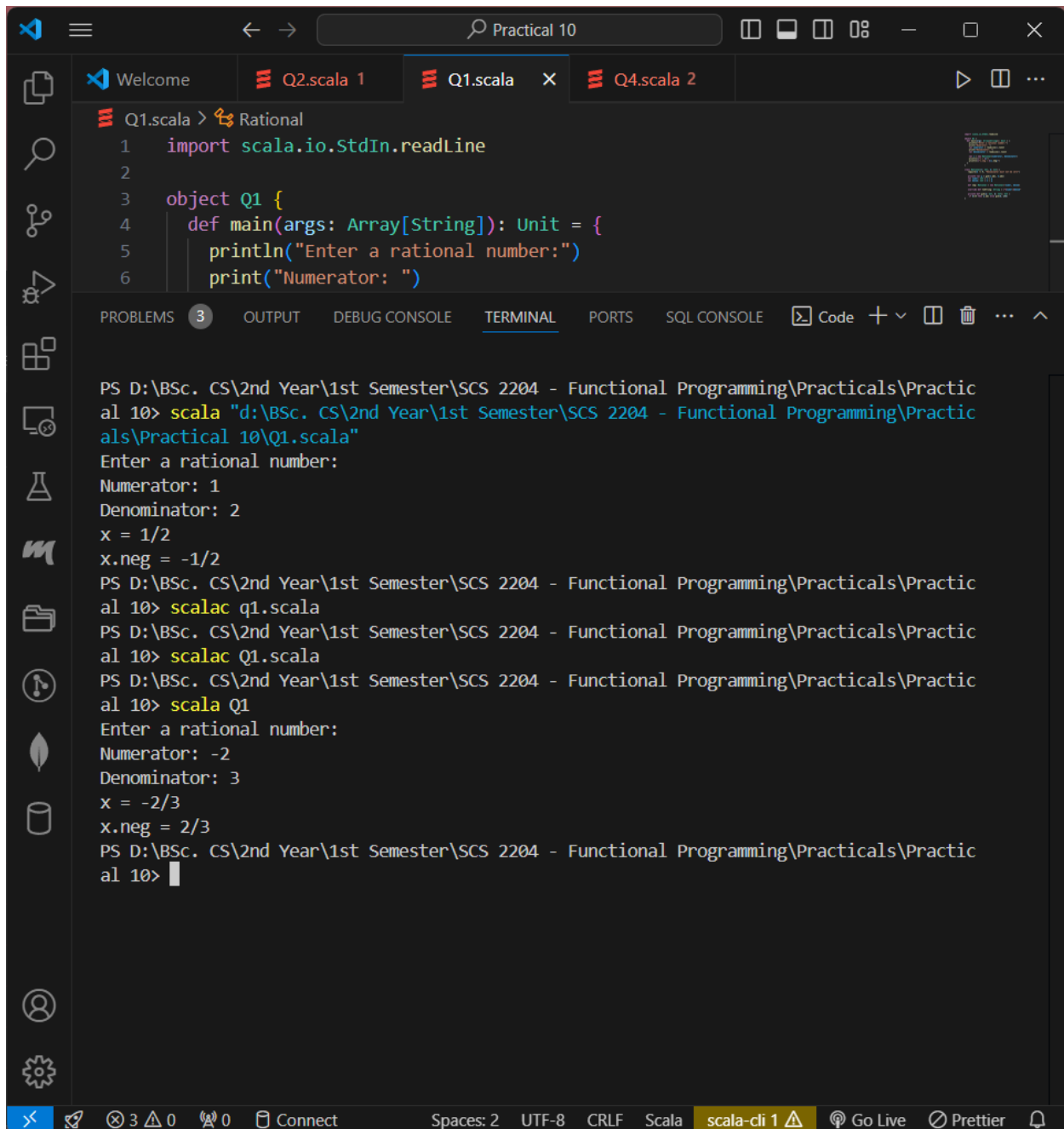


Functional Programming - Practical 10

2022/CS/109

Question 01:



