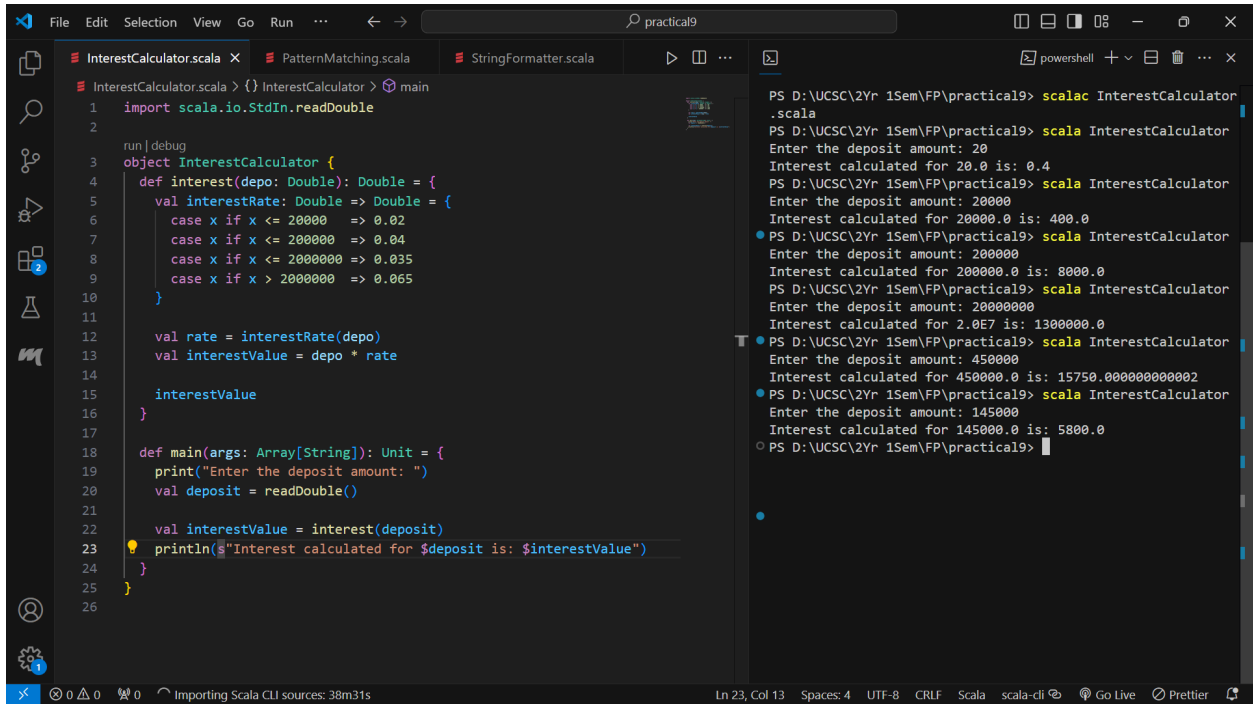


Practical 09 - 22001956

Question 01



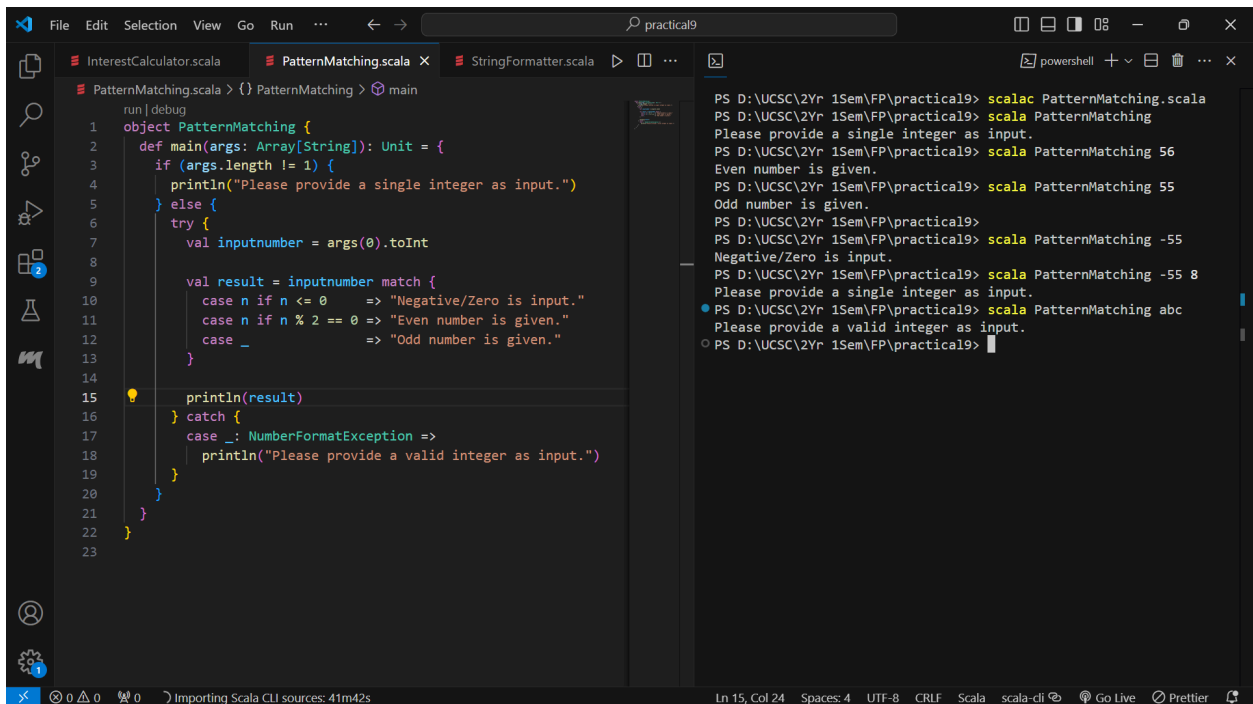
The screenshot shows an IDE with the file `InterestCalculator.scala` open. The code defines a function `interest` that calculates interest based on a deposit amount and a rate. The `main` function prompts the user for a deposit amount and prints the calculated interest.

```
1 import scala.io.StdIn.readDouble
2
3 run | debug
4 object InterestCalculator {
5   def interest(depo: Double): Double = {
6     val interestRate: Double => Double = {
7       case x if x <= 20000 => 0.02
8       case x if x <= 200000 => 0.04
9       case x if x <= 2000000 => 0.035
10      case x if x > 2000000 => 0.065
11    }
12
13    val rate = interestRate(depo)
14    val interestValue = depo * rate
15
16    interestValue
17  }
18
19  def main(args: Array[String]): Unit = {
20    print("Enter the deposit amount: ")
21    val deposit = readDouble()
22
23    val interestValue = interest(deposit)
24    println(s"Interest calculated for $deposit is: $interestValue")
25  }
26 }
```

