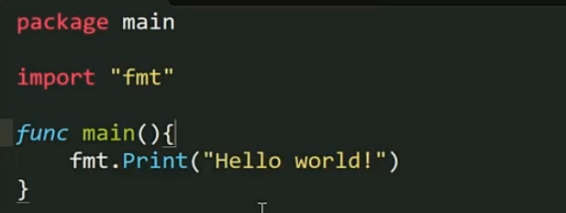
# 安装

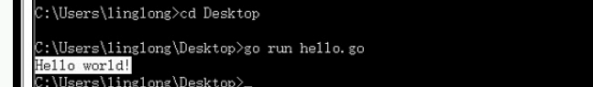


# 入门案例

## 代码

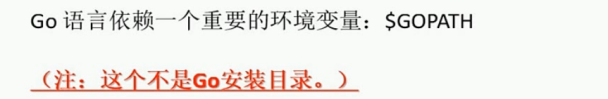


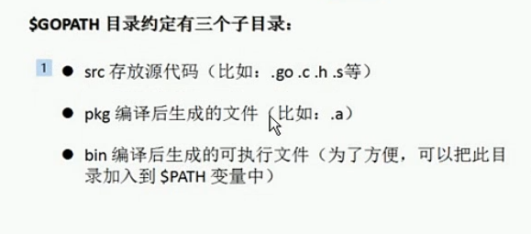
## 运行



# GOPATH

## 简介

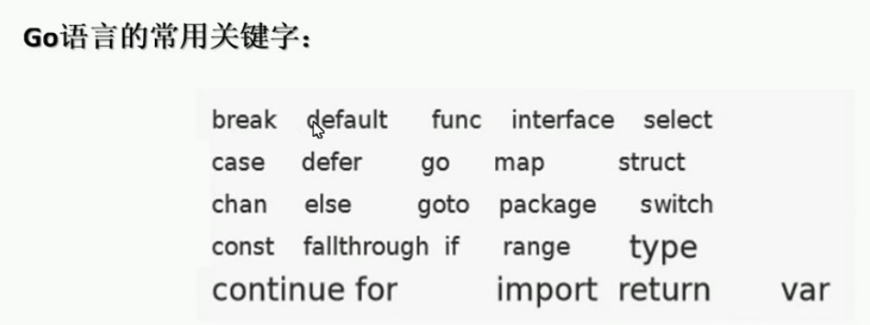




## Bee

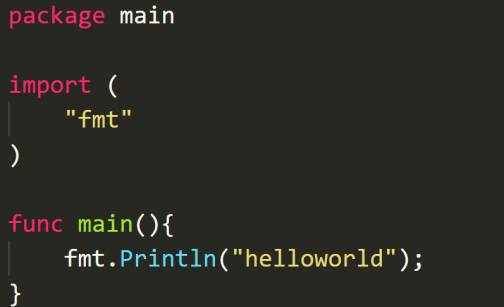


# 关键字

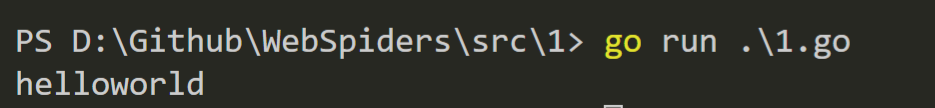


