Name: Shamin Chokshi SEC1 (NUID: 002763957)

Big Data System Engineering with Scala
Spring 2023
Assignment 5(Functional Composition)

-List of tasks Implemented

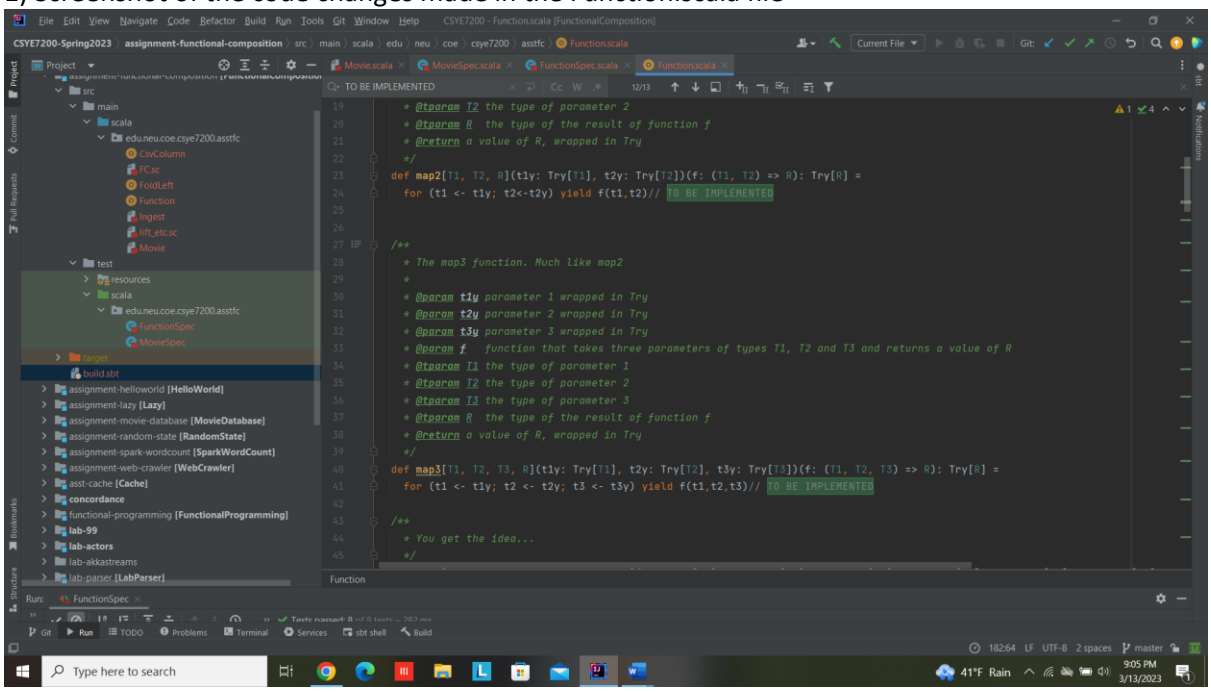
1)First I went to the Assignment-functional composition directory and opened Function.scala and Movie.scala where I had to make changes in the code

