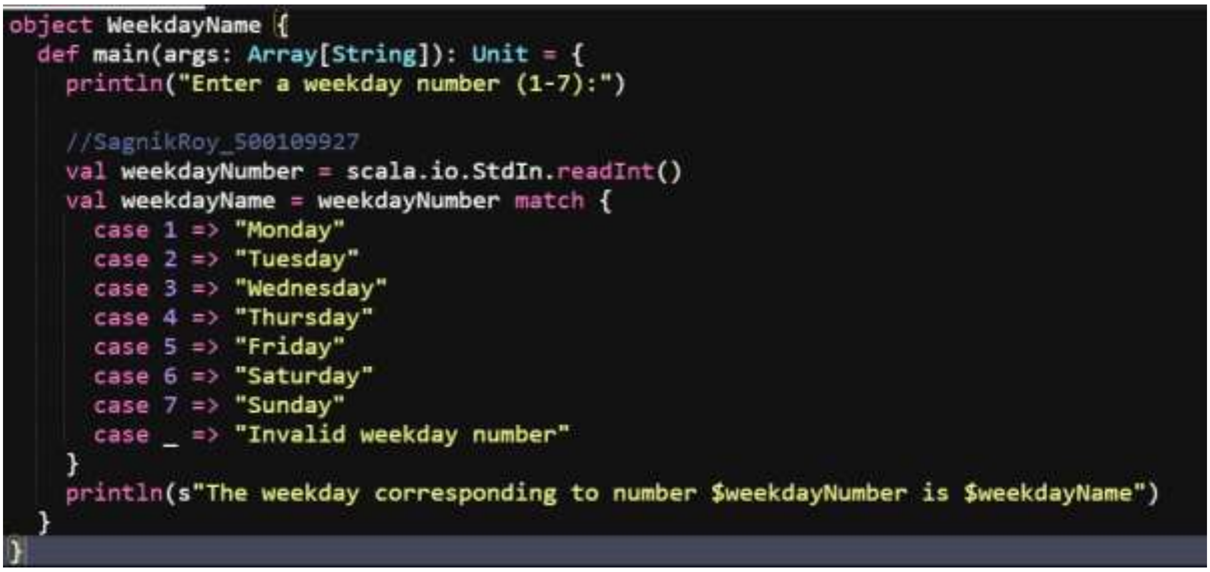
LAB 4

Q1. Scala program to read a weekday number and print weekday name using match case.

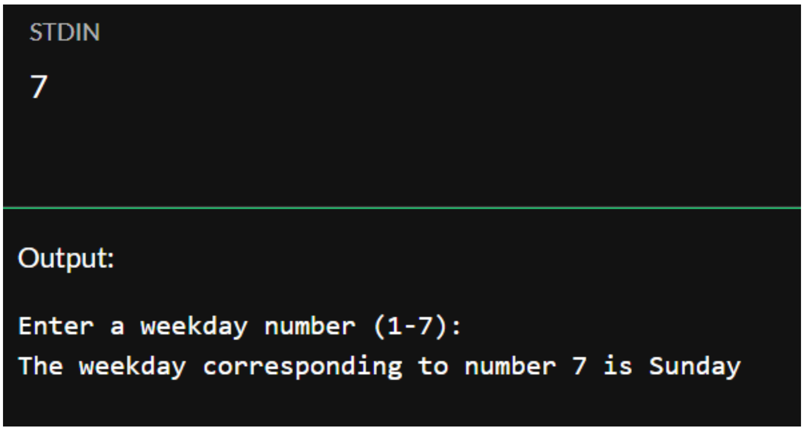
Code:



```
object WeekdayName {
  def main(args: Array[String]): Unit = {
    println("Enter a weekday number (1-7):")

    //SagnikRoy_500109927
    val weekdayNumber = scala.io.StdIn.readInt()
    val weekdayName = weekdayNumber match {
      case 1 => "Monday"
      case 2 => "Tuesday"
      case 3 => "Wednesday"
      case 4 => "Thursday"
      case 5 => "Friday"
      case 6 => "Saturday"
      case 7 => "Sunday"
      case _ => "Invalid weekday number"
    }
    println(s"The weekday corresponding to number $weekdayNumber is $weekdayName")
  }
}
```