# 例子

## 程序

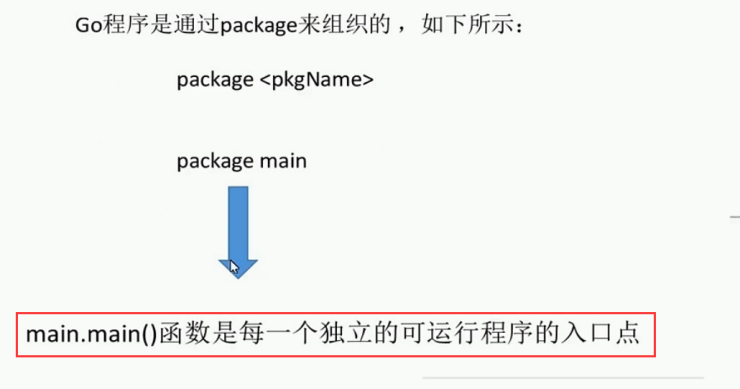


## 运行

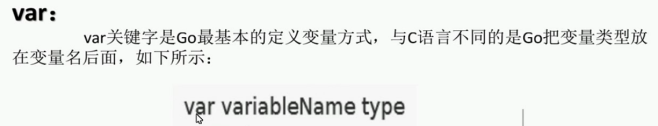


# 包

## 入口点



## Var



# 声明变量

## 普通声明



Print带f就是格式化

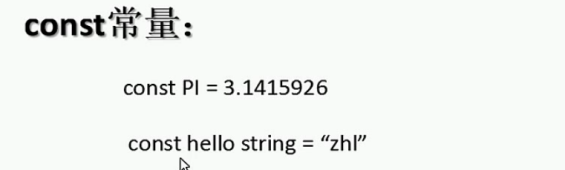
不带f的话就是输出

## 简短声明

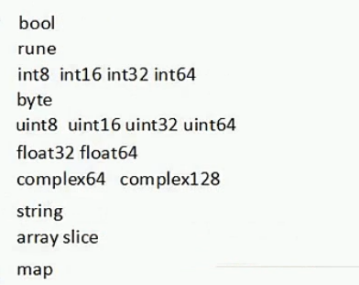
只能使用在函数的内部



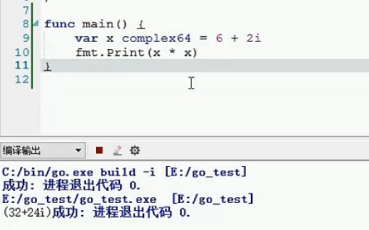
## const常量



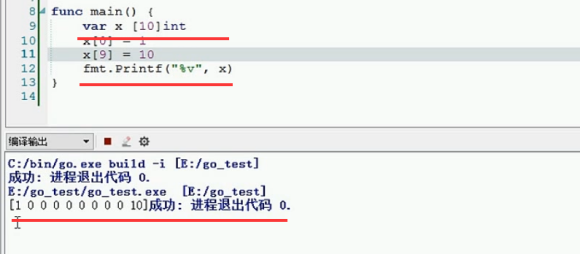
## 数据类型

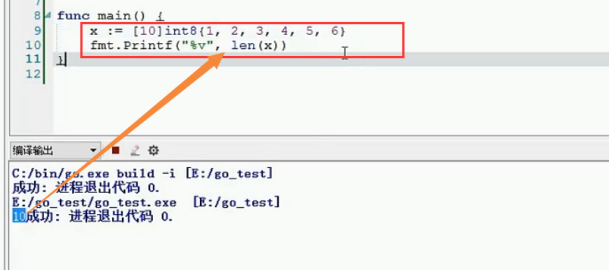