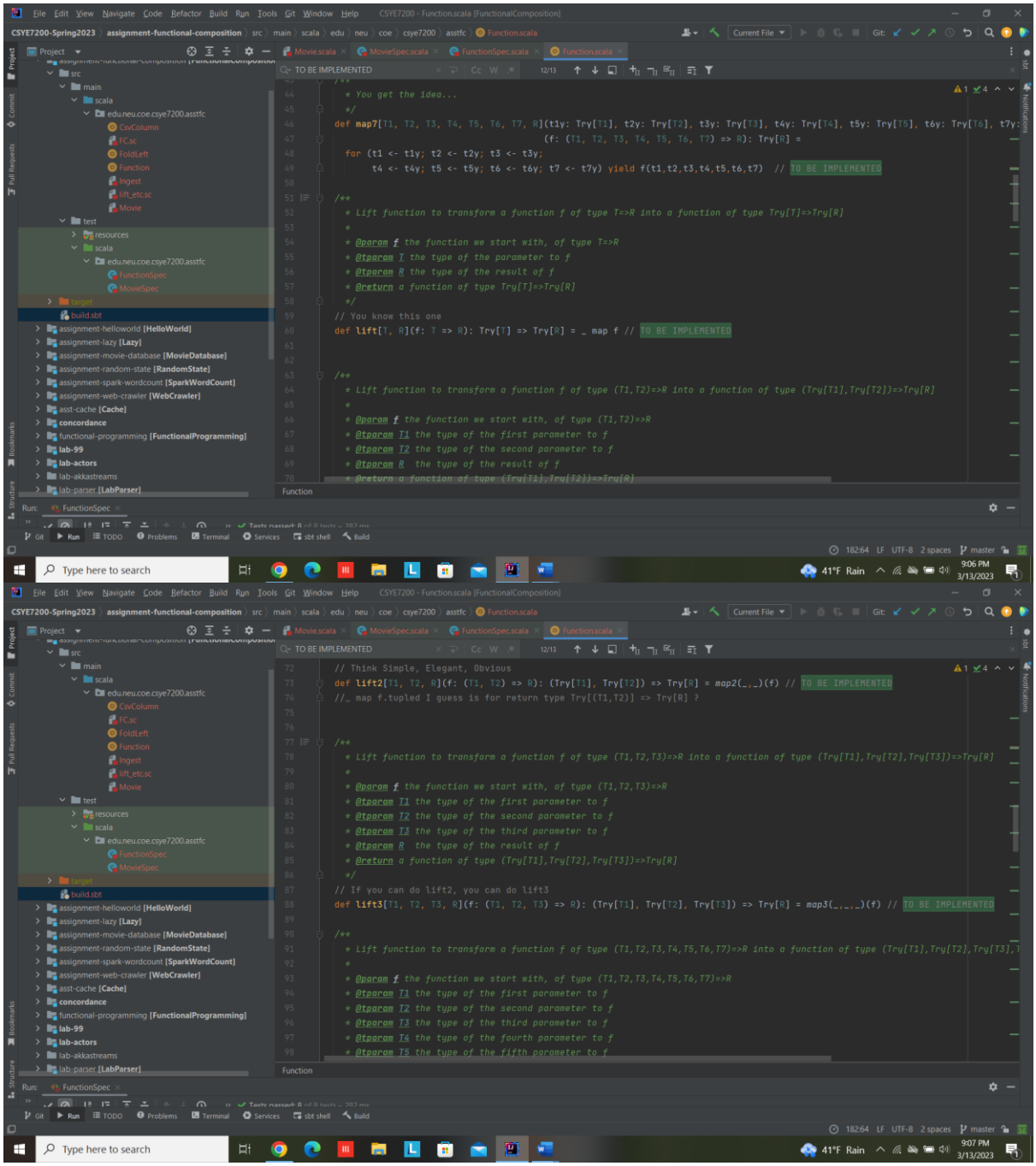
2) The next task was to run the file Moviespec.scala and functionspec.scala to test all the test cases

