Intermediate Scala

Practical Exercises

Chapter 5: Algebraic Data Types

1. Define a simple Algebraic Sum Data Type to represent days of the week. Provide the methods **`tomorrow`** and **`nextBusinessDay`** for the type.

2. Look at the **`Box[A]`** type defined in the slides. What changes, if any, are needed to make this type covariant in its element type? Make the same changes to the MyList[A] type.

3. Using the fold method defined on the MyList[A] type, implement the method

   `contains(el: A): Boolean`

   which returns true if the list contains the element supplied as the parameter, and false otherwise.

4. Binary Tree is a type that can easily be defined using a recursive ADT. A Binary Tree is
   - A leaf node, which contains a value of some type
   - An internal node, which contains a left and a right node

   Implement this type, and provide a **`fold`** method for it. Using the fold method, implement a **`length`** method, which returns the number of leaf nodes in the tree.