Ex No : Date :	**SCALA OBJECT ORIENTATION, TRAITS, ABSTRACT CLASSES**

AIM:

The aim of this Scala program is to demonstrate object-oriented programming concepts.

ALGORITHM:

1. Define an abstract class Animal with:
   - Primary constructor (name)
   - Auxiliary constructor (default name)
   - Abstract method sound()
   - Concrete method eat()
2. Define traits Mammal and Pet with methods nurse() and play(), respectively.
3. Create concrete classes Dog and Cat extending Animal and mixing in Mammal and Pet traits:
   - Override sound() method
   - Override eat() method (optional)
   - Define auxiliary constructors (default breed/color)
4. Create companion objects Dog and Cat with apply methods to create instances.
5. Create a singleton object AnimalKingdom with a main method to:
   - Create instances of Dog and Cat using primary and auxiliary constructors
   - Call methods on these instances to demonstrate their behaviors

PROGRAM:

```
// Abstract Class: Animal
abstract class Animal(val name: String) {
  def sound(): Unit
  // Primary constructor

  // Auxiliary constructor
  def this() = this("Unknown")

  def eat(): Unit = println(s"$name is eating...")
}

// Trait: Mammal
```

```
trait Mammal {
  def nurse(): Unit = println("Nursing...")
}

// Trait: Pet
trait Pet {
  def play(): Unit = println("Playing...")
}

// Concrete Class: Dog
class Dog(val breed: String) extends Animal(breed) with Mammal with Pet {
  override def sound(): Unit = println("Woof!")

  override def eat(): Unit = println(s"$breed is eating...")

  // Auxiliary constructor
  def this() = this("Golden Retriever")
}

// Concrete Class: Cat
class Cat(val color: String) extends Animal(color) with Mammal with Pet {
  override def sound(): Unit = println("Meow!")

  override def eat(): Unit = println(s"$color is eating...")

  // Auxiliary constructor
  def this() = this("Black")
}

// Companion Object: Dog
object Dog {
  def apply(): Dog = new Dog()
}

// Companion Object: Cat
object Cat {
  def apply(): Cat = new Cat()
}

// Singleton Object: AnimalKingdom
object AnimalKingdom {
  def main(args: Array[String]): Unit = {
```

```
    val dog = new Dog("Poodle")
    dog.sound()
    dog.eat()
    dog.nurse()
    dog.play()

    val defaultDog = Dog()
    defaultDog.sound()
    defaultDog.eat()
    defaultDog.nurse()
    defaultDog.play()

    val cat = new Cat("White")
    cat.sound()
    cat.eat()

    cat.nurse()

    cat.play()

    val defaultCat = Cat()
    defaultCat.sound()
    defaultCat.eat()
    defaultCat.nurse()
    defaultCat.play()
  }
}
```

PREPARATION	10		
OBSERVATION	10		
OUTPUT	10		
VIVA	10		
RECORD	10		

RESULT:

Object Oriented programming concepts of Scala is studied and executed successfully.