Either

Either is an alternate to `Option` with regards to Success or failure. An instance of `Either` is an instance of either `Left` or `Right`. A common use of `Either` is as an alternative to scala. `Option` for dealing with possibly missing values. In this usage, `None` is replaced with a `Left` which can contain useful information. Right takes the place of `Some`. Convention dictates that `Left` is used for **failure** and `Right` is used for **success**. For example, you could use `Either[String, Int]` to indicate whether a received input is a String or an Int. Here is an example :

```
object EitherLeftRightExample extends App {

  /**
   * A method to demonstrate how to declare that a method returns an Either,
   * and code that returns a Left or Right.
   */
  def divideXByY(x: Int, y: Int): Either[String, Int] = {
      if (y == 0) Left("oops, can't divide by 0")
      else Right(x / y)
  }

  // a few different ways to use Either, Left, and Right
  println(divideXByY(1, 0))
  println(divideXByY(1, 1))
  divideXByY(1, 0) match {
      case Left(s) => println("Answer: " + s)
      case Right(i) => println("Answer: " + i)
  }

}
```

We can run the pattern match also as can be seen in above example.

Exercises:

- Create a function which can either divide a number or send an exception message, return type of a fucntion should be `Either[String,Int]`
- Create a function which can send the message `Not allowed for driving` or nothing if `Allowed`, i.e `def isAllowedToVote(age:Int):Either[String,Unit]`