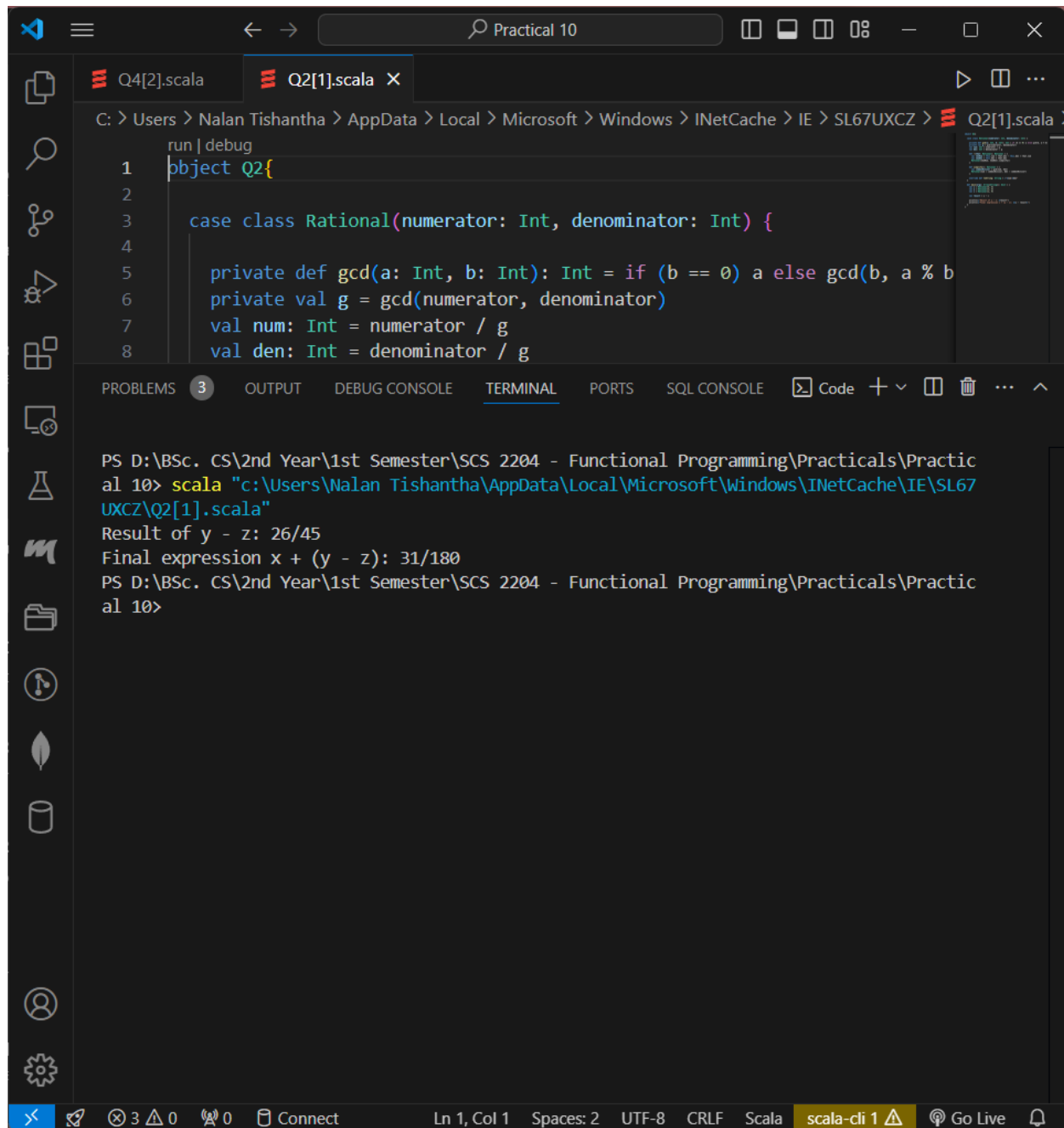
The screenshot shows a VS Code editor with a Scala file named `Q1.scala` open. The code defines a `Rational` object with a `main` function that prompts the user for a rational number and prints its components and negation.

```
Q1.scala > Rational
1 import scala.io.StdIn.readLine
2
3 object Q1 {
4   def main(args: Array[String]): Unit = {
5     println("Enter a rational number:")
6     print("Numerator: ")
```