- Findings and analysis

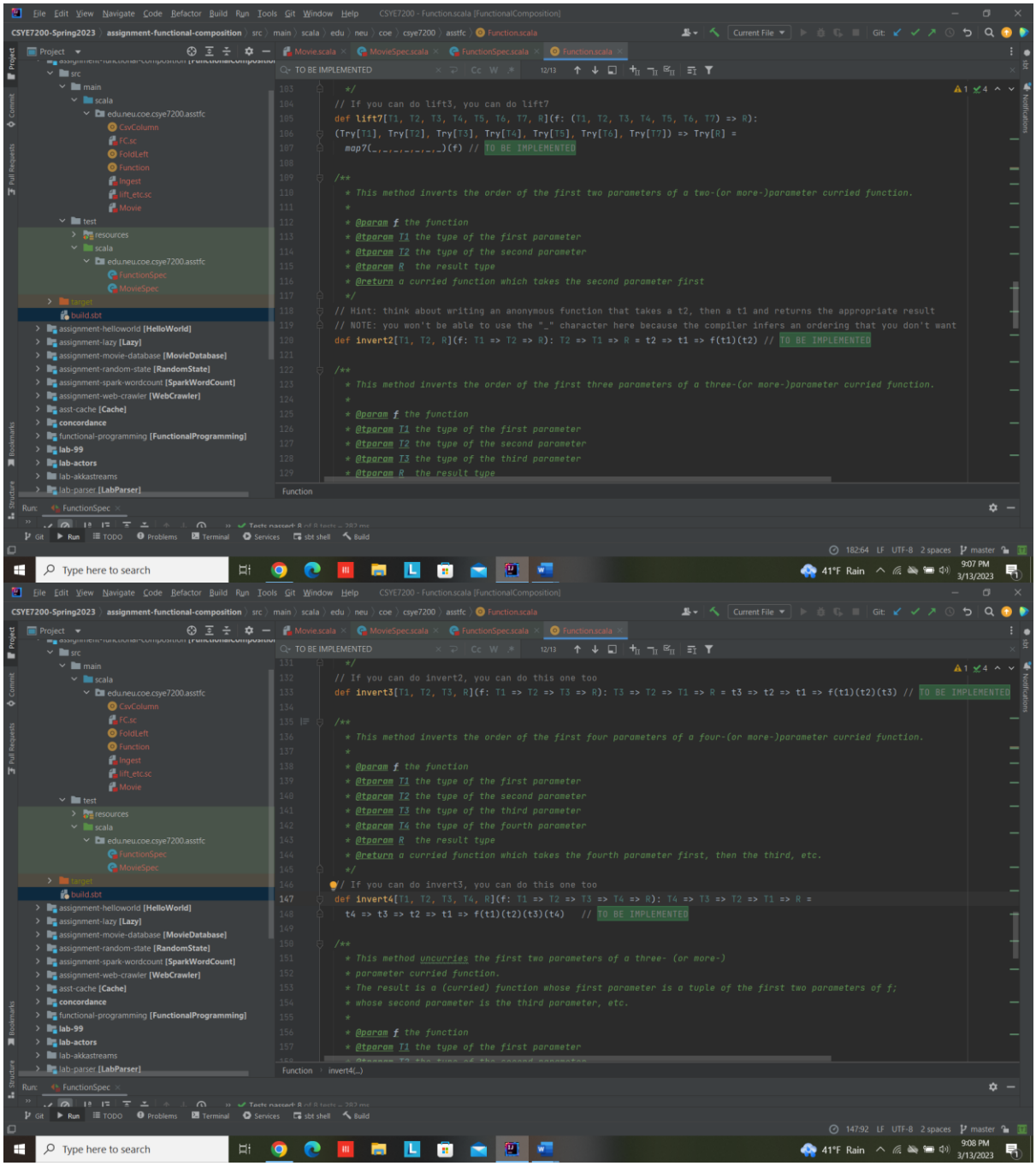
-Code

1) Screenshot of the code changes made in the Function.scala file

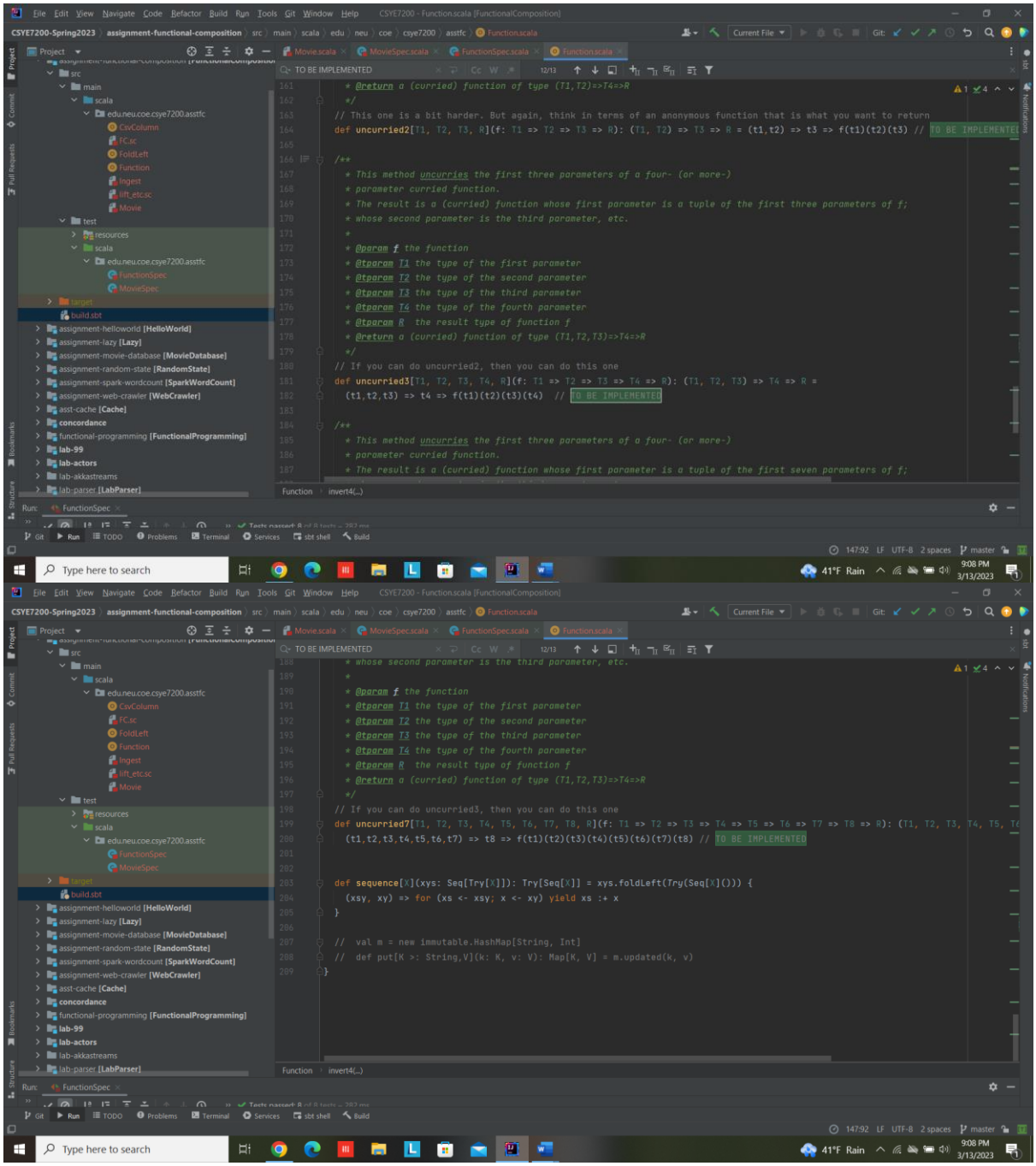



File Edit View Navigate Code Refactor Build Run Tools Git Window Help
CSYE7200 - Function.scala [FunctionalComposition]
CSYE7200-Spring2023 assignment-functional-composition src main scala edu neu coe csye7200 asstfc Function.scala
Movie.scala MovieSpec.scala FunctionSpec.scala Function.scala
TO BE IMPLEMENTED
/**
* You get the idea...
*/
def map7[T1, T2, T3, T4, T5, T6, T7, R](t1y: Try[T1], t2y: Try[T2], t3y: Try[T3], t4y: Try[T4], t5y: Try[T5], t6y: Try[T6], t7y:
(f: (T1, T2, T3, T4, T5, T6, T7) => R): Try[R] =
for (t1 <- t1y; t2 <- t2y; t3 <- t3y;
t4 <- t4y; t5 <- t5y; t6 <- t6y; t7 <- t7y) yield f(t1,t2,t3,t4,t5,t6,t7) // TO BE IMPLEMENTED
/**
* Lift function to transform a function f of type T=>R into a function of type Try[T]=>Try[R]
*
* @param f the function we start with, of type T=>R
* @tparam T the type of the parameter to f
* @tparam R the type of the result of f
* @return a function of type Try[T]=>Try[R]
*/
// You know this one
def lift[T, R](f: T => R): Try[T] => Try[R] = _ map f // TO BE IMPLEMENTED
/**
* Lift function to transform a function f of type (T1,T2)=>R into a function of type (Try[T1],Try[T2])=>Try[R]
*
* @param f the function we start with, of type (T1,T2)=>R
* @tparam T1 the type of the first parameter to f
* @tparam T2 the type of the second parameter to f
* @tparam R the type of the result of f
* @return a function of type (Try[T1],Try[T2])=>Try[R]
Function
Run: FunctionSpec
Git Run TODO Problems Terminal Services sbt shell Build
182:64 LF UTF-8 2 spaces master
Type here to search
41°F Rain
9:06 PM
3/13/2023