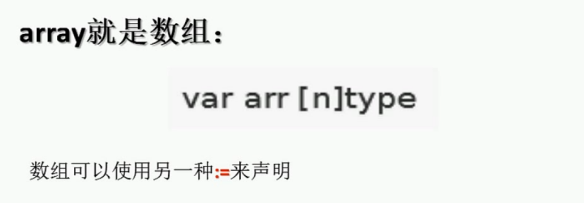
## 例子



# Array







# Slice

## 说明



第一个数字是长度len

第二个数字是容量cap cap是一个建议值

如果查过了cap golang会继续的进行扩展cap的容量

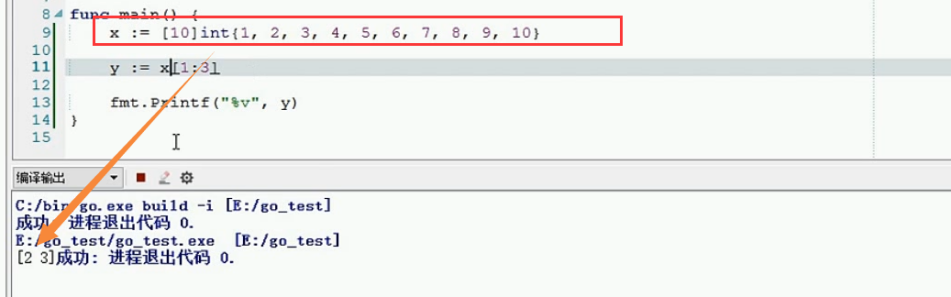
## 例子

开始的是正常的

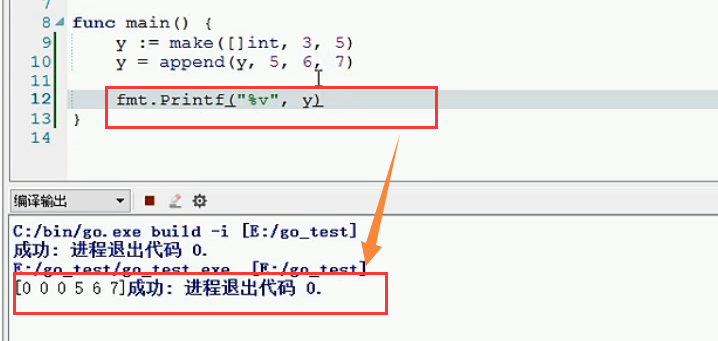
但是最终是按照个数来的

个数的计算后面的-前面的 就是（3-1=2）

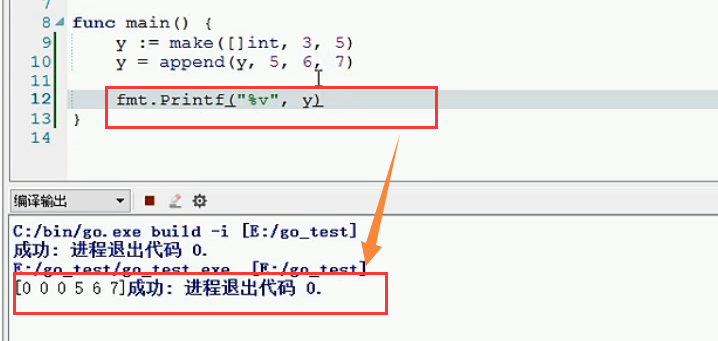
开始的位置是就是开始的 计算是从0开始的