The terminal output shows the program being executed twice. The first run uses input 1/2, resulting in `x = 1/2` and `x.neg = -1/2`. The second run uses input -2/3, resulting in `x = -2/3` and `x.neg = 2/3`.

```
PS D:\BSc. CS\2nd Year\1st Semester\SCS 2204 - Functional Programming\Practicals\Practical 10> scala "d:\BSc. CS\2nd Year\1st Semester\SCS 2204 - Functional Programming\Practicals\Practical 10\Q1.scala"
Enter a rational number:
Numerator: 1
Denominator: 2
x = 1/2
x.neg = -1/2
PS D:\BSc. CS\2nd Year\1st Semester\SCS 2204 - Functional Programming\Practicals\Practical 10> scalac q1.scala
PS D:\BSc. CS\2nd Year\1st Semester\SCS 2204 - Functional Programming\Practicals\Practical 10> scalac Q1.scala
PS D:\BSc. CS\2nd Year\1st Semester\SCS 2204 - Functional Programming\Practicals\Practical 10> scala Q1
Enter a rational number:
Numerator: -2
Denominator: 3
x = -2/3
x.neg = 2/3
PS D:\BSc. CS\2nd Year\1st Semester\SCS 2204 - Functional Programming\Practicals\Practical 10>
```

Question 02:



The image shows a Visual Studio Code editor window with two tabs: Q4[2].scala and Q2[1].scala. The active tab is Q2[1].scala, which contains the following Scala code:

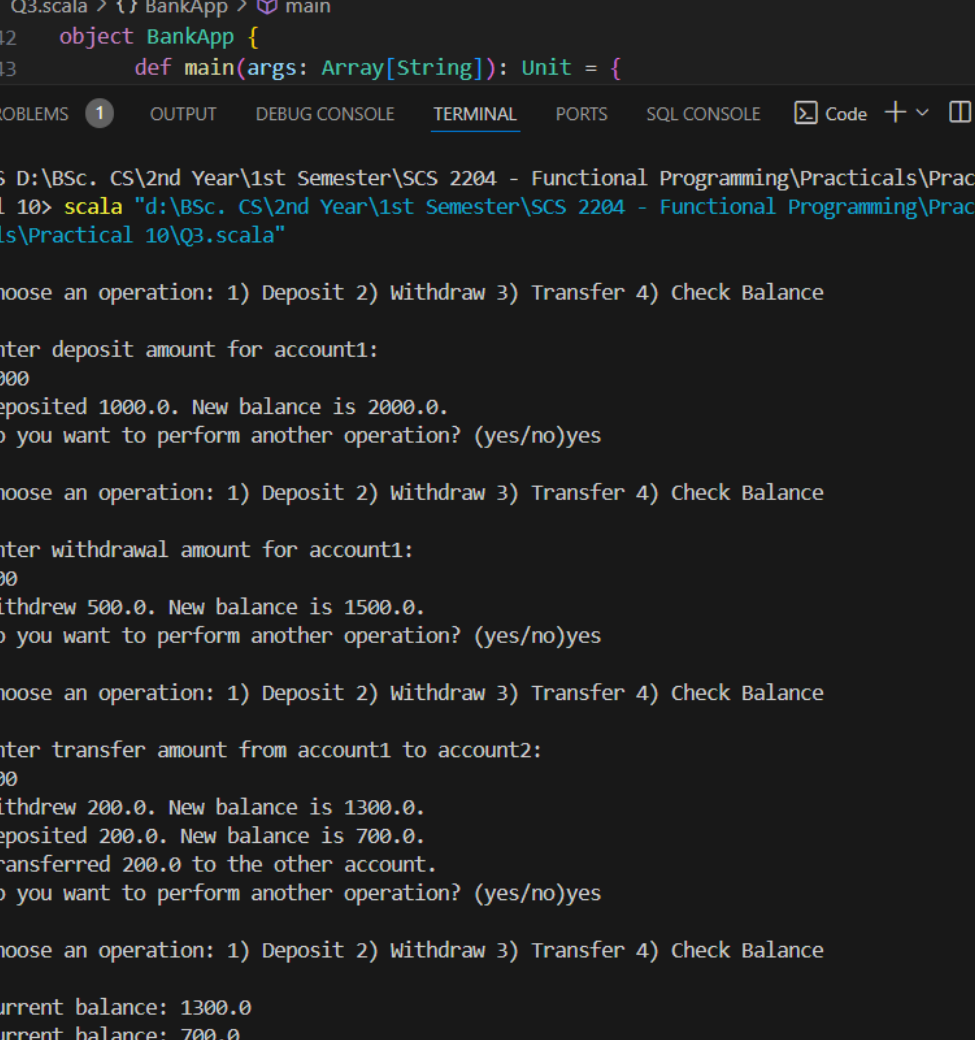
```
1  run | debug
2  object Q2{
3
4      case class Rational(numerator: Int, denominator: Int) {
5
6          private def gcd(a: Int, b: Int): Int = if (b == 0) a else gcd(b, a % b)
7          private val g = gcd(numerator, denominator)
8          val num: Int = numerator / g
9          val den: Int = denominator / g
10     }
```