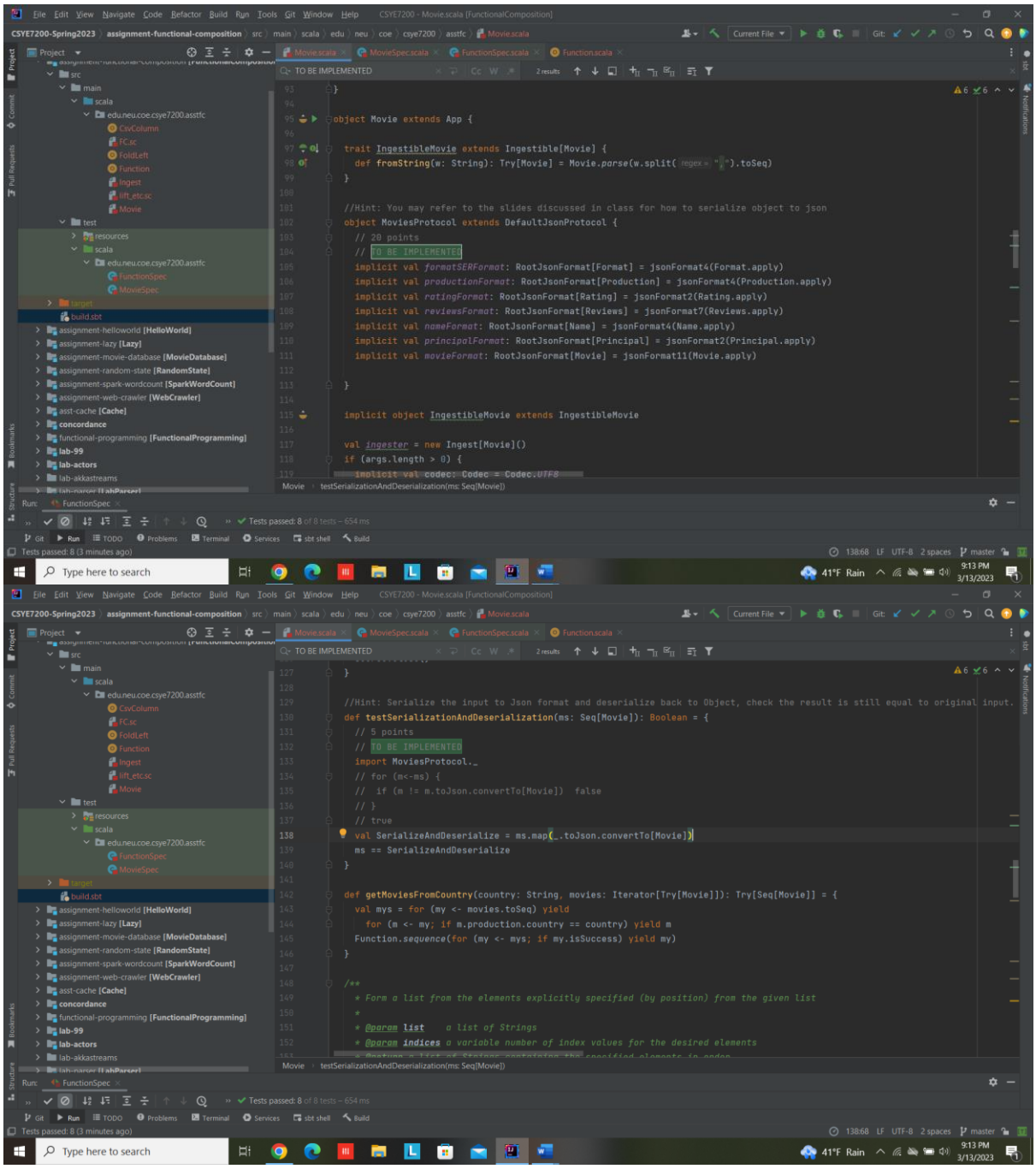
File Edit View Navigate Code Refactor Build Run Tools Git Window Help
CSYE7200 - Function.scala [FunctionalComposition]
CSYE7200-Spring2023 assignment-functional-composition src main scala edu neu coe csye7200 asstfc Function.scala
Movie.scala MovieSpec.scala FunctionSpec.scala Function.scala
TO BE IMPLEMENTED
// Think Simple, Elegant, Obvious
def lift2[T1, T2, R](f: (T1, T2) => R): (Try[T1], Try[T2]) => Try[R] = map2(_,_)(f) // TO BE IMPLEMENTED
//_ map f.tupled I guess is for return type Try[(T1,T2)] => Try[R] ?
/**
* Lift function to transform a function f of type (T1,T2,T3)=>R into a function of type (Try[T1],Try[T2],Try[T3])=>Try[R]
*
* @param f the function we start with, of type (T1,T2,T3)=>R
* @tparam T1 the type of the first parameter to f
* @tparam T2 the type of the second parameter to f
* @tparam T3 the type of the third parameter to f
* @tparam R the type of the result of f
* @return a function of type (Try[T1],Try[T2],Try[T3])=>Try[R]
*/
// If you can do lift2, you can do lift3
def lift3[T1, T2, T3, R](f: (T1, T2, T3) => R): (Try[T1], Try[T2], Try[T3]) => Try[R] = map3(_,_,_)(f) // TO BE IMPLEMENTED
/**
* Lift function to transform a function f of type (T1,T2,T3,T4,T5,T6,T7)=>R into a function of type (Try[T1],Try[T2],Try[T3],
*
* @param f the function we start with, of type (T1,T2,T3,T4,T5,T6,T7)=>R
* @tparam T1 the type of the first parameter to f
* @tparam T2 the type of the second parameter to f
* @tparam T3 the type of the third parameter to f
* @tparam T4 the type of the fourth parameter to f
* @tparam T5 the type of the fifth parameter to f
Function
Run: FunctionSpec
Git Run TODO Problems Terminal Services sbt shell Build
182:64 LF UTF-8 2 spaces master
Type here to search
41°F Rain
9:07 PM
3/13/2023


