面向对象编程2

作者：少林之巅

目录

方法的定义

1. 和其他语言不一样，Go的方法采用另外一种方式实现。

方法的定义

2. Go的方法是在函数前面加上一个接受者，这样编译器就知道这个方法属于哪个类型了

```
type A struct {

}

func (a A) Test(s string) {

}
```

Test的接受者，因此A这个对象有一个Test方法

通过a来访问A的实例中的成员变量，也就是struct中的字段

方法的定义

3. 可以为当前包内定义的任何类型增加方法

```
type int Integer   //Integer是int的别名

func (a Integer) Test(s string) {

}
```

通过a来访问Integer的实例中的成员变量，也就是int

Test的接受者，因此Integer这个对象有一个Test方法

函数和方法的区别

4. 函数不属于任何类型，方法属于特定的类型

值类型和指针类型

5. 指针类型作为接受者

```
type A struct {

}

func (a *A) Test(s string) {

}
```

通过a来访问A的实例中的成员变量，也就是struct中的字段

Test的接受者，因此A这个对象有一个Test方法

值类型和指针类型

6. 指针类型和值类型作为接受者的区别

值类型和指针类型

7. 什么时候用值类型/指针类型作为接受者?

A. 需要修改接受者中的值的时候

B. 接受者是大对象的时候，副本拷贝代价比较大

C. 一般来说，通常使用指针类型作为接受者

匿名字段与继承

8. 匿名结构体与继承

```
type Animal struct {
    Name            string
}
```

```
type People struct {
    Sex         string
    Age         int
    Animal      //or *Animal
}
```

匿名字段与继承

9. 多重继承与冲突解决

```
type Mother struct {
    Name            string
}
```

```
type Father struct {
    Name            string
}
```

```
type People struct {
    Sex         string
    Age         int
    *Mother
    *Father
}
```

结构体与json序列化

10.结构体序列化：结构体转成json数据格式

```
type People struct {
    Sex         string
    Age         int
    *Mother
    *Father
}
```

结构体与json序列化

11.结构体反序列化：json数据格式转成结构体

课后练习

1. 实现一个简单的学生管理系统，每个学生有分数、年级、性别、名字等字段，用户可以在控制台添加学生、修改学生信息、打印所有学生列表的功能。**使用面向对象的方式实现！**

```
package main
import (
    “fmt”
    “flag”
)
type Student struct {
    Username string
    Score float32
    Grade string
    Sex int
}

func main() {
}
```