打印

fmt.Println(err)

func Printf(format string, list of variables to be printed)

%t: 代表布尔

%d: 格式化整数

fmt.Printf()

fmt.Errorf()

定义变量 var

var name type = expression

var s string

var s <- 类型推断

短变量声明,不需要定义类型，用于局部

name := expression

t := 0.0

t=2 //重新赋值

指针

x :=1

p := &x &取地址

\*p \*获取值

new函数

表达式new(T) 创建一个未命名的T类型变量，初始化为T类型的零值，并返回其地址（地址类型为\*T）

p := new(int)

多重赋值

i,j,k = 2,3,5

\_, err = io.Copy(dst,src)

可以将不需要的值赋给空标识符

定义自己的类型

type name underlying-type

type Celsius float64

基本类型

基本类型：如int,float,bool,string

结构化的，如struct,array,切片slice,map,通道channel

描述类型的行为：如interface

1. 整数

符号整数：int8,int16,int32,int64,int

无符号整数：uint8,uint16,unint32,uint64,uint

1. 浮点数

float32

float64

布尔值 bool

true, false

! && ||

逻辑运算符

==， !=, <,<=,>,>=

字符串 类型 string

str=“string”

str[:] 下标读取

常量

const pi=

const (

e=

pi=

)

const (

a = iota

b = iota

c = iota

)

a=0，b=1,c=2

比较长度len(str)

复合类型

数组

var a [3]int

初始化

var q [3]int = [3]int{1,2}

q := [...]int{1,2,3}

for i,v := range a {

}

slice 类似于java list,可变序列，写成[]T，底层是数组

属性：指针，长度和容量

指针指向数组的第一个可以从slice中访问的元素

长度是指slice中的元素个数，len

容量大小是从slice的起始元素到底层数组的最后一个元素间元素的个数,cap

s[:] 特性和python一样，切片

判断是否为空 len(s)==0，而不是s == nil

创建一个具有指定元素类型，长度和容量的slice

make([]T,len)

make([]T,len,cap)

copy(dest,source) 拷贝数组

append:给slice添加元素

var x []int

x = append(x,1,2,3)

append(x,x...) 追加x中的所有元素

Map 类型是map[K]V , 拥有键值对元素的无序集合

创建map, 用make

ages := make(map[string]int)

ages := map[string]int{

“alice”: 31,

“charlie”:34,

}

设值 ages[“alice”] = 32

删除 delete(ages,”alice”)

for name, age := range ages{  
}

判断是否为空

map == nil

len(map) ==0

判断值是否存在

if age,ok :=ages[‘’bob’];!ok{}

ok为布尔值

结构体

多个任意类型的命令变量组合在一起的聚合数据类型

type Employee struct{

ID int

Name string

}

变量名称是首字母大写的，那么变量是可导出的。

结构体类型S不可以定义一个拥有相同结构体类型S的成员变量，但是可以定义一个S的指针类型，既\*S

赋值

Point{1，2} 结构体Point，用户X，Y,且有顺序

anim := git.GIF{LoopCount: nframes} 定义名称

匿名结构体

type Circle struct {

Point 匿名结构体写法， 这样可以直接调用Point的变量 #Point point 不匿名

Radius int

}

Json

Go对象转化为JSON还是从JSON转换为Go对象

json.Marshal 和unmarshal方法

函数

func name(parameter-list)(result-list){

body

}

parameter\_list 的形式为 (param1 type1, param2 type2, …)

return\_value\_list 的形式为 (ret1 type1, ret2 type2, …)

result-list可以是多个值，如（string,string）

init()函数，包会首先执行这个函数

错误

当函数调用发生错误时返回一个附加的结果作为错误值，习惯上讲错误值作为最后一个结果返回。如果错误只有一种情况，结果通常设置为布尔类型

value ,ok := cache.lookup(key)

Go语言通过使用普通的值而非异常来报告错误

函数变量

函数可以赋给变量

f :=square (square为函数)

f() 调用

匿名函数：func关键字后面没有函数的名称

ex:

func(r rune) rune {return r + 1}

defer 语句会将其后面跟随的语句进行延迟处理，跟在defer后面的语言，将会在程序进行最后的return之后再执行。

在defer 函数即将返回时，将延迟处理的语句按defer的逆序进行执行

方法

声明：

func (p Point) Distance(q Point) float64{  
 body

}

(p Point) 跟结构体绑定

然后p.Distance来调用

指针接收者的我方法

主调函数会复制每一个实参变量，如果一个实参太大，希望避免复制整个实参，必须使用指针类传递变量的地址

func (p \*Point) ScaleBy(factor float64)

首字母大写的标识符时可以从包中导出的，而首字母没有大写的则不导出，同样的机制也同样作用于结构体内的字段和类型中的方法。

Go语言中封装的单元式包而不是类型，无论实在函数内的代码还是方法内的代码，结构体类型内的字段对于同一个包中的所有代码都是可见的。

包名要小写

导入多个包, import， const, var,type都有这样的写法

import “fmt”

import “os”

import ( //因式分解的方式

“fmt”

“os”

)

import(“fmt”;”os”)

可见性规则

大写为可导出，小写不可导出

类型转换

不存在隐式类型转换，所以所有的转换都必须显示说明

valueOfTypeB = typeB(valueOfTypesA)

如 b := int(a)

类型别名

type TZ int : TZ 代替int

time包提供时间的处理