Solution Review: Decode the Contents

This lesson discusses the solution to the challenge given in the previous lesson.

```go
package main
import (
        "bufio"
        "fmt"
        "encoding/gob"
        "log"
        "os"
)

type Address struct {
        Type             string
        City             string
        Country          string
}

type VCard struct {
        FirstName       string
        LastName        string
        Addresses       []*Address
        Remark          string
}

var content     string
var vc VCard

func main() {
                // using a decoder:
        file, _ := os.Open("vcard.gob")
        defer file.Close()
        inReader := bufio.NewReader(file)
        dec := gob.NewDecoder(inReader)
        err := dec.Decode(&vc)
        if err != nil {
                log.Println("Error in decoding gob")
        }
        fmt.Println(vc)
}
```

This is the *inverse* from the code example in the previous lesson: we have a file **vcard.gob**, and we know that it contains gob data structured as `VCard`. We make an instance `vc` of `Vcard` at **line 24**. At **line 28**, we open the file, discard error-handling and close the file at the end with `defer`. At **line 30**, we

construct a reader on the file, and at **line 31**, we construct a new decoder `dec` on that reader. We use the `Decode` method on `dec` to store our data in `vc`, which we print out at **line 36**. There's usual error-handling from **line 33** to **line 35**.

That's it for the solution. In the next lesson, you'll see how Go provides support for cryptography.