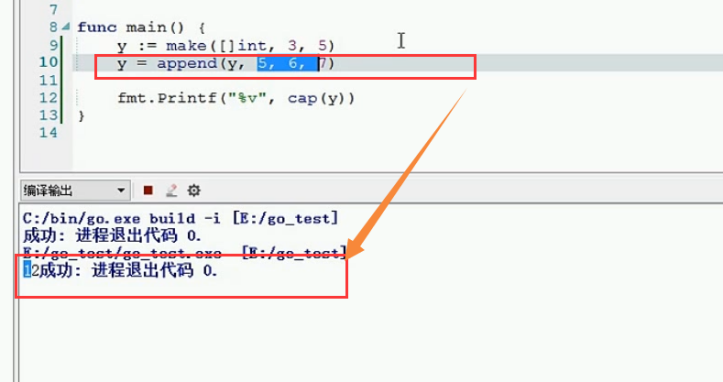
## Len

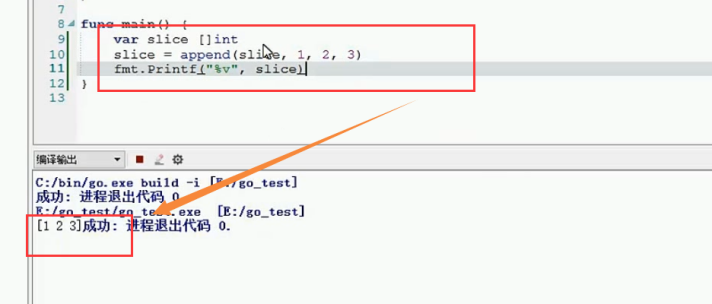


## 值的扩充

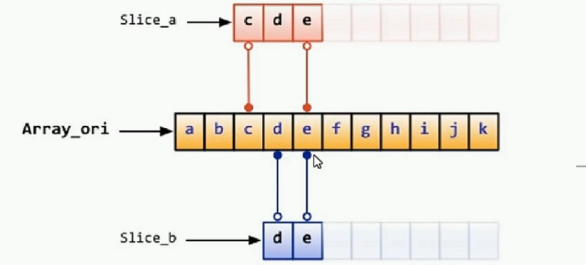


## cap扩充





## 图示

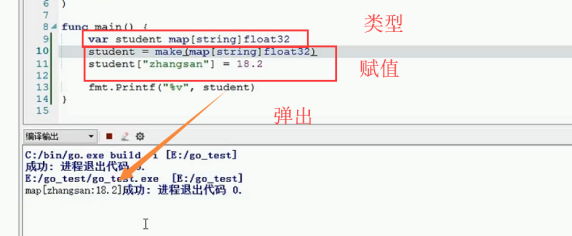


# Map

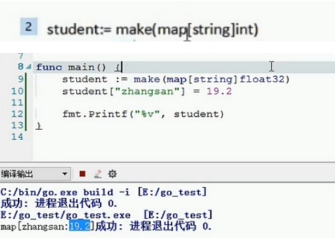
## 说明



## 例子



## 简短声明

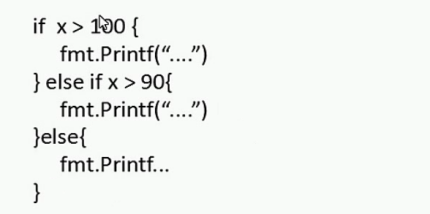


## 总结



# 流程控制

## If

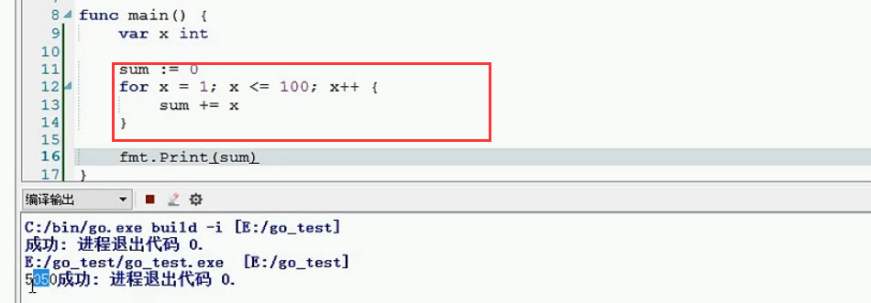