File Edit View Navigate Code Refactor Build Run Tools Git Window Help
CSYE7200 - Function.scala [FunctionalComposition]
CSYE7200-Spring2023 assignment-functional-composition src main scala edu neu coe csye7200 asstfc Function.scala
Movie.scala MovieSpec.scala FunctionSpec.scala Function.scala
TO BE IMPLEMENTED
12/13
// If you can do lift3, you can do lift7
def lift7[T1, T2, T3, T4, T5, T6, T7, R](f: (T1, T2, T3, T4, T5, T6, T7) => R):
(Try[T1], Try[T2], Try[T3], Try[T4], Try[T5], Try[T6], Try[T7]) => Try[R] =
map7(_._._._._._._)(f) // TO BE IMPLEMENTED
/**
* This method inverts the order of the first two parameters of a two-(or more-)parameter curried function.
*
* @param f the function
* @tparam T1 the type of the first parameter
* @tparam T2 the type of the second parameter
* @tparam R the result type
* @return a curried function which takes the second parameter first
*/
// Hint: think about writing an anonymous function that takes a t2, then a t1 and returns the appropriate result
// NOTE: you won't be able to use the "_" character here because the compiler infers an ordering that you don't want
def invert2[T1, T2, R](f: T1 => T2 => R): T2 => T1 => R = t2 => t1 => f(t1)(t2) // TO BE IMPLEMENTED
/**
* This method inverts the order of the first three parameters of a three-(or more-)parameter curried function.
*
* @param f the function
* @tparam T1 the type of the first parameter
* @tparam T2 the type of the second parameter
* @tparam T3 the type of the third parameter
* @tparam R the result type
Function
Run: FunctionSpec
Git Run TODO Problems Terminal Services sbt shell Build
182:64 LF UTF-8 2 spaces master
Type here to search
41°F Rain
9:07 PM
3/13/2023

