Solution Review: Decide Employee Salary

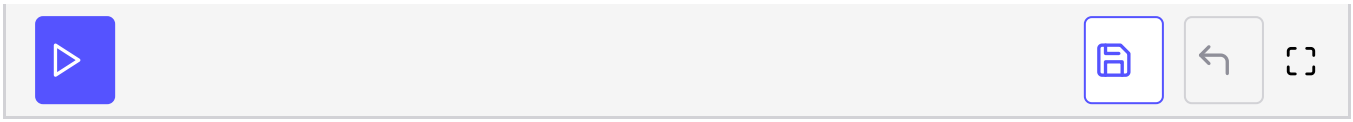
This lesson discusses the solution to the challenge given in the previous lesson.

```
package main
import "fmt"

/* basic data structure upon which we'll define methods */
type employee struct {
     salary float32
}

/* a method which will add a specified percent to an
   employees salary */
func (this *employee) giveRaise(pct float32) {
     this.salary += this.salary * pct
}

func main() {
     /* create an employee instance */
     var e = new(employee)
     e.salary = 100000;
     /* call our method */
     e.giveRaise(0.04)
     fmt.Printf("Employee now makes %f", e.salary)
}
```



Decide Employee Salary

In the above code, at **line 5**, we make a struct `employee` containing one field `salary` of type *float32*. Then, we have an important method `giveRaise()`. Look at the header of `giveRaise()` method at **line 11** as: `func (this *employee) giveRaise(pct float32)`. It updates the `salary` of employee `this` by adding the previous `salary` to the `salary` multiplied by `pct`.

Let's see the main function. In **line 17**, we make an employee `e` with `new()`. In the next line, we are assigning the value to the `salary` of `e`. At **line 20**, we are calling method `giveRaise()` on `e`. At **line 21**, we are printing the updated value of the `salary` of `e`.

That’s it about the solution. In the next lesson, you’ll study the variations of call methods.