## For

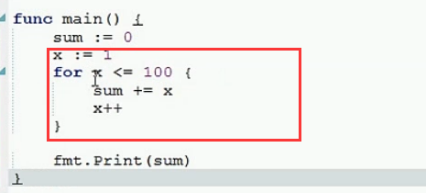
### 简述



### 例子

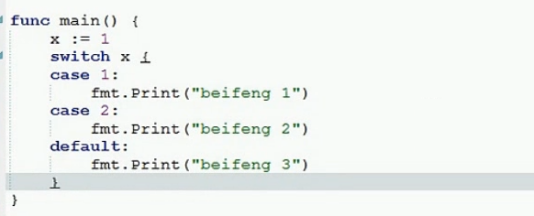


### 作为while使用



## Switch

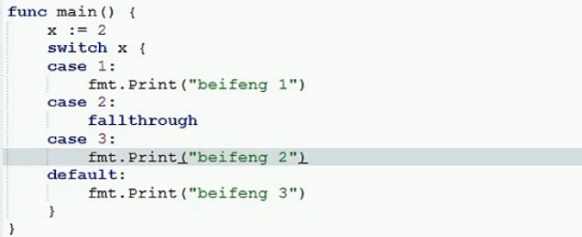
### 例子



### Fallthrough

他的意思就是继续往下走

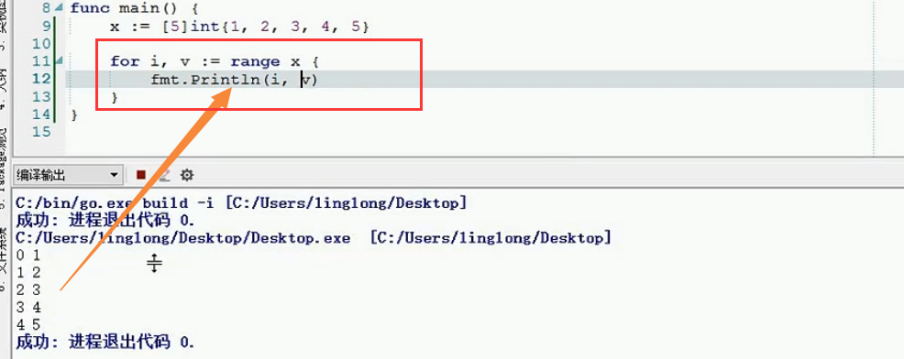
Break弹出

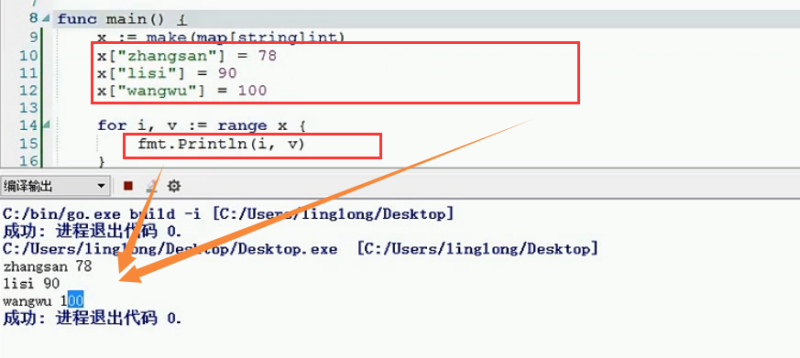


## Range

### 例子







### 占位符

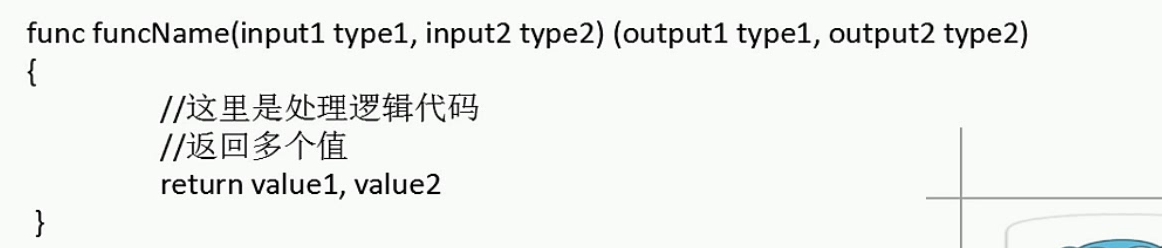


