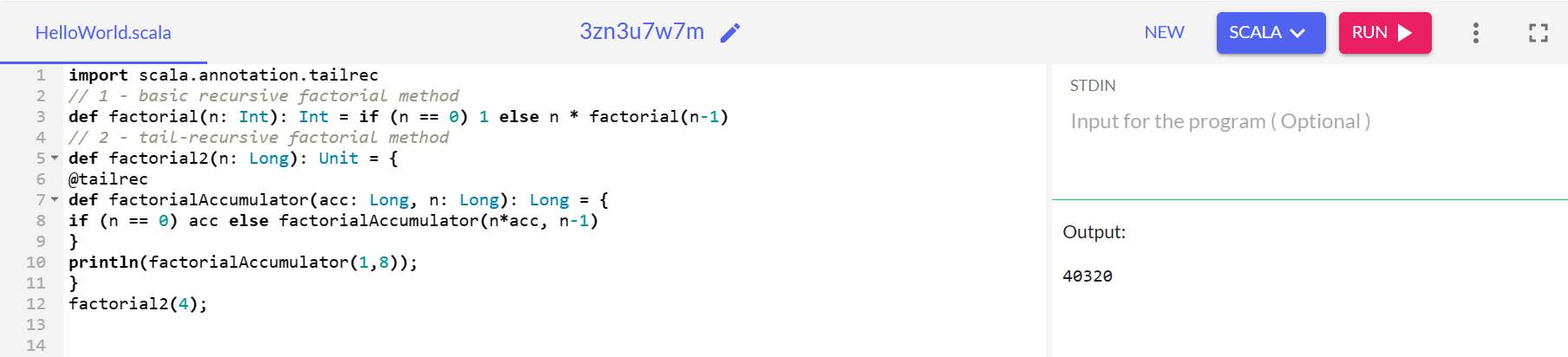
SCALA PROGRAMMING

**Factorial problem:** For the defined number of values. In this problem statement the factorial runs up to 8!



**CODE:**

import scala.annotation.tailrec

// 1 - basic recursive factorial method

def factorial(n: Int): Int = if (n == 0) 1 else n \* factorial(n-1)

// 2 - tail-recursive factorial method

def factorial2(n: Long): Unit = {

@tailrec

def factorialAccumulator(acc: Long, n: Long): Long = {

if (n == 0) acc else factorialAccumulator(n\*acc, n-1)

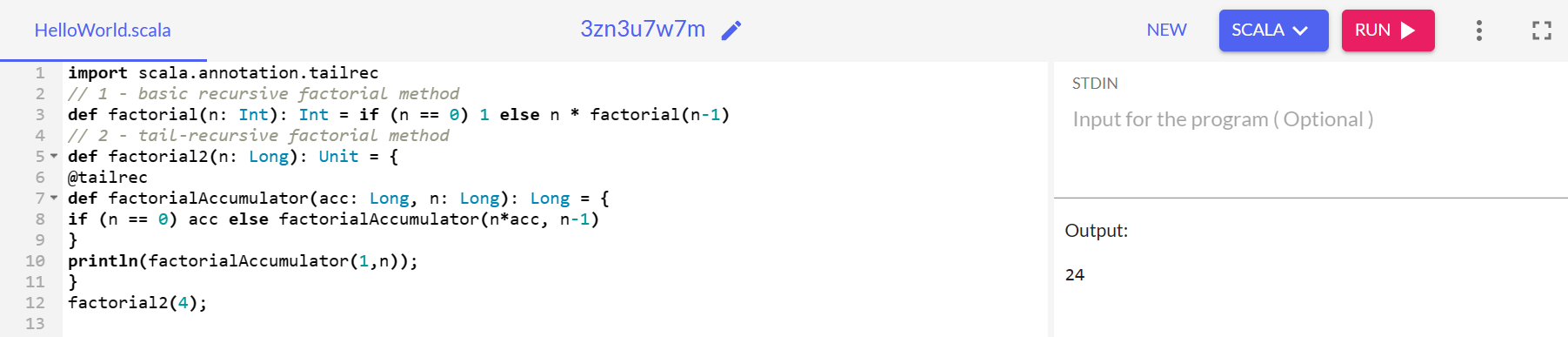
}

println(factorialAccumulator(1,8));

}

factorial2(4);

**Factorial problem:** If the factorial is up to n! in one of the given function the other function will be executed



**CODE:**

import scala.annotation.tailrec

// 1 - basic recursive factorial method

def factorial(n: Int): Int = if (n == 0) 1 else n \* factorial(n-1)

// 2 - tail-recursive factorial method

def factorial2(n: Long): Unit = {

@tailrec

def factorialAccumulator(acc: Long, n: Long): Long = {

if (n == 0) acc else factorialAccumulator(n\*acc, n-1)

