Option

Scala has a standard type named Option for optional values. Such a value can be of two forms. It can be of the form Some(x) where x is the actual value. Or it can be the None object, which represents a missing value. Optional values are produced by some of the standard operations on Scala's collections. For instance, the get method of Scala's Map produces `Some(value)` if a value corresponding to a given key has been found, or `None` if the given key is not defined in the Map. Here's an example:

```
scala> val capitals =
Map("France" -> "Paris", "Japan" -> "Tokyo")
capitals: scala.collection.immutable.Map[java.lang.String, java.lang.String] =
Map(France -> Paris, Japan -> Tokyo)

scala> capitals get "France"
res23: Option[java.lang.String] = Some(Paris)
scala> capitals get "North Pole" res24: Option[java.lang.String] = None
```

The most common way to take optional values apart is through a pattern match. For instance:

```
scala> def show(x: Option[String]) = x match {
                 case Some(s) => s
                 case None => "?"
              }

show: (x: Option[String])String
scala> show(capitals get "Japan")
res25: String = Tokyo
scala> show(capitals get "France")
res26: String = Paris
scala> show(capitals get "North Pole")
res27: String = ?
```

Getting the value from an Option

As a consumer of a method that returns an Option, there are several good ways to call it and access its result:

- Use getOrElse
- Use foreach
- Use a match expression

To get the actual value if the method succeeds, or use a default value if the method fails, use `getOrElse`:

```
 val x = toInt("1").getOrElse(0)
x: Int = 1
```

where `toInt()` returns Option[Int].

Because an Option is a collection with zero or one elements, the foreach method can be used in many situations:

```
toInt("1").foreach{ i =>
    println(s"Got an int: $i")
}
```

Exercises:

- Create a function which can greet a person with name if present or just print unknown user.
- Create a function which can either return `Some(T)` or `None` , print the value or assign some default if value is not present (Hint: Use `getOrElse`)