Scala Programming

(14/09/2023)

Name: Niranjana J
USN: 22BTRAD027
Branch: 22BTRAD027

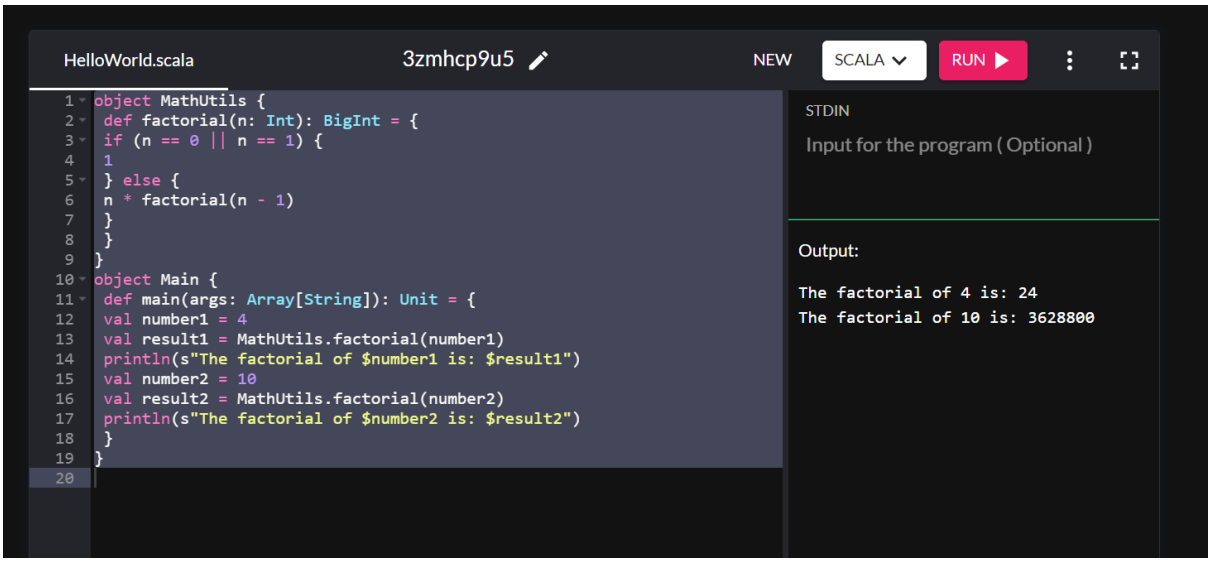
Question:

Finding the factorial of an integer using recursive function.

Code:

```
object MathUtils {
 def factorial(n: Int): BigInt = {
 if (n == 0 || n == 1) {
 1
 } else {
 n * factorial(n - 1)
 }
 }
}
object Main {
 def main(args: Array[String]): Unit = {
 val number1 = 4
 val result1 = MathUtils.factorial(number1)
 println(s"The factorial of $number1 is: $result1")
 val number2 = 10
 val result2 = MathUtils.factorial(number2)
 println(s"The factorial of $number2 is: $result2")
 }
}
```

Output:



Since we are using recursion to find the factorial of the required number, it is essential to define a base case so that the function can break out of the function when necessary. For instance, to find the factorial of number n, we have to find the product of the numbers from n to 1 (ie, n*(n-1)*(n-2)*...*3*2*1). In the base case, we predefine the values of 0! and 1! As 1. For numbers greater than 1, we use recursion. At first we multiply n with the factorial of n-1 (to find the factorial of n-1, we pass n-1 into the factorial function unless n-1 becomes 1). The value returned from the factorial function each time it is called, is multiplied with the next number until we get the factorial of n. Recursion requires the minimum number of codes hence it improves the efficiency of the code.

Github link:
https://github.com/niranjana628/Scala-Programming