记住\n是换行

%c是字符

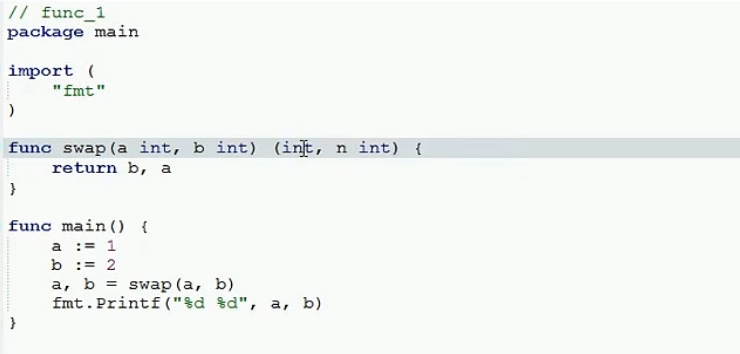
# 函数

## 简介



## 传值

### 值传递

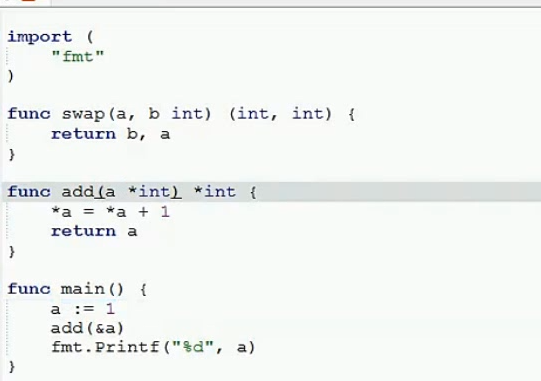


### 地址传递

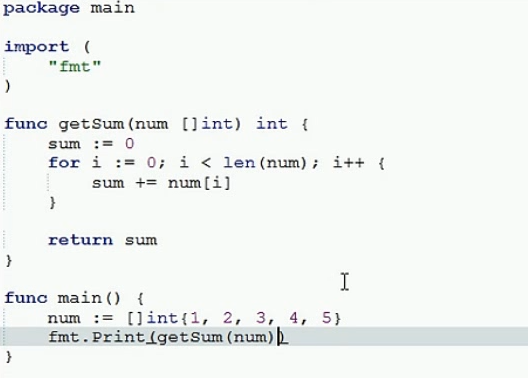
传递给子函数的为地址 所以采用&

子函数不能使用地址 只能使用值 所以采用\*

记住哦 采用了\*与& 地址对应的值都发生了变化



### 函数例子



Slice就是动态数组

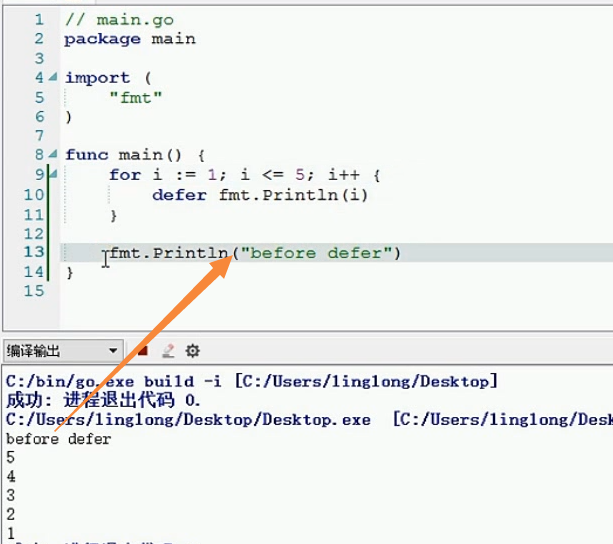
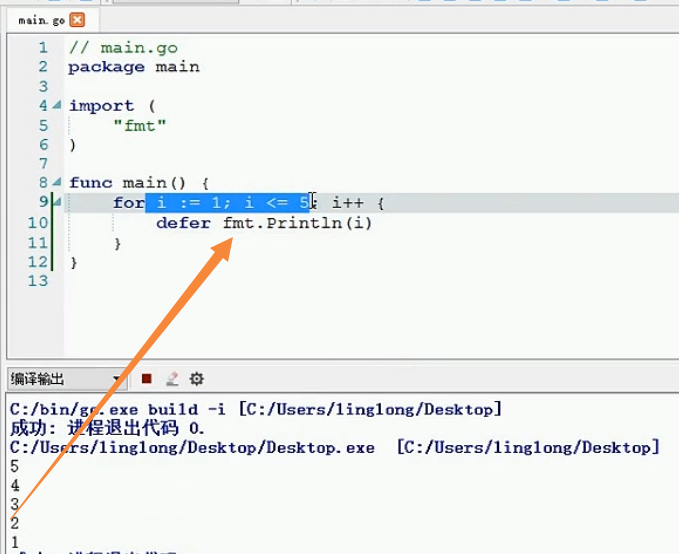
### 匿名函数