The terminal output shows the execution of the code:

```
PS D:\BSc. CS\2nd Year\1st Semester\SCS 2204 - Functional Programming\Practicals\Practical 10> scala "c:\Users\Nalan Tishantha\AppData\Local\Microsoft\Windows\INetCache\IE\SL67UXCZ\Q2[1].scala"
Result of y - z: 26/45
Final expression x + (y - z): 31/180
PS D:\BSc. CS\2nd Year\1st Semester\SCS 2204 - Functional Programming\Practicals\Practical 10>
```

The status bar at the bottom indicates the current file is Q2[1].scala, the encoding is UTF-8, and the language is Scala.

Question 03:



```
Q3.scala > {} BankApp > main
42 object BankApp {
43     def main(args: Array[String]): Unit = {

PS D:\BSc. CS\2nd Year\1st Semester\SCS 2204 - Functional Programming\Practicals\Practical 10> scala "d:\BSc. CS\2nd Year\1st Semester\SCS 2204 - Functional Programming\Practicals\Practical 10\Q3.scala"

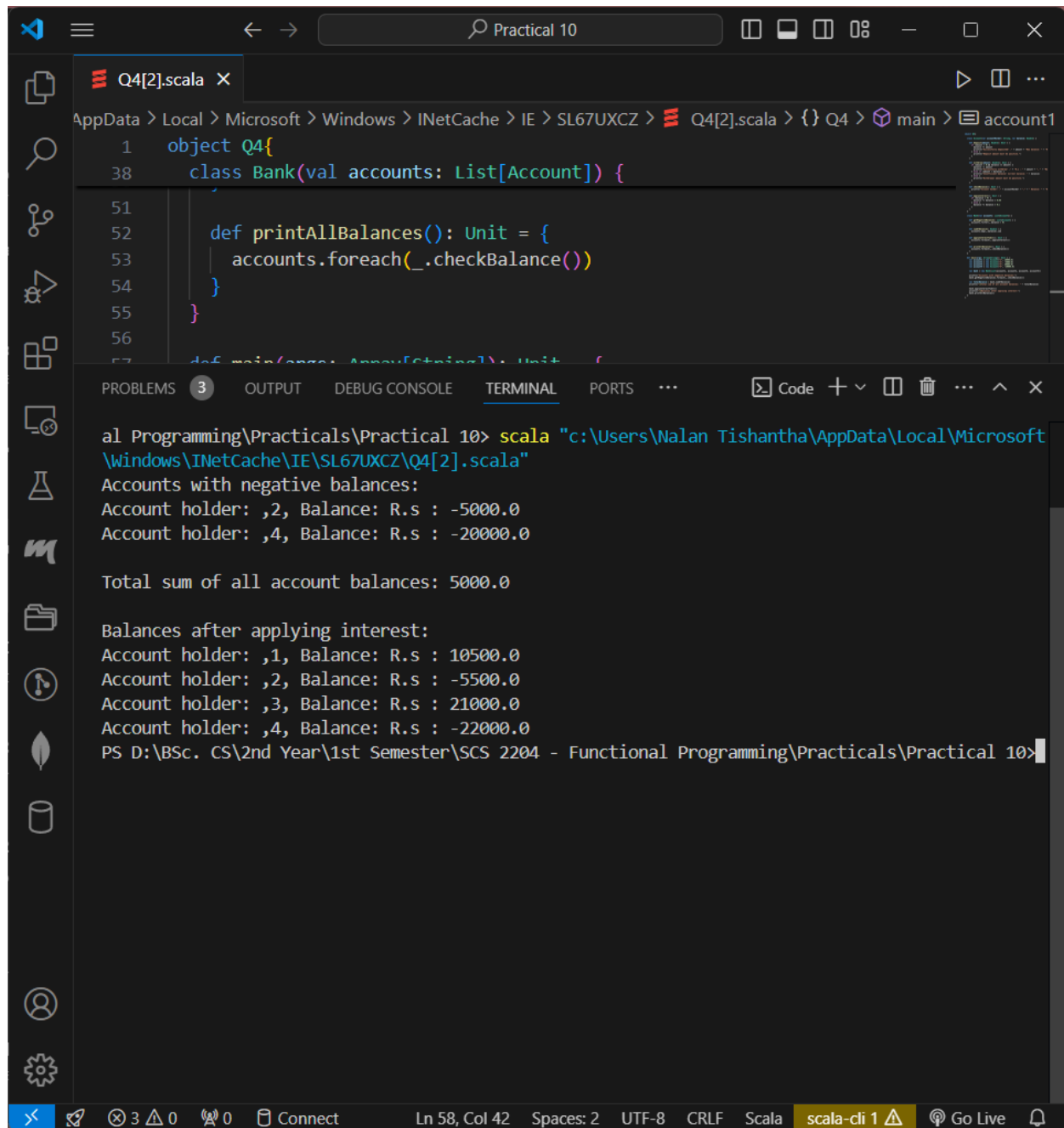
Choose an operation: 1) Deposit 2) Withdraw 3) Transfer 4) Check Balance
1
Enter deposit amount for account1:
1000
Deposited 1000.0. New balance is 2000.0.
Do you want to perform another operation? (yes/no)yes

Choose an operation: 1) Deposit 2) Withdraw 3) Transfer 4) Check Balance
2
Enter withdrawal amount for account1:
500
Withdrawn 500.0. New balance is 1500.0.
Do you want to perform another operation? (yes/no)yes

Choose an operation: 1) Deposit 2) Withdraw 3) Transfer 4) Check Balance
3
Enter transfer amount from account1 to account2:
200
Withdrawn 200.0. New balance is 1300.0.
Deposited 200.0. New balance is 700.0.
Transferred 200.0 to the other account.
Do you want to perform another operation? (yes/no)yes

Choose an operation: 1) Deposit 2) Withdraw 3) Transfer 4) Check Balance
4
Current balance: 1300.0
Current balance: 700.0
Do you want to perform another operation? (yes/no)no
Thank you for using the banking system!
PS D:\BSc. CS\2nd Year\1st Semester\SCS 2204 - Functional Programming\Practicals\Practical 10>
```

