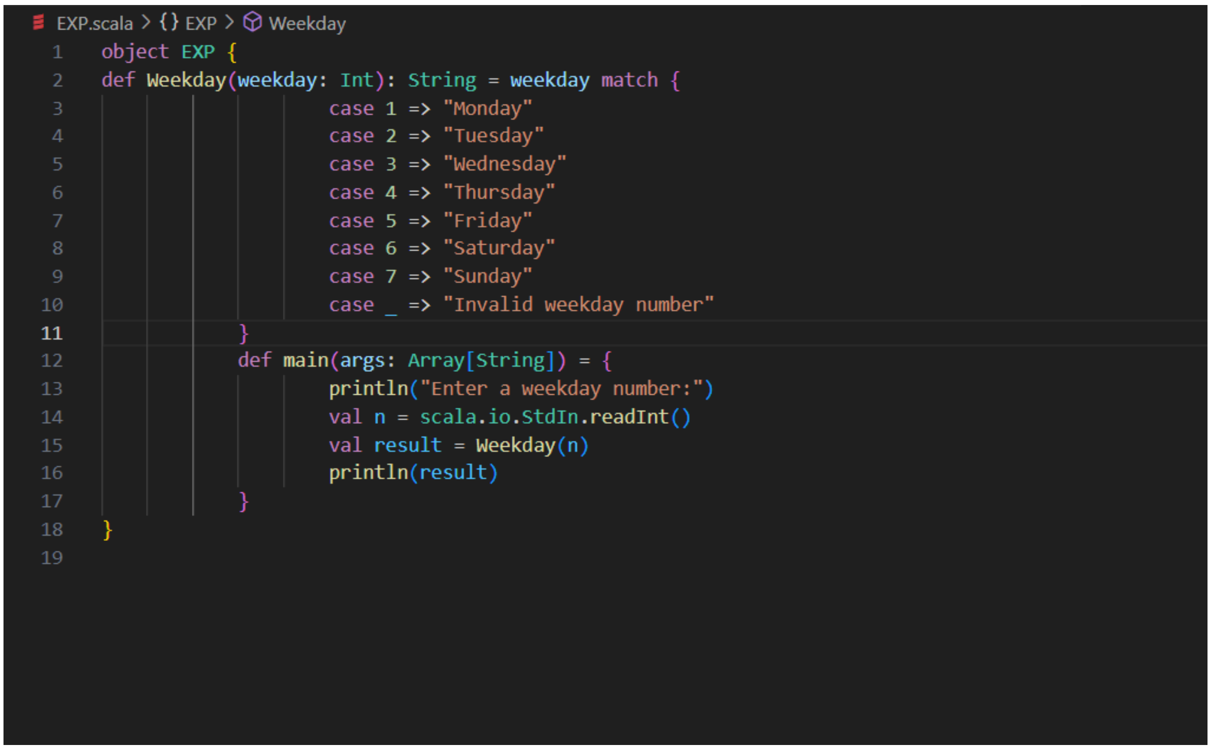
LAB 4

Advanced Functional Thinking

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

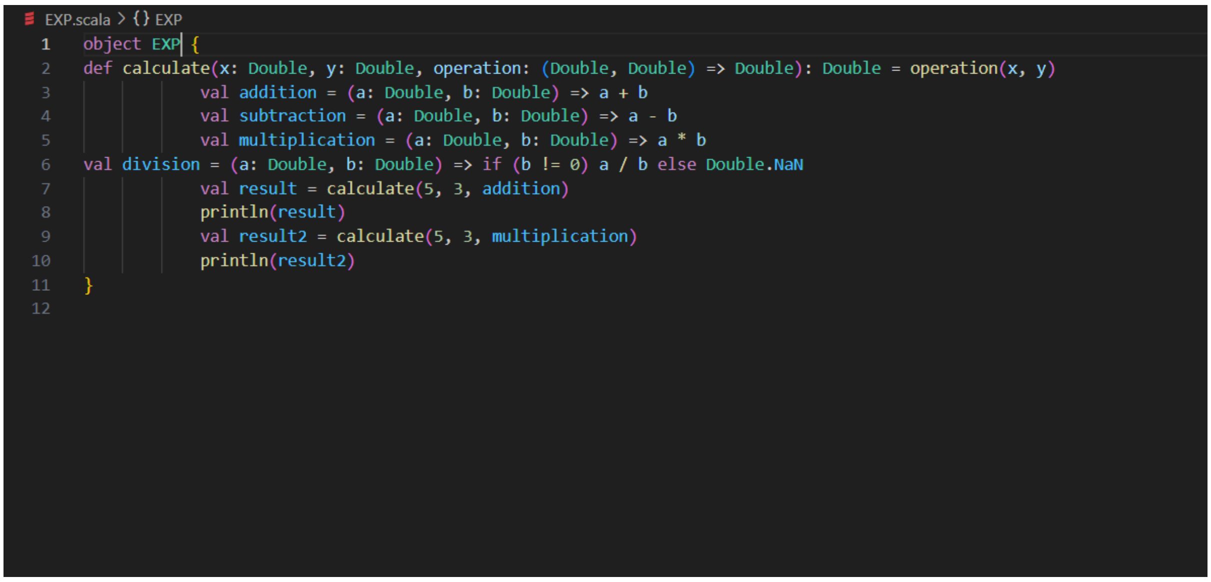
ANS-



```
EXP.scala > {} EXP > Weekday
object EXP {
def Weekday(weekday: Int): String = weekday 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"
            }
            def main(args: Array[String]) = {
                    println("Enter a weekday number:")
                    val n = scala.io.StdIn.readInt()
                    val result = Weekday(n)
                    println(result)
            }
}
```