The PowerShell terminal on the right shows the execution of the program. It prompts for a deposit amount and calculates the interest for several different values.

```
PS D:\UCSC\2Yr 1Sem\FP\practical9> scalac InterestCalculator.scala
PS D:\UCSC\2Yr 1Sem\FP\practical9> scala InterestCalculator
Enter the deposit amount: 20
Interest calculated for 20.0 is: 0.4
PS D:\UCSC\2Yr 1Sem\FP\practical9> scala InterestCalculator
Enter the deposit amount: 20000
Interest calculated for 20000.0 is: 400.0
PS D:\UCSC\2Yr 1Sem\FP\practical9> scala InterestCalculator
Enter the deposit amount: 200000
Interest calculated for 200000.0 is: 8000.0
PS D:\UCSC\2Yr 1Sem\FP\practical9> scala InterestCalculator
Enter the deposit amount: 2000000
Interest calculated for 2.0E7 is: 1300000.0
PS D:\UCSC\2Yr 1Sem\FP\practical9> scala InterestCalculator
Enter the deposit amount: 450000
Interest calculated for 450000.0 is: 15750.000000000002
PS D:\UCSC\2Yr 1Sem\FP\practical9> scala InterestCalculator
Enter the deposit amount: 145000
Interest calculated for 145000.0 is: 5800.0
PS D:\UCSC\2Yr 1Sem\FP\practical9>
```