Output:



STDIN

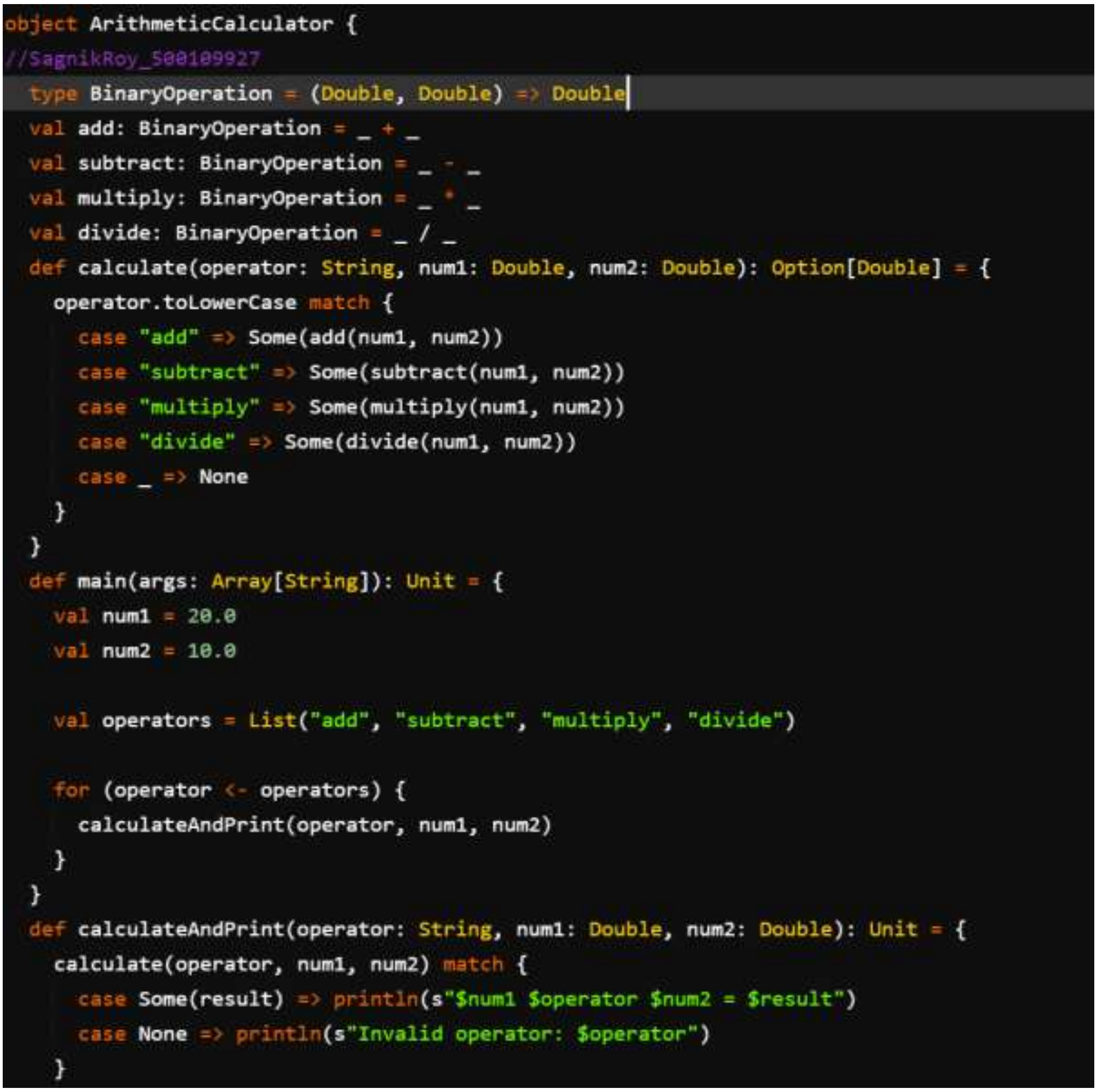
```
7
```

Output:

```
Enter a weekday number (1-7):
The weekday corresponding to number 7 is Sunday
```

Q2. Scala program to implement an arithmetic calculator using higher-order functions.