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

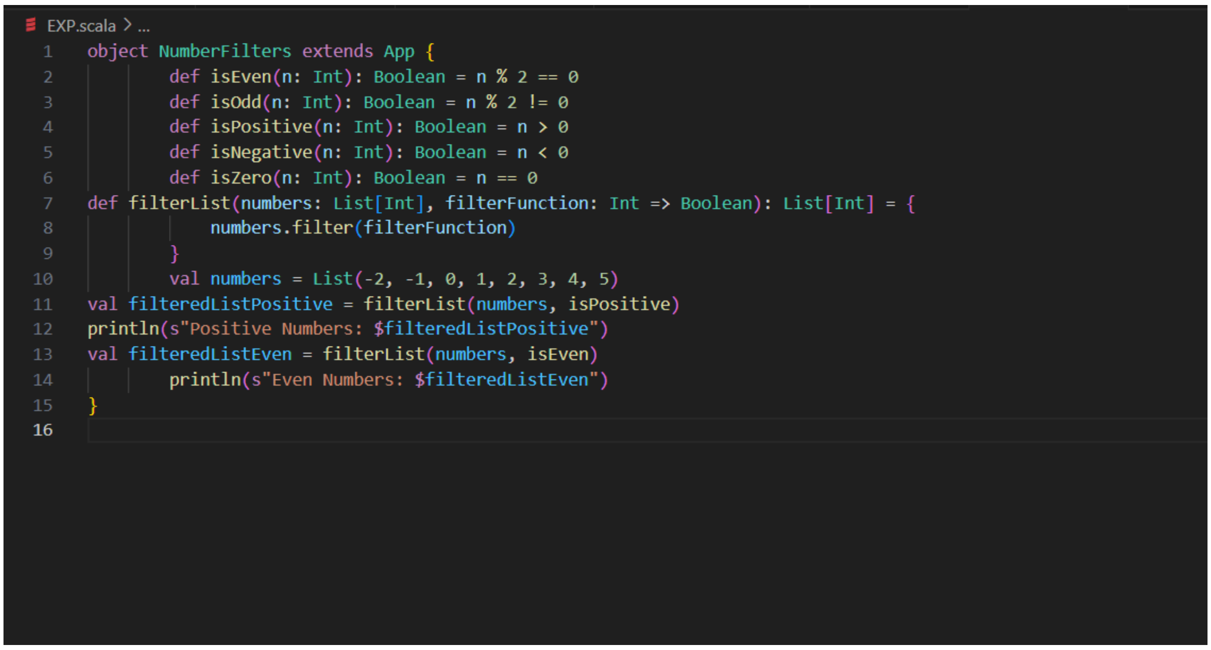
ANS-



```
EXP.scala > {} EXP
object EXP {
def calculate(x: Double, y: Double, operation: (Double, Double) => Double): Double = operation(x, y)
            val addition = (a: Double, b: Double) => a + b
            val subtraction = (a: Double, b: Double) => a - b
            val multiplication = (a: Double, b: Double) => a * b
val division = (a: Double, b: Double) => if (b != 0) a / b else Double.NaN
            val result = calculate(5, 3, addition)
            println(result)
            val result2 = calculate(5, 3, multiplication)
            println(result2)
}
```

Q3. Create functions isEven, isOdd, isPositive, isNegative, isZero. Create a higher order function filterList that takes a list of numbers and any one filter function created above.

