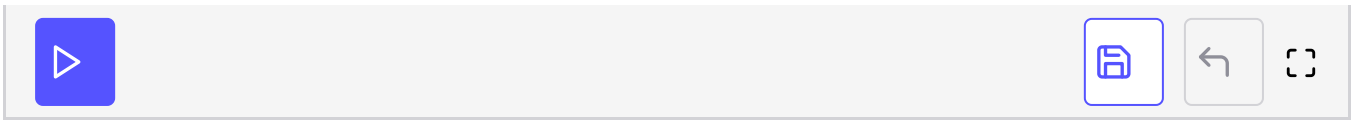
Solution Review: Anonymous Struct

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

```
package main
import "fmt"

type C struct {   // struct
        x float32
        int      // int type anonymous field
        string     // string type anonymous field
}

func main() {
        c := C{3.14, 7, "hello"}  // making struct via literal expression
        fmt.Println(c.x, c.int, c.string) // output: 3.14 7 hello
        fmt.Println(c)  // output: {3.14 7 hello}
}
```



Anonymous Struct

In the code above, look at **line 4**, where we declare a struct of type `C`. It has *three* fields. The first field is a named field `x` of type `float32`. The second and third fields are *anonymous* and of types `int` and `string`, respectively.

Now, look at the `main` function. At **line 11**, we make a variable `c` of type `C` using the literal expression: `c := C{3.14, 7, "hello"}`. So, `x` of `c` is **3.14**. The anonymous `int` and `string` variable gets 7 and **hello**, respectively. In the last two lines, we are printing the `c` struct. At **line 12**, we are printing struct's fields independently, using the selector. At **line 13**, we are printing `c` as a whole, which will automatically print all its fields enclosed in **braces**({}).

That's it about the solution. In the next lesson, you'll be studying *methods*, an important concept in Go.