Week3 day4

Practiced regex expressions in logfile-

grep -oE '[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}' logfile.txt

`[a-zA-Z0-9._%+-]+`

- `[ ]` → Defines a character set (matches any one of the specified characters).
- `a-zA-Z0-9` → Matches any uppercase (`A-Z`), lowercase (`a-z`), or digit (`0-9`).
- `._%+-` → Matches special characters commonly found in email usernames (dot `.` underscore `_`, percent `%`, plus `+`, and hyphen `-`).
- `+` → Matches one or more of the preceding characters (ensures at least one character in the username).
- **Example Matches:** `john.doe`, `user_name123`, `test-email+filter`
- `@`
- Matches the `@` symbol, which is mandatory in all email addresses.

`[a-zA-Z0-9.-]+`

- Matches the domain name part of the email (before the `.com`, `.org`, etc.).
- `a-zA-Z0-9` → Matches letters and digits.
- `.-` → Matches dots `.` and hyphens `-` (valid in domain names).
- `+` → Ensures at least one character is present.
- **Example Matches:** `gmail.com`, `my-company.co`, `university.edu`
- `\.`
- Matches a literal dot (`.`) before the top-level domain (TLD) like `.com`, `.net`, `.org`.
- The backslash `\` escapes the dot since `.` in regex normally matches any character.

`[a-zA-Z]{2,}`

- Matches the top-level domain (TLD).
- `{2,}` → Ensures at least **two or more** letters (like `.com`, `.edu`, `.uk`).
- **Example Matches:** `com`, `org`, `net`, `co.uk`

Installation of python3 and bazel-

Got lama working

Starting with python -

Looping through a list -

```
fruits = ["apple", "banana", "cherry"]
for fruit in fruits:
    print(fruit)
```

Using range -

```
for i in range(5):
# Loops from 0 to 4
print(i)
```

Python classes and objects -

What is a Class?

A **class** is a blueprint for creating objects. It defines properties (variables) and behaviors (methods) that the objects created from the class will have.

What is an Object?

An **object** is an instance of a class. It has its own **data** (attributes) and can perform **functions** (methods).

```
class Car:
    # Constructor (initializer)
    def __init__(self, brand, model, year):
        self.brand = brand  # Attribute
        self.model = model
        self.year = year

    # Method (function inside a class)
    def display_info(self):
        print(f"{self.year} {self.brand} {self.model}")

# Creating an object of the Car class
car1 = Car("Toyota", "Corolla", 2022)
car1.display_info()  # Output: 2022 Toyota Corolla
```