echo "Welcome Back!" >> Session4

#cat Agenda.txt

Part 1

- Users
- Groups
- File Permissions and Ownership

Part 2

- Text Editors
- Nano
- Vim

What are users?

In Linux, users are individual accounts that allow people or processes to access the system. Each user has a unique identity (user ID) and their own home directory, which keeps their files separate from others.

Users Importance

Having multiple users is important because:

- **Multi-user support**: Allows multiple people to use the system safely.
- **Security**: Users have separate permissions to protect files.
- **Privacy**: Each user has their own space and settings.
- **Accountability**: Actions are tracked by user accounts.

Users Types

Superuser (root)
UID = 0

- Full control over the system
- Can modify any system files and manage users
- Runs powerful administrative commands

System Users
UID 1–999

- Created by the system or installed services
- Used to run background services (daemons)
- Limited permissions for security isolation

Regular Users
UID ≥ 1000

- Accounts for normal human users
- Can perform daily tasks with limited access
- Use **sudo** when admin actions are needed

What are groups?

A **group** is a collection of users who share access permissions to files and directories.

Groups help manage permissions more easily for multiple users.
A group has a **group name** and a **GID (Group ID)** - a unique number used internally by the system.

Why we have groups?

- **Simplify management:** Set permissions for multiple users at once.
- **Collaboration:** Group members can easily share files.
- **Security:** Restrict access to specific users only.
- **Flexibility:** Users can belong to multiple groups for different access levels.

Groups Types

Primary Group

- The default group assigned to a user when their account is created.
- Usually has the **same name as the user**.
- Any files created by that user **belong to this primary group**.
- Each user has **only one** primary group.

Supplementary Groups

- Additional groups a user can join besides their primary group.
- Allow users to access shared resources with other teams.
- A user can belong to **multiple secondary groups** at the same time.

COMMANDS!

AT FIRST we need to know which user is logged in and its id!

Whoami command is used to know the current user.

```
jurykassem@juryahmed:~$ whoami
jurykassem
```

id command is used to know the current user's id.

```
root@juryahmed:/home/jurykassem# id
uid=0(root) gid=0(root) groups=0(root)
```

Dealing with Users

In etc/passwd everything about your users is stored

username:password:UID:GID:comment/home directory:shell

Dealing with Users

- Only **root** or a user with **sudo privileges** can create new accounts.
- Command syntax: **useradd [options] username**
- When executed without any option, useradd creates a new user account using the default settings specified in the /etc/default/useradd file.
- The variables defined in this file differ from distribution to distribution, which causes the useradd command to produce different results on different systems.
- **-m**: option to create a user with home directory.

Dealing with Users

- We can switch between users using **su username** command.
- You need to write - after su to have a home directory in the user. **su - username**
- The password for the user we are switching to is needed unless you are the root user.
- **userdel** is used to remove the details of username from /etc/passwd without removing the user's home directory by default. If the -r flag is specified, the userdel command also removes the user's home directory.

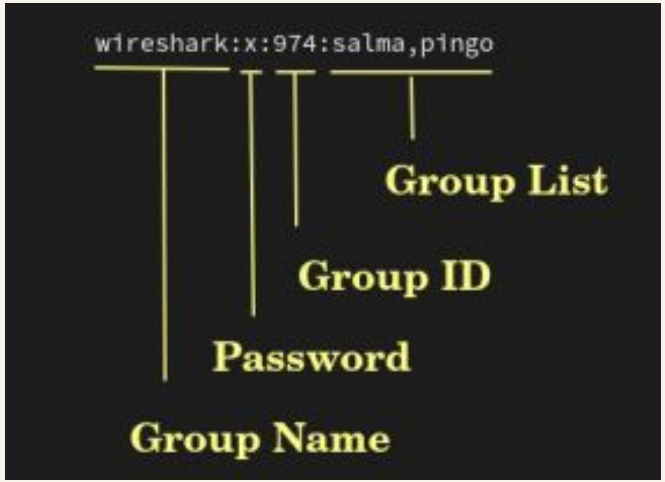
Setting passwords to users

- To be able to log in as the newly created user, you need to set the user password. To do that run the **passwd** command followed by the username.
- Only the root user or a user with sudo privilege can change or set password.

```
:~$ sudo useradd -m myuser
:~$ sudo passwd myuser
```