Code:

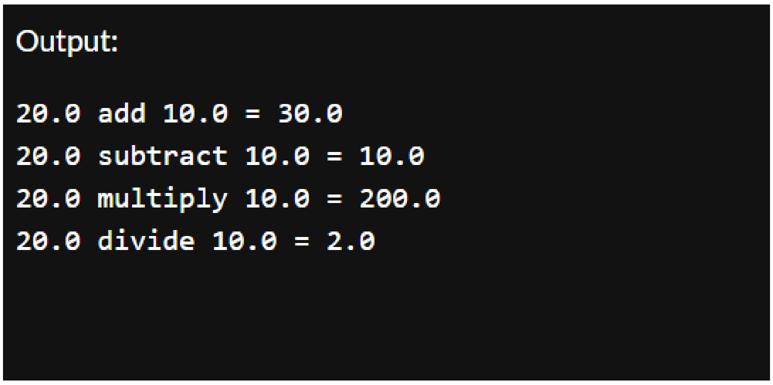


```scala
object ArithmeticCalculator {
//SagnikRoy_500109927
  type BinaryOperation = (Double, Double) => Double
  val add: BinaryOperation = _ + _
  val subtract: BinaryOperation = _ - _
  val multiply: BinaryOperation = _ * _
  val divide: BinaryOperation = _ / _
  def calculate(operator: String, num1: Double, num2: Double): Option[Double] = {
    operator.toLowerCase match {
      case "add" => Some(add(num1, num2))
      case "subtract" => Some(subtract(num1, num2))
      case "multiply" => Some(multiply(num1, num2))
      case "divide" => Some(divide(num1, num2))
      case _ => None
    }
  }
  def main(args: Array[String]): Unit = {
    val num1 = 20.0
    val num2 = 10.0

    val operators = List("add", "subtract", "multiply", "divide")

    for (operator <- operators) {
      calculateAndPrint(operator, num1, num2)
    }
  }
  def calculateAndPrint(operator: String, num1: Double, num2: Double): Unit = {
    calculate(operator, num1, num2) match {
      case Some(result) => println(s"$num1 $operator $num2 = $result")
      case None => println(s"Invalid operator: $operator")
    }
```

Output:

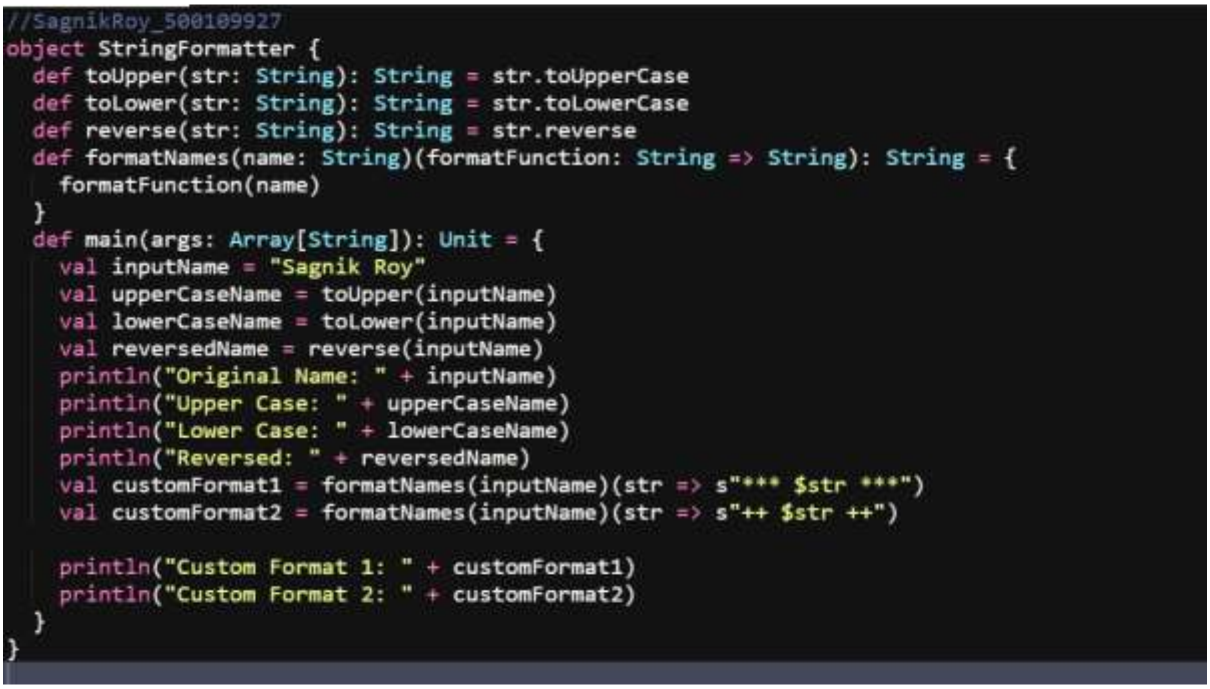


```
Output:

20.0 add 10.0 = 30.0
20.0 subtract 10.0 = 10.0
20.0 multiply 10.0 = 200.0
20.0 divide 10.0 = 2.0
```