Question 04:



The screenshot shows an IDE with a Scala file named `Q4[2].scala` open. The code defines an object `Q4` containing a class `Bank` that takes a list of `Account` objects. It has a method `printAllBalances` that iterates over the accounts and calls `checkBalance`. The `main` method is partially visible at the bottom of the code editor.

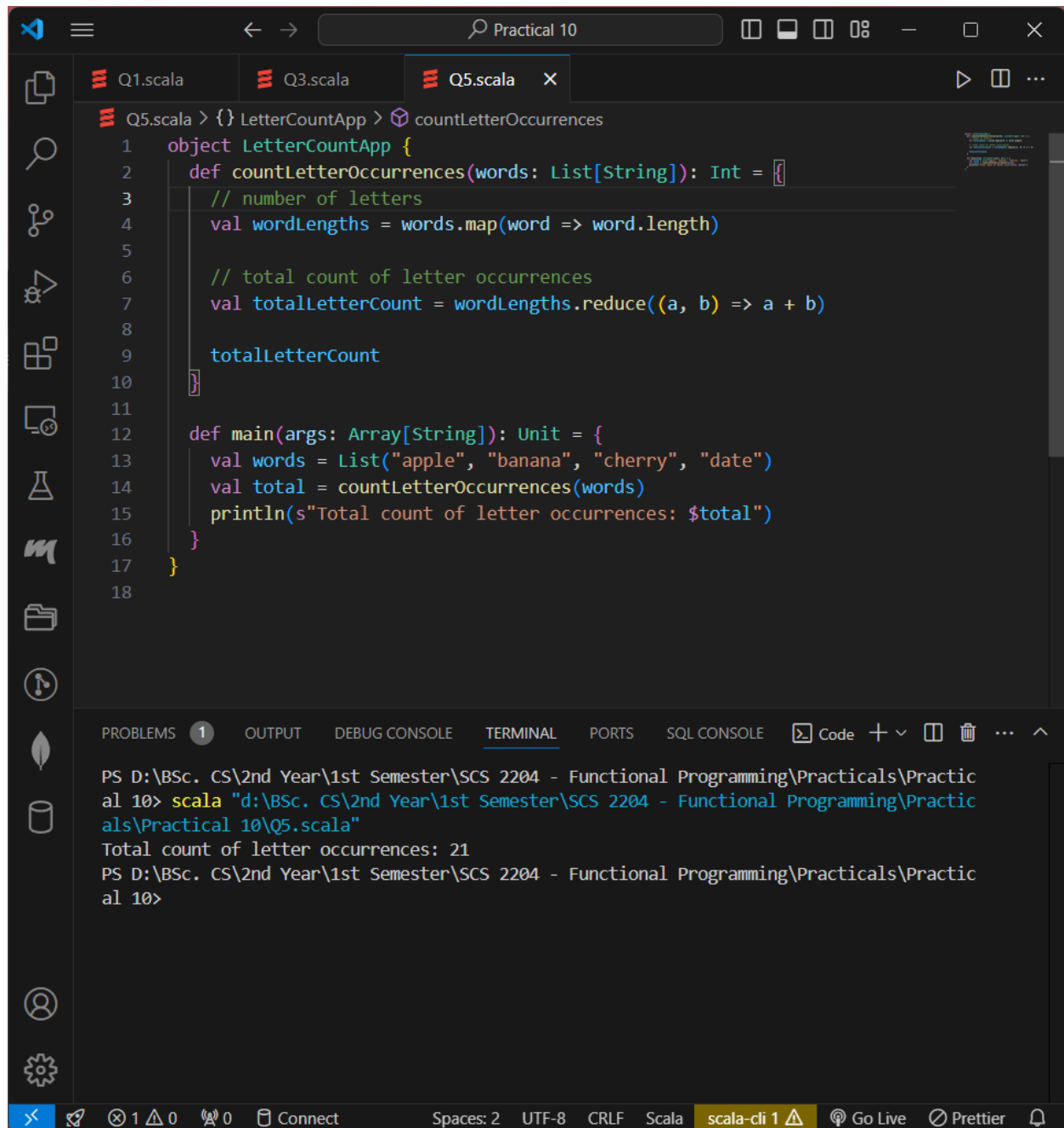
The terminal window shows the execution of the program using the `scala` command. The output displays the accounts with negative balances, the total sum of all account balances, and the balances after applying interest.

```
al Programming\Practicals\Practical 10> scala "c:\Users\Walan Tishantha\AppData\Local\Microsoft\Windows\INetCache\IE\SL67UXCZ\Q4[2].scala"
Accounts with negative balances:
Account holder: ,2, Balance: R.s : -5000.0
Account holder: ,4, Balance: R.s : -20000.0

Total sum of all account balances: 5000.0

Balances after applying interest:
Account holder: ,1, Balance: R.s : 10500.0
Account holder: ,2, Balance: R.s : -5500.0
Account holder: ,3, Balance: R.s : 21000.0
Account holder: ,4, Balance: R.s : -22000.0
PS D:\BSc. CS\2nd Year\1st Semester\SCS 2204 - Functional Programming\Practicals\Practical 10>
```

Question 05:



The screenshot shows an IDE with a Scala file named Q5.scala. The code defines an object LetterCountApp with a method countLetterOccurrences that takes a List[String] and returns an Int. It calculates the total count of letter occurrences by mapping each word to its length and then reducing the list. A main method is also defined, which uses the countLetterOccurrences method on a list of words: "apple", "banana", "cherry", and "date". The terminal output shows the command to run the program and the resulting output: "Total count of letter occurrences: 21".

```
Q5.scala > {} LetterCountApp > countLetterOccurrences
1 object LetterCountApp {
2   def countLetterOccurrences(words: List[String]): Int = {
3     // number of letters
4     val wordLengths = words.map(word => word.length)
5
6     // total count of letter occurrences
7     val totalLetterCount = wordLengths.reduce((a, b) => a + b)
8
9     totalLetterCount
10  }
11
12  def main(args: Array[String]): Unit = {
13    val words = List("apple", "banana", "cherry", "date")
14    val total = countLetterOccurrences(words)
15    println(s"Total count of letter occurrences: $total")
16  }
17 }
18
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS SQL CONSOLE Code + - - - ^

PS D:\BSc. CS\2nd Year\1st Semester\SCS 2204 - Functional Programming\Practicals\Practical 10> scala "d:\BSc. CS\2nd Year\1st Semester\SCS 2204 - Functional Programming\Practicals\Practical 10\Q5.scala"

Total count of letter occurrences: 21

PS D:\BSc. CS\2nd Year\1st Semester\SCS 2204 - Functional Programming\Practicals\Practical 10>

Spaces: 2 UTF-8 CRLF Scala scala-cli 1 Go Live Prettier