Question 02



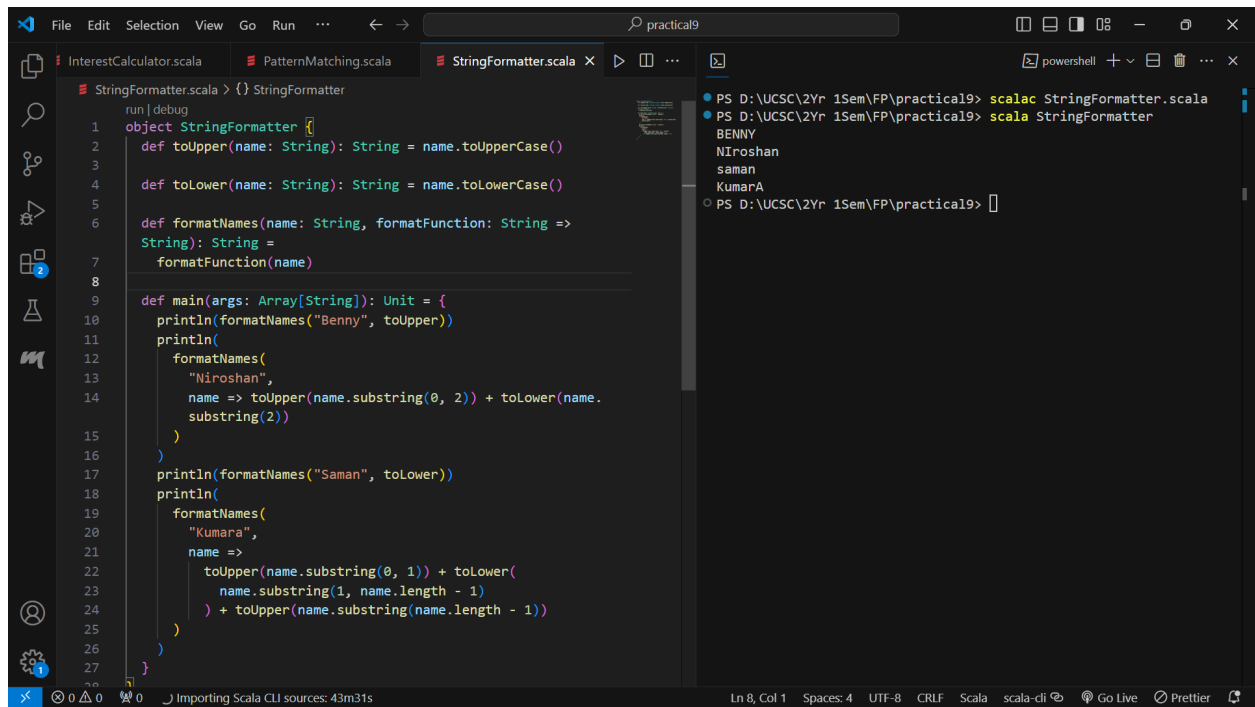
The screenshot shows an IDE with the file `PatternMatching.scala` open. The code defines a function `main` that prompts the user for a single integer and checks if it is even, odd, or negative/zero. It also handles exceptions for invalid input.

```
1 run | debug
2 object PatternMatching {
3   def main(args: Array[String]): Unit = {
4     if (args.length != 1) {
5       println("Please provide a single integer as input.")
6     } else {
7       try {
8         val inputnumber = args(0).toInt
9
10        val result = inputnumber match {
11          case n if n <= 0 => "Negative/Zero is input."
12          case n if n % 2 == 0 => "Even number is given."
13          case _ => "Odd number is given."
14        }
15
16        println(result)
17      } catch {
18        case _: NumberFormatException =>
19          println("Please provide a valid integer as input.")
20      }
21    }
22  }
23 }
```

The PowerShell terminal on the right shows the execution of the program. It prompts for a single integer and checks if it is even, odd, or negative/zero. It also handles exceptions for invalid input.

```
PS D:\UCSC\2Yr 1Sem\FP\practical9> scalac PatternMatching.scala
PS D:\UCSC\2Yr 1Sem\FP\practical9> scala PatternMatching
Please provide a single integer as input.
PS D:\UCSC\2Yr 1Sem\FP\practical9> scala PatternMatching 56
Even number is given.
PS D:\UCSC\2Yr 1Sem\FP\practical9> scala PatternMatching 55
Odd number is given.
PS D:\UCSC\2Yr 1Sem\FP\practical9> scala PatternMatching -55
Negative/Zero is input.
PS D:\UCSC\2Yr 1Sem\FP\practical9> scala PatternMatching -55 8
Please provide a single integer as input.
PS D:\UCSC\2Yr 1Sem\FP\practical9> scala PatternMatching abc
Please provide a valid integer as input.
PS D:\UCSC\2Yr 1Sem\FP\practical9>
```

Question 03



The screenshot shows an IDE with three tabs: InterestCalculator.scala, PatternMatching.scala, and StringFormatter.scala. The StringFormatter.scala tab is active, displaying the following Scala code:

```
1  run | debug
2  object StringFormatter {
3      def toUpper(name: String): String = name.toUpperCase()
4
5      def toLower(name: String): String = name.toLowerCase()
6
7      def formatNames(name: String, formatFunction: String =>
8      String): String =
9          formatFunction(name)
10
11      def main(args: Array[String]): Unit = {
12          println(formatNames("Benny", toUpper))
13          println(
14              formatNames(
15                  "Niroshan",
16                  name => toUpper(name.substring(0, 2)) + toLower(name.
17                      substring(2))
18              )
19          )
20          println(formatNames("Saman", toLower))
21          println(
22              formatNames(
23                  "Kumara",
24                  name =>
25                      toUpper(name.substring(0, 1)) + toLower(
26                          name.substring(1, name.length - 1))
27                      + toUpper(name.substring(name.length - 1))
28              )
29          )
30      }
31  }
```

The terminal on the right shows the execution of the code:

```
PS D:\UCSC\2Yr 1Sem\FP\practical9> scalac StringFormatter.scala
PS D:\UCSC\2Yr 1Sem\FP\practical9> scala StringFormatter
BENNY
NIroshan
saman
Kumara
PS D:\UCSC\2Yr 1Sem\FP\practical9>
```

The status bar at the bottom indicates the current position is Line 8, Column 1, with 4 spaces, UTF-8 encoding, CRLF line endings, and the Scala compiler. It also shows the Scala CLI version and the Go Live button.