Dealing with Groups

In etc/group everything about your groups is stored



group_name:password:GID:group_list

Dealing with Groups

- Command Syntax : **groupadd [options] Group_name**
- To list all groups you are a member of use **groups** command.
- To list all groups of a specific user: **groups username** command.
- Only the root or a user with sudo privileges can create new groups.
- Groups are deleted using the groupdel command.

  Command syntax : **groupdel group_name**
- You **cannot delete** the primary group of a user account.

Dealing with Users and Groups

- Existing users accounts are added to groups using the **usermod** command.
- Command syntax : **usermod [options] <group name> <username>**
- So to add the user **myuser** to the group myGroup we will write: **usermod -a -G myGroup myuser**
- The **-G option** tells the command that we will add the user to a supplementary group. The **-a option** puts the command in append mode. Otherwise, the command will remove the user from all groups unspecified in the command.
- To remove a user from a group we use gpasswd command. This command accesses the /etc/group file and the **-d option** is for deleting the a user from a group
  Command syntax: **gpasswd -d username group_name**

Commands Recap

id	useradd	groupadd
su	userdel	groupdel
passwd	usermod	groups

HANDS ON TIME

Hands on

1. Create two users named `ahmed` and `sara` with a home directory.
2. Create two groups named `developers` and `testers`.
3. Set the password for user `ahmed`.
4. Switch from the current user to user `ahmed`, make a file with your name and switch back to your user.
5. Add user `ahmed` to the group `developers`.
6. Add user `sara` to the group `testers`.
7. Show all groups that user `ahmed` belongs to.
8. Change the primary group of user `ahmed` to `testers`.
9. Remove user `ahmed` from the `developers` group.
10. Delete user `sara`.
11. Delete the group `testers`.

Hands on

- sudo useradd -m ahmed
- sudo useradd -m sara
- sudo groupadd developers
- sudo groupadd testers
- sudo passwd ahmed
- su - ahmed

- sudo usermod -aG developers ahmed
- sudo usermod -aG testers sara
- groups ahmed
- sudo usermod -g testers ahmed
- sudo gpasswd -d ahmed developers
- sudo userdel sara
- sudo groupdel -f testers

Files Permissions and Ownerships

Every file or directory on Linux system has 3 possible permissions:

Read (r)

Gives you the permission to open and read a file or list the directories content

Write (w)

Gives you the permission to open and **modify** a file or add, remove and rename files stored in the directory.

Exectute (x)

Gives you the permission to **run** a file or to access the directories content. (By default, any newly created files are not executable.)

Types of File Owners in Linux

- **User (Owner)** – The person who created the file, has primary control over it.
- **Group** – A set of users who share permissions on the file.
- **Others** – All remaining users on the system who are neither the owner nor in the group

Changing permissions

The **chmod** command is used to change a file/directory's permissions.

