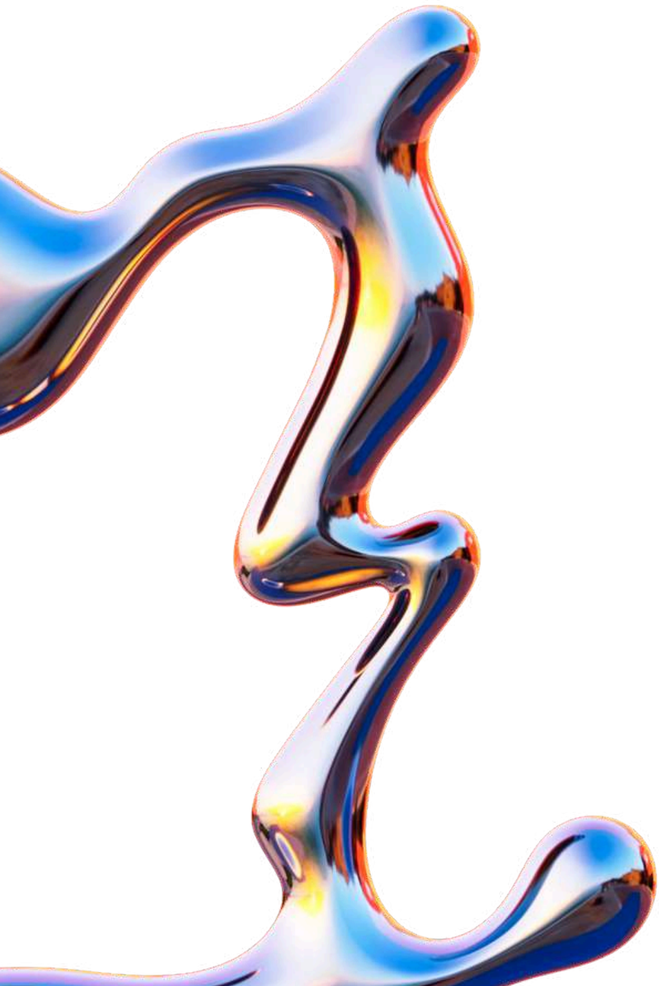
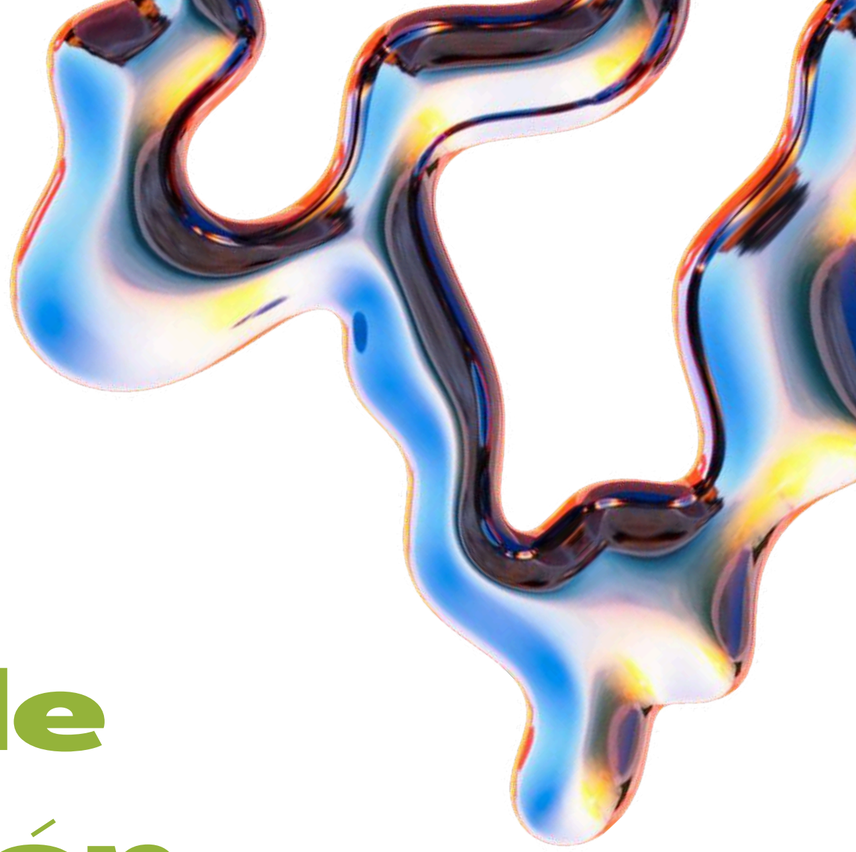
GRUPO 3