## Defer

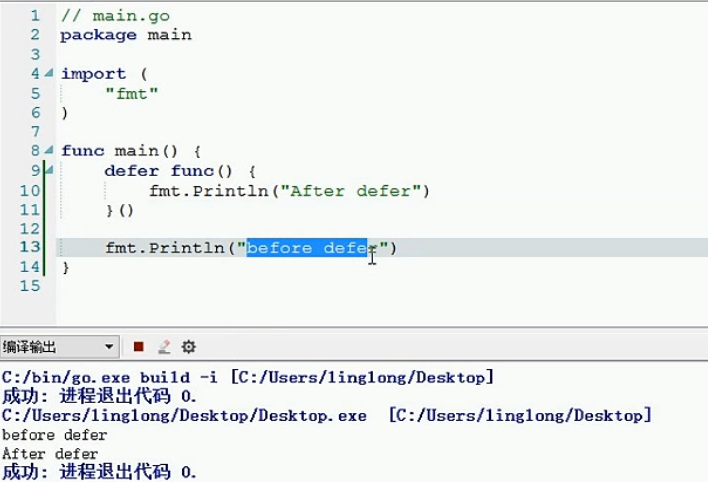
### 简述

defer就是在函数执行完毕之后才执行



### defer func

### 第一种写法



Defer func(){

}只是表明是一个函数体

加上()表明要执行这个函数体

### 第二种写法

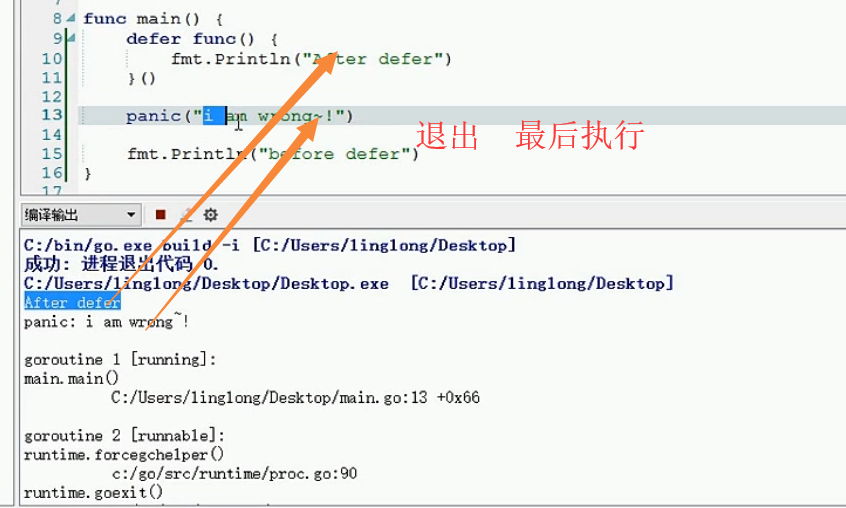


Defer close 达到一个安全执行的目的

## Panic

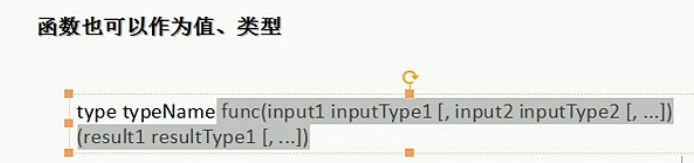