Q3. Write a Scala program which defines a methods named "toUpper", "toLower", and "reverse", which accepts a String as input parameter and formats it. Define another method named "formatNames" which also has an input String called "name". This method however has a parameter group which accepts a functions with an input of type String and also outputs a String. This particular function will be used to apply the given format to the "name" input.

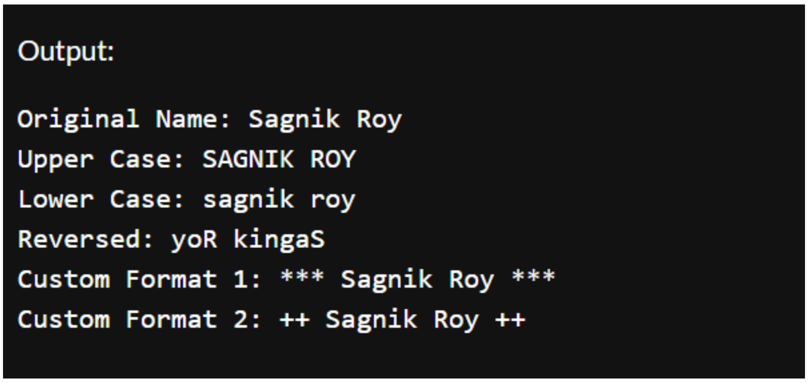
Code:



```
//SagnikRoy_500109927
object StringFormatter {
  def toUpper(str: String): String = str.toUpperCase
  def toLower(str: String): String = str.toLowerCase
  def reverse(str: String): String = str.reverse
  def formatNames(name: String)(formatFunction: String => String): String = {
    formatFunction(name)
  }
  def main(args: Array[String]): Unit = {
    val inputName = "Sagnik Roy"
    val upperCaseName = toUpper(inputName)
    val lowerCaseName = toLower(inputName)
    val reversedName = reverse(inputName)
    println("Original Name: " + inputName)
    println("Upper Case: " + upperCaseName)
    println("Lower Case: " + lowerCaseName)
    println("Reversed: " + reversedName)
    val customFormat1 = formatNames(inputName)(str => s"*** $str ***")
    val customFormat2 = formatNames(inputName)(str => s"++ $str ++")

    println("Custom Format 1: " + customFormat1)
    println("Custom Format 2: " + customFormat2)
  }
}
```

Output:



Output:

```
Original Name: Sagnik Roy
Upper Case: SAGNIK ROY
Lower Case: sagnik roy
Reversed: yoR kingaS
Custom Format 1: *** Sagnik Roy ***
Custom Format 2: ++ Sagnik Roy ++
```