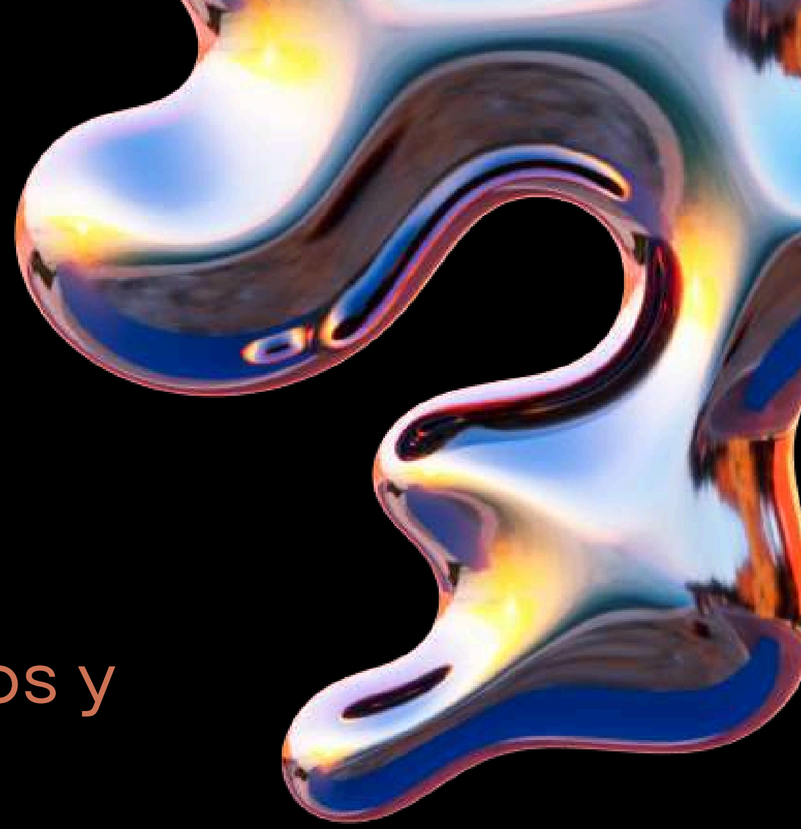


Teoría de Lenguajes de Programación

Sara Acevedo Maya
saacevedom@unal.edu.co
Universidad Nacional de Colombia

UNIVERSIDAD NACIONAL DE COLOMBIA

Contenido



01/ Clases , atributos y métodos
02/Objetos
03/ Constructores
04/Herencia

UNIVERSIDAD NACIONAL DE COLOMBIA

01 - Clases , atributos y métodos

UNIVERSIDAD
NACIONAL
DE COLOMBIA

Clases , atributos y métodos

Una **clase** en Scala es un molde para crear objetos. Define atributos y métodos que los objetos creados a partir de la clase compartirán.

```scala
// Definiendo la clase Persona con atributos y métodos
class Persona(val nombre: String, val edad: Int) {
  // Método para mostrar información
  def mostrarInformacion(): Unit = {
    println(s"Nombre: $nombre, Edad: $edad")
  }

  // Método para saludar
  def saludar(): Unit = {
    println(s"Hola, soy $nombre.")
  }
}
```

Los atributos son variables definidas dentro de una clase que representan las propiedades de los objetos.

Los métodos son funciones definidas dentro de una clase que describen los comportamientos de los objetos

UNIVERSIDAD NACIONAL DE COLOMBIA

02-Objetos

UNIVERSIDAD
NACIONAL
DE COLOMBIA

Objetos

Los objetos son instancias de una clase. Representan entidades concretas creadas a partir del molde de la clase.

```
object Main extends App {
  // Creando un objeto de la clase Persona
  val persona1 = new Persona("Ana", 23)

  // Llamando a los métodos del objeto
  persona1.mostrarInformacion()  // Output: Nombre: Ana, Edad: 23
  persona1.saludar()  // Output: Hola, soy Ana.
}
```

UNIVERSIDAD NACIONAL DE COLOMBIA

03-Constructores

UNIVERSIDAD
NACIONAL
DE COLOMBIA

Constructores

Los constructores son métodos especiales que se utilizan para *inicializar objetos. En Scala, el constructor primario se define junto a la clase, y se pueden definir constructores auxiliares con this.

```scala
class Persona(val nombre: String, val edad: Int) {
  // Constructor auxiliar
  def this(nombre: String) = this(nombre, 0)

  def mostrarInformacion(): Unit = {
    println(s"Nombre: $nombre, Edad: $edad")
  }
}

object Main extends App {
  val persona1 = new Persona("Ana", 23)
  val persona2 = new Persona("Carlos")

  persona1.mostrarInformacion()  // Output: Nombre: Ana, Edad: 23
  persona2.mostrarInformacion()  // Output: Nombre: Carlos, Edad: 0
}
```

*Cuando inicializas un objeto, estás creando una nueva instancia de una clase y asignándole valores a sus atributos a través de un constructor.

