Scala

Scala stands for **Scalable Language.** It is a general-purpose, object-oriented language which has a few commonalities with the Java programming language. It provides support for functional programming. Many of Scala's design decisions were aimed to address the drawbacks of the Java programming language.

Why Choose Scala?

1. Combines Functional + Object-Oriented Programming

Scala is a **hybrid language** — you can write code in both:

- **Object-Oriented style** (like Java)
- **Functional style** (like Haskell or Python)

This gives developers **expressiveness** and **flexibility**.

```
// Functional style
val doubled = List(1, 2, 3).map(_ * 2)
```

2. Interoperable with Java

Scala runs on the **JVM** and can use any Java library or framework.

You can:

- Call Java code from Scala
- Use Java SDKs like JDBC, Kafka, Spark, Spring
- Mix Scala and Java in the same project

3. Concise and Expressive

Compared to Java, Scala code is **shorter** and **easier to read**.

```
// Java
List<Integer> nums = Arrays.asList(1, 2, 3);
for (int n : nums) {
  System.out.println(n * 2);
}

// Scala
List(1, 2, 3).foreach(n => println(n * 2))
```

4. Ideal for Big Data and Distributed Systems

- Apache **Spark** is written in Scala.
- Scala is the **first-class language** for Spark programming.
- Used in Big Data, ETL, ML pipelines.

---

5. Immutability and Safe Code

Scala emphasizes **immutability**, **type safety**, and **null-safety**, reducing bugs in production systems.

```
val name = "Dani"   // immutable
// name = "YK"      // ❌ compile error
```

---

6. Advanced Language Features

- Pattern matching
- Case classes
- Traits (like interfaces + partial implementation)
- Higher-order functions
- Type inference
- Implicits (advanced, powerful feature)

---

7. Growing Ecosystem and Community

Used by:

- Twitter
- LinkedIn
- Netflix
- AirBnB
- Uber

What is REPL?

REPL stands for **Read-Eval-Print Loop**. It is an interactive shell where Scala expressions are read, evaluated, and results are printed.

How REPL Works:

1. **Read**: Takes user input (Scala expression).
2. **Eval**: Evaluates the expression.
3. **Print**: Displays the result.
4. **Loop**: Repeats the above steps until exited.

Program

```
scala> val name = "Scala"
val name: String = Scala

scala> 2 + 3
val res0: Int = 5

scala> println("Hello, REPL!")
Hello, REPL!
```

Useful REPL Commands

Command	Description
:help	List all commands
:quit	Exit the REPL
:load file.scala	Load and run a Scala script file
:type expr	Show type of an expression
:reset	Clear all defined variables and imports

Code Samples for Practice:

```
// Declare and print variable
val x = 10
val y = 20
val sum = x + y
println("Sum is: " + sum)


// Define a simple function
def square(n: Int): Int = n * n
println(square(5))
```