File Edit View Navigate Code Refactor Build Run Tools Git Window Help
CSYE7200 - Function.scala [FunctionalComposition]
CSYE7200-Spring2023 assignment-functional-composition src main scala edu neu coe csye7200 asstfc Function.scala
Movie.scala MovieSpec.scala FunctionSpec.scala Function.scala
TO BE IMPLEMENTED
12/13
// If you can do invert2, you can do this one too
def invert3[T1, T2, T3, R](f: T1 => T2 => T3 => R): T3 => T2 => T1 => R = t3 => t2 => t1 => f(t1)(t2)(t3) // TO BE IMPLEMENTED
/**
* This method inverts the order of the first four parameters of a four-(or more-)parameter curried function.
*
* @param f the function
* @tparam T1 the type of the first parameter
* @tparam T2 the type of the second parameter
* @tparam T3 the type of the third parameter
* @tparam T4 the type of the fourth parameter
* @tparam R the result type
* @return a curried function which takes the fourth parameter first, then the third, etc.
*/
// If you can do invert3, you can do this one too
def invert4[T1, T2, T3, T4, R](f: T1 => T2 => T3 => T4 => R): T4 => T3 => T2 => T1 => R =
t4 => t3 => t2 => t1 => f(t1)(t2)(t3)(t4) // TO BE IMPLEMENTED
/**
* This method uncurries the first two parameters of a three- (or more-)
* parameter curried function.
* The result is a (curried) function whose first parameter is a tuple of the first two parameters of f;
* whose second parameter is the third parameter, etc.
*
* @param f the function
* @tparam T1 the type of the first parameter
Function invert4(...)
Run: FunctionSpec
Git Run TODO Problems Terminal Services sbt shell Build
147:92 LF UTF-8 2 spaces master
Type here to search
41°F Rain
9:08 PM
3/13/2023