There are two ways **Symbolic** mode and **Absolute** mode

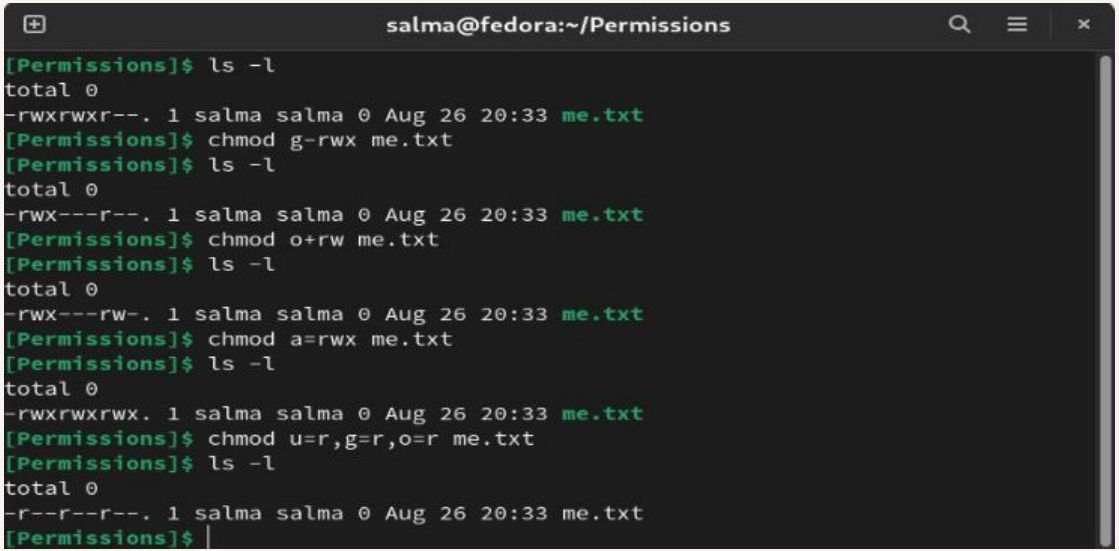
1. Symbolic Mode :

In symbolic mode, you can modify the permissions of a specific owner.
Syntax: **chmod [ownerType] [operator] [new permission] [file name]**

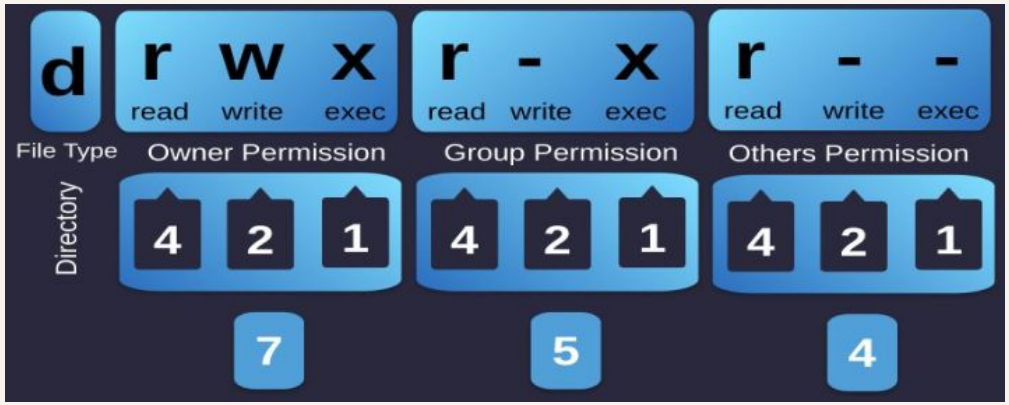
User	Denotations
u	user/owner
g	group
o	other
a	all

Operator	Description
+	Adds a permission to a file or directory.
-	Removes the permission.
=	Sets the permission and **overrides** the permissions set earlier.

1. Symbolic Mode :



2. Absolute Mode




```
[Permissions]$ ls -l
total 0
-rw-r--r--. 1 salma salma 0 Aug 26 20:33 me.txt
[Permissions]$ chmod 774 me.txt
[Permissions]$ ls -l
total 0
-rwxrwxr--. 1 salma salma 0 Aug 26 20:33 me.txt
[Permissions]$ chmod 774 me.txt
[Permissions]$ ls -l
total 0
-rwxrwxr--. 1 salma salma 0 Aug 26 20:33 me.txt
[Permissions]$
```

Changing Ownerships

- For changing the ownership of a file/directory, you can use :
  chown username filename

- In case you want to change the user as well as group for a file or directory use the command :
  chown user:group filename

- In case you want to change group-owner only :
  chgrp group_name filename

Files Permission and Ownership Commands Recap

chmod

chown

chgrp

HANDS ON TIME

Hands on

1. Create a file named `report.txt` and a directory named `project`.
2. Check the permissions and owner of `report.txt` and `project`.
3. Make `report.txt` readable and writable only by the owner.
4. Change the group of `report.txt` to `developers`.
5. Change the owner of `report.txt` to `alice`.
6. Create a directory `shared` owned by you and group `developers` with full permissions for owner and group, and no permissions for others.
7. Inside `shared`, create a file `public.txt` readable by everyone, but writable only by the owner.