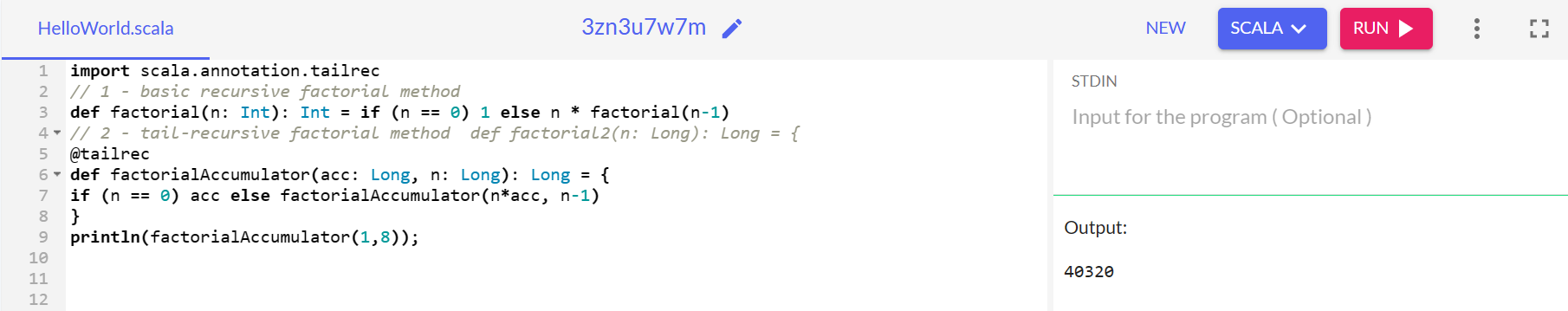
}

println(factorialAccumulator(1,n));

}

factorial2(4);

**Factorial problem:** Other way of solving the factorial statement by commenting the error function



**CODE:**

import scala.annotation.tailrec

// 1 - basic recursive factorial method

def factorial(n: Int): Int = if (n == 0) 1 else n \* factorial(n-1)

// 2 - tail-recursive factorial method def factorial2(n: Long): Long = {

@tailrec

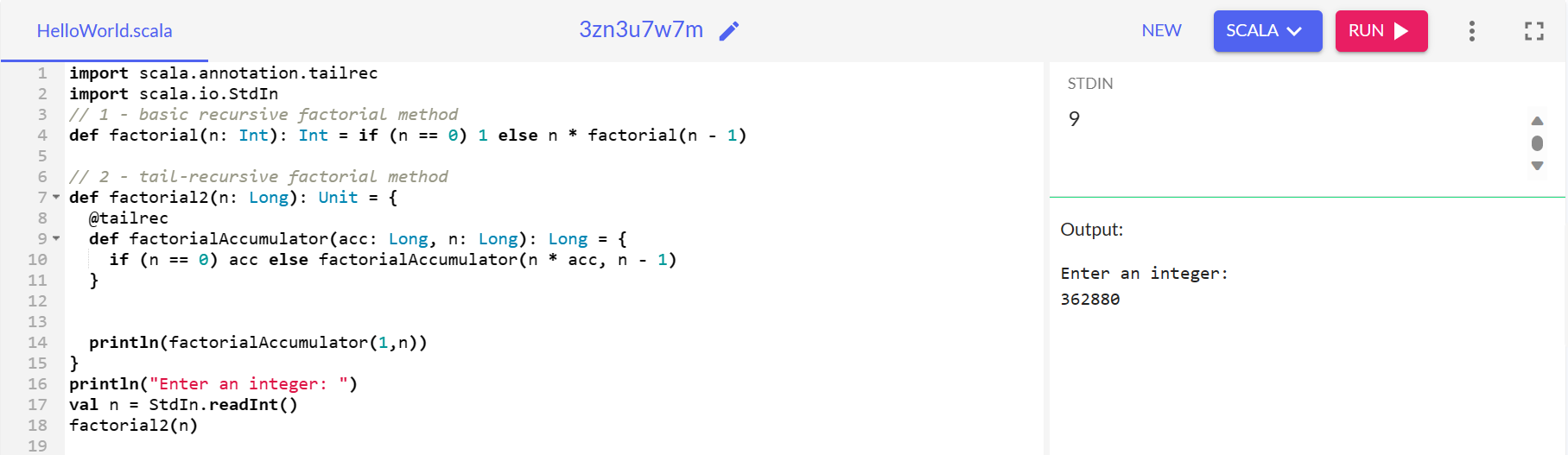
def factorialAccumulator(acc: Long, n: Long): Long = {

if (n == 0) acc else factorialAccumulator(n\*acc, n-1)

}

println(factorialAccumulator(1,8));

**Factorial problem:** Adding a scanner to provide user defined values to the code.



**CODE:**

import scala.annotation.tailrec

import scala.io.StdIn

// 1 - basic recursive factorial method

def factorial(n: Int): Int = if (n == 0) 1 else n \* factorial(n - 1)

// 2 - tail-recursive factorial method

def factorial2(n: Long): Unit = {

@tailrec

def factorialAccumulator(acc: Long, n: Long): Long = {

if (n == 0) acc else factorialAccumulator(n \* acc, n - 1)

}

println(factorialAccumulator(1,n))

}

println("Enter an integer: ")

val n = StdIn.readInt()

factorial2(n)