异常处理机制

在defer之后进行执行 在最后执行

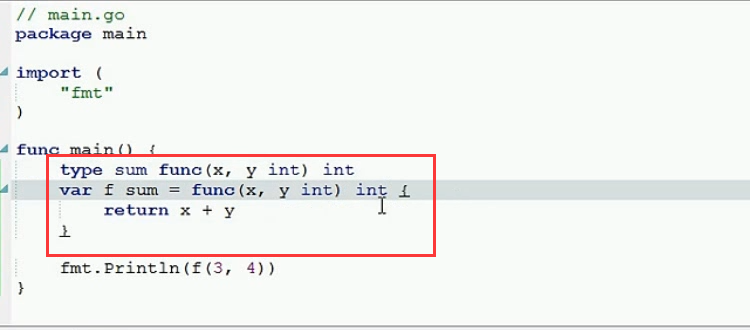


## 函数作为类型

### 简介



### 例子



# Struct

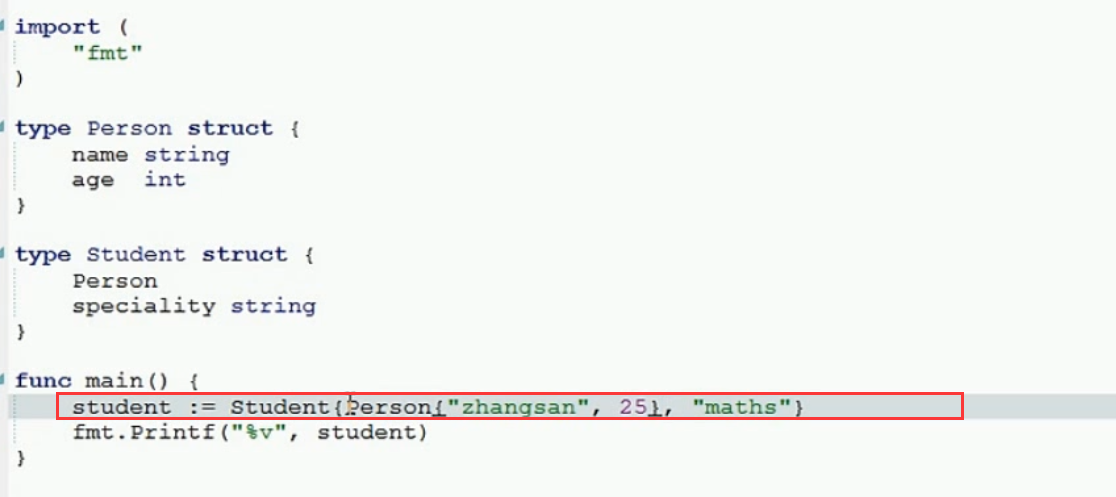
## 例子



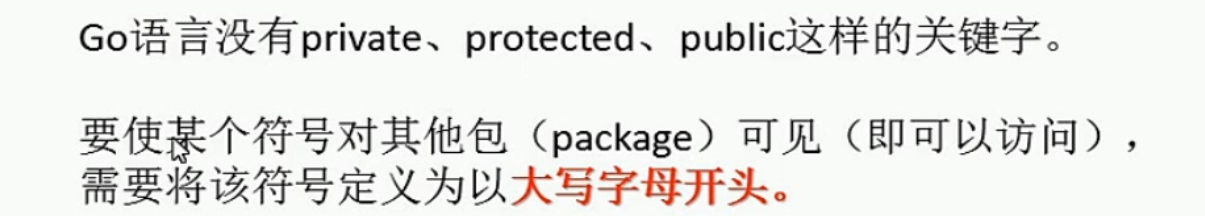


## 包含属性

注意应该是包含一层括号哦！！！



## 大小写



# OOP