UNIVERSIDAD NACIONAL DE COLOMBIA

04-Herencia

UNIVERSIDAD
NACIONAL
DE COLOMBIA

Herencia

La herencia es un mecanismo que permite que una clase (subclase) herede propiedades y métodos de otra clase (superclase).

```scala
// Superclase Persona
class Persona(val nombre: String, val edad: Int) {
  def mostrarInformacion(): Unit = {
    println(s"Nombre: $nombre, Edad: $edad")
  }
}

// Subclase Estudiante que hereda de Persona
class Estudiante(nombre: String, edad: Int, val carrera: String) extends Persona(nombre, edad) {
  def estudiar(): Unit = {
    println(s"$nombre está estudiando $carrera.")
  }

  override def mostrarInformacion(): Unit = {
    println(s"Nombre: $nombre, Edad: $edad, Carrera: $carrera")
  }
}

object Main extends App {
  val estudiante1 = new Estudiante("Carlos", 21, "Ingeniería")

  estudiante1.mostrarInformacion()  // Output: Nombre: Carlos, Edad: 21, Carrera: Ingeniería
  estudiante1.estudiar()  // Output: Carlos está estudiando Ingeniería.
}
```

UNIVERSIDAD NACIONAL DE COLOMBIA

Herencia

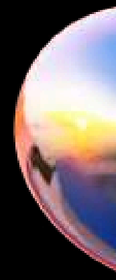
```scala
// Superclase Persona
class Persona(val nombre: String, val edad: Int) {
  def mostrarInformacion(): Unit = {
    println(s"Nombre: $nombre, Edad: $edad")
  }
}

// Subclase Estudiante que hereda de Persona
class Estudiante(nombre: String, edad: Int, val carrera: String) extends Persona(nombre, edad) {
  def estudiar(): Unit = {
    println(s"$nombre está estudiando $carrera.")
  }

  override def mostrarInformacion(): Unit = {
    println(s"Nombre: $nombre, Edad: $edad, Carrera: $carrera")
  }
}

object Main extends App {
  val estudiante1 = new Estudiante("Carlos", 21, "Ingeniería")

  estudiante1.mostrarInformacion()  // Output: Nombre: Carlos, Edad: 21, Carrera: Ingeniería
  estudiante1.estudiar()  // Output: Carlos está estudiando Ingeniería.
}
```

UNIVERSIDAD NACIONAL DE COLOMBIA

Conceptos importantes

- **val** vs **var**
- Hacerlo en una sóla pag vs separadas
- **Override:** Se utiliza para sobrescribir un método o variable en una subclase que ya está definido en la superclase, haciendo explícito el reemplazo.
- **Unit:** Representa la ausencia de un valor significativo y es similar a void en otros lenguajes, utilizado como tipo de retorno para métodos que no devuelven un valor útil.

UNIVERSIDAD NACIONAL DE COLOMBIA



```
// Definiendo la superclase Persona
class Persona(val nombre: String, val edad: Int) {
  // Constructor auxiliar
  def this(nombre: String) = this(nombre, 0)

  // Método para mostrar información
  def mostrarInformacion(): Unit = {
    println(s"Nombre: $nombre, Edad: $edad")
  }

  // Método para saludar
  def saludar(): Unit = {
    println(s"Hola, soy $nombre.")
  }
}

// Definiendo la subclase Estudiante que hereda de Persona
class Estudiante(nombre: String, edad: Int, val carrera: String) extends Persona(nombre, edad) {
  // Método específico de Estudiante
  def estudiar(): Unit = {
    println(s"$nombre está estudiando $carrera.")
  }

  // Sobrescribiendo el método mostrarInformacion
  override def mostrarInformacion(): Unit = {
    println(s"Nombre: $nombre, Edad: $edad, Carrera: $carrera")
  }
}

object Main extends App {
  // Creando objetos de la clase Persona usando ambos constructores
  val persona1 = new Persona("Ana", 23)
  val persona2 = new Persona("Carlos")

  // Creando un objeto de la subclase Estudiante
  val estudiante1 = new Estudiante("Carlos", 21, "Ingeniería")

  // Llamando a los métodos para mostrar información
  persona1.mostrarInformacion()  // Output: Nombre: Ana, Edad: 23
  persona2.mostrarInformacion()  // Output: Nombre: Carlos, Edad: 0
  estudiante1.mostrarInformacion()  // Output: Nombre: Carlos, Edad: 21, Carrera: Ingeniería

  // Llamando al método específico de Estudiante
  estudiante1.estudiar()  // Output: Carlos está estudiando Ingeniería.
  estudiante1.saludar()   // Output: Hola, soy Carlos.
}
```

UNIVERSIDAD NACIONAL DE COLOMBIA