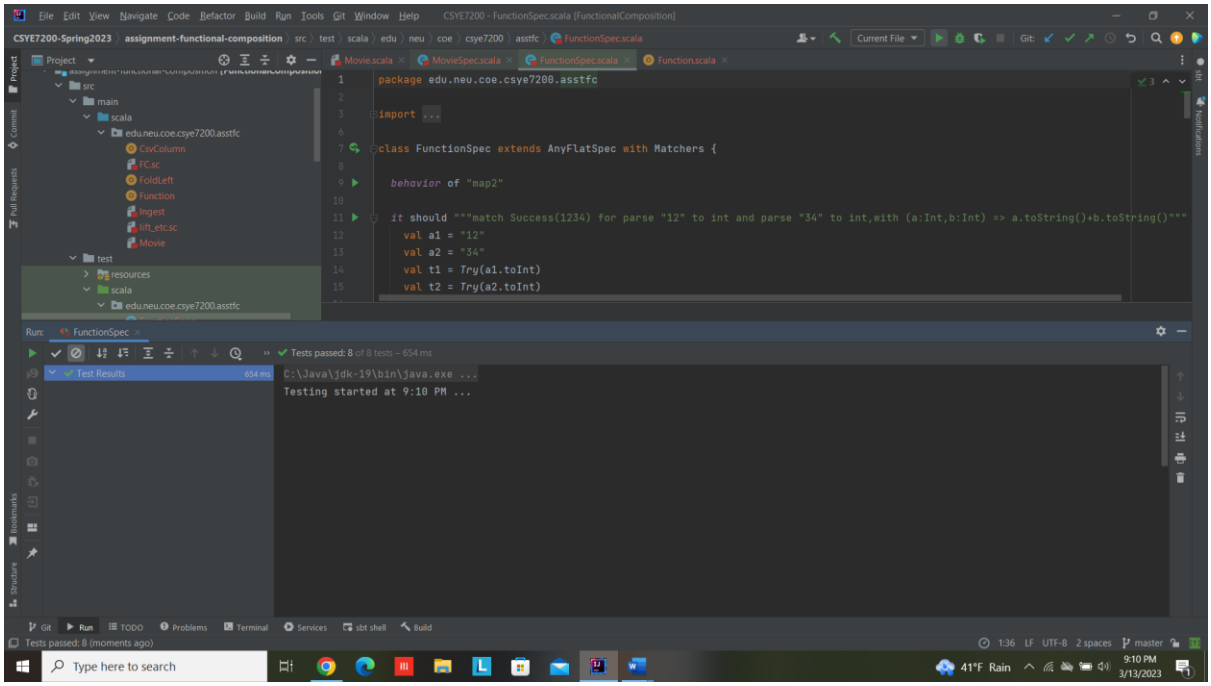


2) Changes mage in Movie.scala

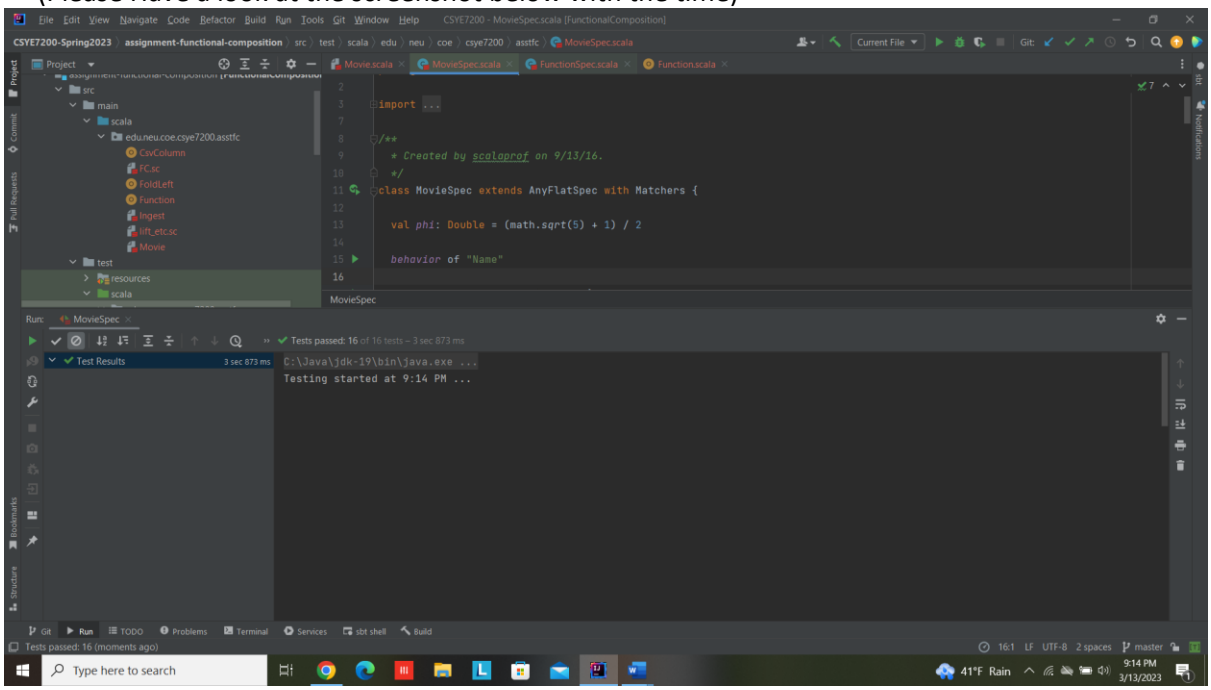


-Unit Tests

1) Screenshot of the file FunctionSpec.scala that was run and I also ran test cases individually
   It tested and successfully passed 8 Test cases
   (Please Have a look at the screenshot below with the time)



2) Screenshot of the file MovieSpec.scala that was run and I also ran test cases individually
It tested and successfully passed 16 Test cases
(Please Have a look at the screenshot below with the time)



-Result

After observation and running of the code a total of 24 test cases was run
8 in Functionspec.scala and 16 in Moviespec.scala