Hands on solution

- touch report.txt
- mkdir project
- ls -l report.txt
- ls -l project
- chmod 600 report.txt
- sudo chgrp developers report.txt
- sudo chown alice report.txt

- mkdir shared
- sudo chgrp developers shared
- chmod 770 shared
- cd shared
- touch public.txt
- chmod 644 public.txt
- mkdir teamwork
- sudo chgrp developers teamwork

- gpasswd

Commands Recap

whoami

useradd

groupadd

chown

id

userdel

groupdel

chgrp

su

usermod

groups

chmod

passwd

gpasswd

Part 2

Text Editors

Nano

vim

What is text editors ?

A **text editor** is a software application used to **create, view, and modify plain text files**.

➔ Unlike word processors (Word, LibreOffice), text editors **do not add formatting** (bold, italics).

➔ They are mainly used for **programming, configuration files, scripts, logs, and notes**.

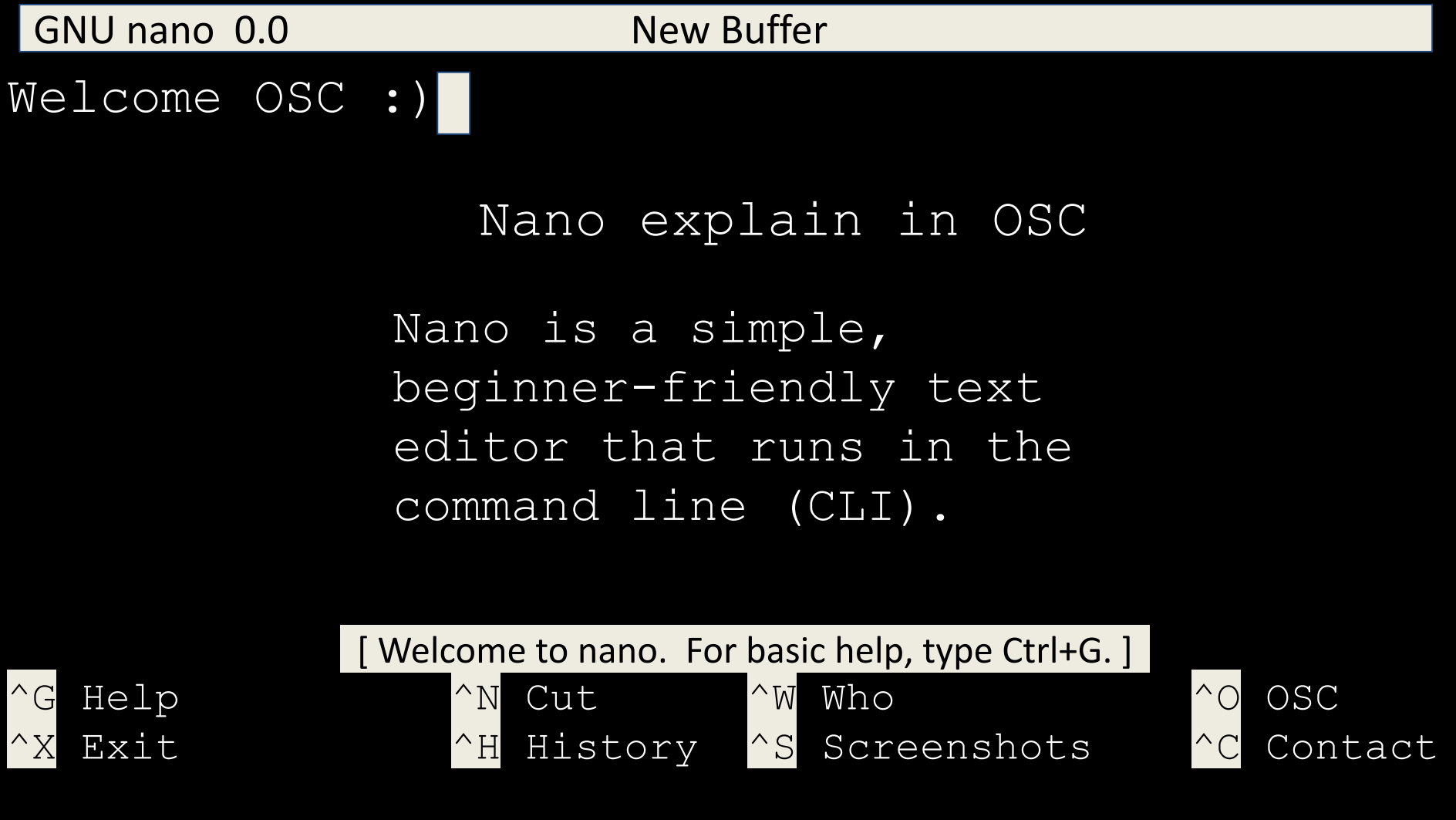
Types GUI and CLI

- **Command-line / terminal editors:** Vim, Nano, Emacs

- **GUI editors:** VS Codium, Sublime Text, gedit, Notepad++

CLI Editors (dark tunnel)

- ★ **Work anywhere (in terminals over ssh)**
- ★ **Fast and Lightweight (no GUI waiting!)**
- ★ **Scriptable and Customizable some times.**
- ★ **Universally Available**
- ★ **Can prevent accidental corruption of system files**


GNU nano 0.0
New Buffer
Welcome OSC :)
Nano explain in OSC
Nano is a simple, beginner-friendly text editor that runs in the command line (CLI).
[Welcome to nano. For basic help, type Ctrl+G.]
^G Help
^X Exit
^N Cut
^H History
^W Who
^S Screenshots
^O OSC
^C Contact

Nano is User-Friendly

- ★ **Simple UI with basic functionality**
- ★ **No modes → type and edit instantly**
- ★ **Shortcuts shown on screen**
- ★ **Installed on many systems**

