Weikun Zhuang – SEC01 (NUID 001537998)

Big Data System Engineering with Scala
Fall 2022
Assignment No. #5

-List of Tasks Implemented

Implemented 13 TODOs in *Function.scala* and 2 TODOs in *Movie.scala*

-Code

```
def map2[T1, T2, R](t1y: Try[T1], t2y: Try[T2])(f: (T1, T2) => R): Try[R] =
  for (t1 <- t1y; t2<-t2y) yield f(t1,t2)// TO BE IMPLEMENTED
```

```
def map3[T1, T2, T3, R](t1y: Try[T1], t2y: Try[T2], t3y: Try[T3])(f: (T1, T2, T3) => R): Try[R] =
  for (t1 <- t1y; t2 <- t2y; t3 <- t3y) yield f(t1,t2,t3)// TO BE IMPLEMENTED
```

```
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
                                        (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
```

```
def lift[T, R](f: T => R): Try[T] => Try[R] = _ map f // TO BE IMPLEMENTED
```

```
def lift2[T1, T2, R](f: (T1, T2) => R): (Try[T1], Try[T2]) => Try[R] = map2(_,_)(f) // TO BE IMPLEMENTED
```

```
def lift3[T1, T2, T3, R](f: (T1, T2, T3) => R): (Try[T1], Try[T2], Try[T3]) => Try[R] = map3(_,_,_)(f) /
```

```
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
```

```
def invert2[T1, T2, R](f: T1 => T2 => R): T2 => T1 => R = t2 => t1 => f(t1)(t2) // TO BE IMPLEMENTED
```

```
def invert3[T1, T2, T3, R](f: T1 => T2 => T3 => R): T3 => T2 => T1 => R = t3 => t2 => t1 => f(t1)(t2)(t3) /
```

```
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
```

```
def uncurried2[T1, T2, T3, R](f: T1 => T2 => T3 => R): (T1, T2) => T3 => R = (t1,t2) => t3 => f(t1)(t2)(t3)
```

```
def uncurried3[T1, T2, T3, T4, R](f: T1 => T2 => T3 => T4 => R): (T1, T2, T3) => T4 => R =
  (t1,t2,t3) => t4 => f(t1)(t2)(t3)(t4)  // TO BE IMPLEMENTED
```

```
def uncurried7[T1, T2, T3, T4, T5, T6, T7, T8, R](f: T1 => T2 => T3 => T4 => T5 => T6 => T7 => T8 => R): (T1, T2,
  (t1,t2,t3,t4,t5,t6,t7) => t8 => f(t1)(t2)(t3)(t4)(t5)(t6)(t7)(t8) // TO BE IMPLEMENTED
```

```scala
object MoviesProtocol extends DefaultJsonProtocol {
  // 20 points
  // TO BE IMPLEMENTED
  implicit val formatSERFormat: RootJsonFormat[Format] = jsonFormat4(Format.apply)
  implicit val productionFormat: RootJsonFormat[Production] = jsonFormat4(Production.apply)
  implicit val ratingFormat: RootJsonFormat[Rating] = jsonFormat2(Rating.apply)
  implicit val reviewsFormat: RootJsonFormat[Reviews] = jsonFormat7(Reviews.apply)
  implicit val nameFormat: RootJsonFormat[Name] = jsonFormat4(Name.apply)
  implicit val principalFormat: RootJsonFormat[Principal] = jsonFormat2(Principal.apply)
  implicit val movieFormat: RootJsonFormat[Movie] = jsonFormat11(Movie.apply)
```

```scala
def testSerializationAndDeserialization(ms: Seq[Movie]): Boolean = {
  // 5 points
  // TO BE IMPLEMENTED
  import MoviesProtocol._
  // for (m<-ms) {
  //  if (m != m.toJson.convertTo[Movie])  false
  // }
  // true
  val SerializeAndDeserialize = ms.map(_.toJson.convertTo[Movie])
  ms == SerializeAndDeserialize
}
```

-Unit tests

```
[info] FunctionSpec:
[info] map2
[info] - should match Success(1234) for parse "12" to int and parse "34" to int,with (a:Int,b:Int) => a.toString()+b.toString()
[info] - should fail for "", Int
[info] map7
[info] - should success
[info] - should fail with bad input
[info] invert2
[info] - should work
[info] invert3
[info] - should work
[info] invert4
[info] - should work
[info] uncurried2
[info] - should work
[info] MovieSpec:
[info] Name
[info] - should work for String
[info] - should work for Name
[info] Principal
[info] - should work for List[String]
[info] Rating
[info] - should work for String, Int
[info] - should work for PG-13
[info] - should work for R
[info] - should work for PG-0-
[info] - should work for PG-XX
[info] Format
[info] - should work for Boolean, String, Double, Int
[info] - should work for List[String]
[info] Production
[info] - should work for String, Int
[info] - should work for List[String]
[info] - should define isKiwi properly
[info] Reviews
[info] - should work for List[String]
[info] Movie.getMoviesFromCountry
[info] - should work for the sample file
[info] Movie.testSerializationAndDeserialization
[info] - should work for the sample file
[info] Run completed in 1 second, 22 milliseconds.
[info] Total number of tests run: 24
[info] Suites: completed 2, aborted 0
[info] Tests: succeeded 24, failed 0, canceled 0, ignored 0, pending 0
[info] All tests passed.
[success] Total time: 20 s, completed Oct 30, 2022, 4:28:18 PM
```