函数命名类型

从前面章节知道可以使用 type NewType OldType 语法定义一种新类型，这种类型都是命名类型，同理可以使用该方法定义一种新类型：函数命名类型，简称函数类型。例如：

```
type NewFuncType FuncLiteral
```

依据 Go 语言类型系统的概念，NewFuncType 为新定义的函数命名类型，FuncLiteral 为函数字面量类型，FuncLiteral 为函数类型 NewFuncType 的底层类型。当然也可以使用 type 在一个函数类型中再定义一个新的函数类型，这种用法在语法上是允许的，但很少这么使用。例如：

```
type NewFuncType OldFuncType
```

函数签名

有了上面的基础，函数签名就比较好理解了，所谓“函数签名”就是“有名函数”或“匿名函数”的字面量类型。所以有名函数和匿名函数的函数签名可以相同，函数签名是函数的“字面量类型”，不包括函数名。

函数声明

Go 语言没有 C 语言中函数声明的语义，准确地说，Go 代码调用 Go 编写的函数不需要声明，可以直接调用，但 Go 调用汇编语言编写的函数还是要使用函数声明语句，示例如下。这里讨论的函数声明主要是为 4.1 节中接口的方法声明做铺垫。

```
//函数声明=函数名+函数签名

//函数签名
func (InputTypeList)OutputTypeList

//函数声明
func FuncName (InputTypeList)OutputTypeList
```

下面通过一个具体的示例来说明上述概念。

```
//有名函数定义，函数名是 add
//add 类型是函数字面量类型 func (int,int) int
func add(a,b int) int {
 return a+b
}

//函数声明语句，用于 Go 代码调用汇编代码
```