech tipech 6 Sep 23 19:37 Documents  
drwxr-xr-x. 2 tipech tipech 6 Sep 23 19:37 Downloads  
drwxr-xr-x. 2 tipech tipech 6 Sep 23 19:37 Music  
drwxr-xr-x. 3 tipech tipech 119 Sep 23 20:55 Pictures  
drwxr-xr-x. 2 tipech tipech 6 Sep 23 19:37 Public  
drwxr-xr-x. 2 tipech tipech 6 Sep 23 19:37 Templates  
drwxr-xr-x. 2 tipech tipech 6 Sep 23 19:37 Videos  
[tipech@localhost ~]$
```

ls, ls -a, ll

Make directory, remove



To make a new directory, use `mkdir`

- e.g. `mkdir Test`

To remove files or empty folders, use `rm/rmdir`

- e.g. `rm Documents/report.doc`, `rmdir Test`

If a folder has items inside, you need to use recursive removal with `-r`

- e.g. `rm -r Documents`

```
tipech@localhost:~  
File Edit View Search Terminal Help  
[tipech@localhost ~]$ ls  
Desktop Documents Downloads Music Pictures Public Templates Videos  
[tipech@localhost ~]$ mkdir Test  
[tipech@localhost ~]$ ls  
Desktop Documents Downloads Music Pictures Public Templates Test Videos  
[tipech@localhost ~]$ rmdir Test  
[tipech@localhost ~]$ ls  
Desktop Documents Downloads Music Pictures Public Templates Videos  
[tipech@localhost ~]$
```

mkdir, rmdir

Copy, move



To copy a file use **cp** (-r for non-empty folders)

- the usage is `cp <original_path> <target_path>`
- e.g. `cp Documents/report.txt .`
- this copies the file to the current directory, “.”
- don’t forget the space between paths!

Similarly, to move a file, use **mv** (-r)

- e.g. `mv Documents/report.txt .`

```
tipech@localhost:~  
File Edit View Search Terminal Help  
[tipech@localhost ~]$ ls  
Desktop Documents Downloads Music Pictures Public Templates Videos  
[tipech@localhost ~]$ ls Documents/  
report.txt  
[tipech@localhost ~]$ mv Documents/report.txt .  
[tipech@localhost ~]$ ls  
Desktop Downloads Pictures report.txt Videos  
Documents Music Public Templates  
[tipech@localhost ~]$ cp report.txt Documents/  
[tipech@localhost ~]$ ls  
Desktop Downloads Pictures report.txt Videos  
Documents Music Public Templates  
[tipech@localhost ~]$ ls Documents/  
report.txt  
[tipech@localhost ~]$
```

cp, mv

File contents



To print a file's contents use **cat** (concatenate)

- e.g. `cat Documents/report.txt .`

Alternatively, use **head**, **tail** or **less**

- `head` prints the first 10 lines of a file
- `tail` prints the last 10
- `less` prints everything, one page at a time

```
tipech@localhost:~  
File Edit View Search Terminal Help  
[tipech@localhost ~]$ cat report_short.txt  
These are the contents of file report.txt  
2nd line  
3rd line  
4th line  
5th line  
[tipech@localhost ~]$ tail report_long.txt  
16th line  
17th line  
18th line  
19th line  
20th line  
21st line  
22nd line  
23rd line  
24th line  
25th line  
[tipech@localhost ~]$
```

cat, tail

Search



To search for something in a file use **grep**

- use the **-i** argument for non case sensitive searches
- use **-v** to display lines that do not match
- use **-n** to print line numbers in results
- use **-c** to print the count of matches instead
- you can combine arguments, e.g. **-ivc**
- e.g. **grep line -i report.txt**

To count the number of words, use **wc**

```
tipech@localhost:~  
File Edit View Search Terminal Help  
[tipech@localhost ~]$ grep line report_short.txt  
2nd line  
3rd line  
4th line  
5th line  
[tipech@localhost ~]$
```

grep

Info, man pages, interrupt

For help, **man** `<command>` or **info** `<command>`

– e.g. `man ls`

If you are stuck in a program at any time, **Ctrl+C** sends an **interrupt signal** (stop) to it

To **copy** or **paste** in Terminal, use **Ctrl+Shift+C** and **Ctrl+Shift+V**, respectively

Permissions, sudo

Linux user permissions:

- As a regular user, you can only modify files and folders under your home directory.
- For many actions, **super user** (administrator) privileges are required
- Do that (in non-lab PCs) with **sudo <command>**
 - e.g. `sudo cp ~/report.txt /usr/`
 - Alternatively, log in as SU with `sudo su` and run commands normally

Programming in C

Creating a program

Steps for making a program in C:

1. Break your goal into smaller pieces
2. **Write** code (**source**) in a text editor for each one
3. **Compile** the source into an executable (program)
4. **Run** the program
5. Find out what went wrong/right (**debugging**)
6. Back to step 2 until you are done

Writing a program

You need to create a C file (.c) with your code

- To do that, you use a **text editor**
- For this lab (and most Linux distros) you use **gedit**
- In gedit, you can write and save a file as <name>.c
- Save that file anywhere in your home directory

1

Activities

Sat 23:36



2

gedit

3



Text Editor

7
CENTOS

Launching gedit



gedit Text Editor

Compiling a program

For the next step, compile the source:

- Open a Terminal window
- Navigate to where you saved the source code
- Compile it with the GNU C Compiler by typing
 - `gcc source.c` or `cc source.c`
- This creates the executable **a.out**
 - Alternatively, you can name the output file with:
 - `cc source.c -o executable`

Running a program

To run the executable, simply enter its name

- e.g. `a.out`
- If it doesn't terminate on its own, use **Ctrl+C** to stop

These 3 steps are combined in an **IDE** program (Integrated Development Environment)

- There you can compile and run with a single button
- Such programs are Eclipse, Visual Studio, Geany
- You can use **Geany** for C development in the lab

Thank You!