ANS-

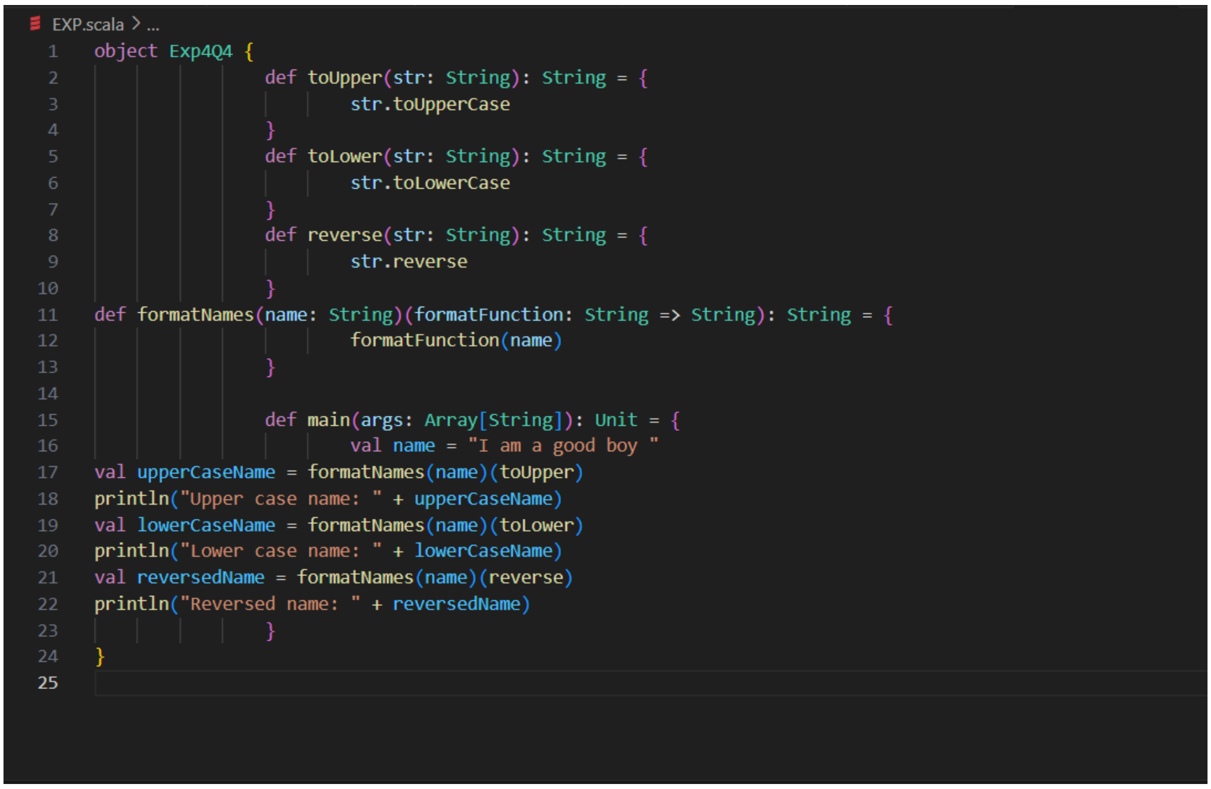


EXP.scala > ...

```scala
object NumberFilters extends App {
        def isEven(n: Int): Boolean = n % 2 == 0
        def isOdd(n: Int): Boolean = n % 2 != 0
        def isPositive(n: Int): Boolean = n > 0
        def isNegative(n: Int): Boolean = n < 0
        def isZero(n: Int): Boolean = n == 0
def filterList(numbers: List[Int], filterFunction: Int => Boolean): List[Int] = {
            numbers.filter(filterFunction)
        }
        val numbers = List(-2, -1, 0, 1, 2, 3, 4, 5)
val filteredListPositive = filterList(numbers, isPositive)
println(s"Positive Numbers: $filteredListPositive")
val filteredListEven = filterList(numbers, isEven)
        println(s"Even Numbers: $filteredListEven")
}
```

Q4. 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.

ANS-



```scala
object Exp4Q4 {
                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 name = "I am a good boy "
val upperCaseName = formatNames(name)(toUpper)
println("Upper case name: " + upperCaseName)
val lowerCaseName = formatNames(name)(toLower)
println("Lower case name: " + lowerCaseName)
val reversedName = formatNames(name)(reverse)
println("Reversed name: " + reversedName)
                }
}
```