Nano: File Handling

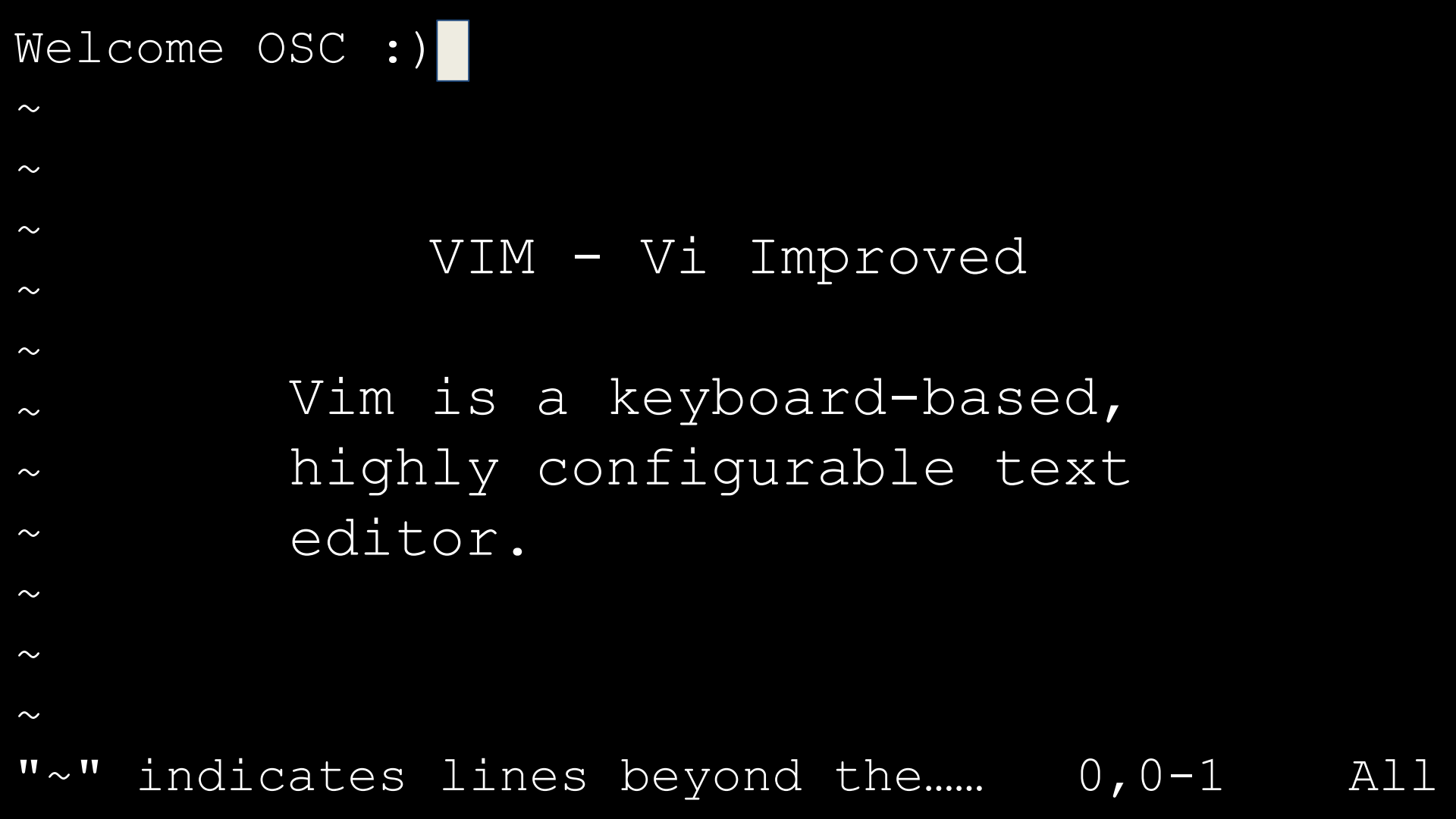
Action	Shortcut
Save current file	Ctrl + S
Save as / offer to write	Ctrl + O filename
Exit Nano	Ctrl + X
Insert another file	Ctrl + R

Nano: Basic Editing

Action	Shortcut
Cut current line	Ctrl + K
Copy current line	Alt + 6
Paste from cut buffer	Ctrl + U
Undo	Alt + U
Redo	Alt + E

Nano: Searching

Action	Shortcut
Forward search	Ctrl + W / Ctrl + F
Backward search	Ctrl + B
Replace (basic)	Alt + R


Welcome OSC :)
VIM - Vi Improved
Vim is a keyboard-based,
highly configurable text
editor.
"~" indicates lines beyond the……
0,0-1
All

VIM : Saving & Exiting

Action	Shortcut
Save	`:w`
Quit	`:q`
Save & quit	`:wq`
Quit without saving	`:q!`

VIM : Basic Editing

Action	Shortcut
Enter Insert mode	`i`
Enter Append mode	`a`
Delete (cut) current line	`d`
Copy (yank) current line	`y`
Paste	`p`
Undo	`u`
Redo	`Ctrl + r`

VIM : Visual Mode (Selecting Text)

Action	Shortcut
Enter visual mode	v
Select whole line	V
Copy (yank) selection	y
Cut (delete) selection	d
Paste	p

VIM : Searching

Action	Shortcut
Search forward	`/text` then Enter
Search backward	`?text` then Enter
Next match	`n`
Previous match	`N`

No task this time!

Practice your Vim skills using vimtutor to build muscle memory.

No deliverable for this task, but you are a lawful penguin who will do it regardless. :D

See you soon!

KAHOOT TIME

THANK YOU