常量的定义

- 常量的值在编译时就已经确定
- 常量的定义格式与变量基本相同
- 等号右侧必须是常量或者常量表达式
- 常量表达式中的函数必须是内置函数

```
// 定义单个常量
const a int = 1
const b = 'A'
const (
    text   = "123"
    length = len(text)
    num    = b * 20
)

// 同时定义多个变量
const i, j, k = 1, "2", '3'
const (
    text2, length2, num2 = "456", len(text2), k * 10
)
```

```
1.  package main
2.
3.  import (
4.      "fmt"
5.  )
6.
7.  const (
8.      a = 1
9.      b
10.     c
11. )
12.
13. func main() {
14.     fmt.Println(a)
15.     fmt.Println(b)
16.     fmt.Println(c)
17.
18. }
```

```
1.  package main
2.
3.  import (
4.      "fmt"
5.  )
6.
7.  var sss = "123"
8.
9.  const (
```

```
10.     a = len(sss)
11.     b
12.     c
13. )
14. 
15. func main() {
16.     fmt.Println(a)
17.     fmt.Println(b)
18.     fmt.Println(c)
19. 
20. }
```

```
./temp.go:10: const initializer len(sss) is not a constant
./temp.go:11: const initializer len(sss) is not a constant
./temp.go:12: const initializer len(sss) is not a constant
错误: 进程退出代码 2.
```

常量的初始化必须是常量，所以说这个 sss 是全局变量，在编译的是并没有对它没有进行处理，所以说在编译的时候sss 是不存在的东西，所以说这个就不是常量的表达式。因为在编译的时候这个值没有办法复制给常量左边的变量。

```
1.  package main
2.  
3.  import (
4.      "fmt"
5.  )
6.  
7.  const (
8.      a = "123"
9.      b = len(a)
10.     c
11. )
12. 
13. func main() {
14.     fmt.Println(a)
15.     fmt.Println(b)
16.     fmt.Println(c)
17. 
18. }
```

```
/home/jiemin/code/GOlang/go/src/temp/temp  [/home/jiemin/code/GOlang/go/src/temp]
123
3
3
成功: 进程退出代码 0.
```

```
1.  package main
2.  
```

```go
import (
    "fmt"
)

const (
    a, b = 1, "2"
    c
)

func main() {
    fmt.Println(a)
    fmt.Println(b)
    fmt.Println(c)

}
```

```
/usr/local/go/bin/go build -i [/home/jiemin/code/GOlang/go/src/temp]
# _/home/jiemin/code/GOlang/go/src/temp
./temp.go:9: extra expression in const declaration
错误: 进程退出代码 2.
```

```go
package main

import (
    "fmt"
)

const (
    a, b = 1, "2"
    c, d
)

func main() {
    fmt.Println(a)
    fmt.Println(b)
    fmt.Println(c)
    fmt.Println(d)
}
```

```
/home/jiemin/code/GOlang/go/src/temp/temp  [/home/jiemin/code/GOlang/go/src/temp]
1
2
1
2
成功: 进程退出代码 0.
```

常量的初始化规则与枚举

- 在定义常量组时，如果不提供初始值，则表示将使用上行的表达式
- 使用相同的表达式不代表具有相同的值
- iota是常量的计数器，从0开始，组中每定义1个常量自动递增1
- 通过初始化规则与iota可以达到枚举的效果
- 每遇到一个const关键字，iota就会重置为0

```
const (
    // a与b都为"A"
    a = "A"
    b
    c = iota
    d // d的值为3
)

const (
    e = iota
    f // f的值为1
)
```

```
// 星期枚举
const (
    // 第一个常量不可省略表达式
    Monday = iota
    Tuesday
    Wednesday
    Thursday
    Friday
    Saturday
    Sunday
)
```

```
1.  package main
2.
3.  import (
4.      "fmt"
5.  )
6.
7.  const (
8.      a = 'A'
9.      b
10.     c = iota
11.     d
12. )
13.
14. func main() {
15.     fmt.Println(a)
16.     fmt.Println(b)
17.     fmt.Println(c)
18.     fmt.Println(d)
19. }
```

```
/home/jiemin/code/GOlang/go/src/temp/temp  [/home/jiemin/code/GOlang/go/src/temp]
65
65
2
3
成功: 进程退出代码 0.
```

```
1.  package main
2.
3.  import (
4.      "fmt"
```

```
)

const (
    a = 'A'
    b
    c = iota
    d
)

const (
    e = iota
)

func main() {
    fmt.Println(a)
    fmt.Println(b)
    fmt.Println(c)
    fmt.Println(d)
    fmt.Println(e)
}
```

iota关键字 只能在常量组中使用，每一个常量组的变量都是从零开始。

```
/home/jiemin/code/GOlang/go/src/temp/temp  [/home/jiemin/code/GOlang/go/src/temp]
65
65
2
3
0
成功: 进程退出代码 0.
```

编码规范命名：

常量的名称一般都要大写字母，golang可见性规则，首字母大写能被包的外部使用到。

```
package main

import (
    "fmt"
)

const (
    MAX_COUNT = 'A'
    b
    c = iota
    d
)

const (
    e = iota
)

func main() {
    fmt.Println(MAX_COUNT)
    fmt.Println(b)
    fmt.Println(c)
```

```
    fmt.Println(d)
    fmt.Println(e)
}
```

如果不想让包的外部使用到，那就把前面加上小写字母或者下划线_

```
package main

import (
    "fmt"
)

const (
    cMAX_COUNT = 'A'
)


func main() {
    fmt.Println(cMAX_COUNT)
}
```

或者

```
package main

import (
    "fmt"
)

const (
    _MAX_COUNT = 'A'
)


func main() {
    fmt.Println(_MAX_COUNT)
}
```

运算符

- Go中的运算符均是从左至右结合

优先级（从高到低）

- ^ !（一元运算符）
- * / % << >> & &^
- + - | ^（二元运算符）
- == != < <= >= >
- <-（专门用于channel）
- &&
- ||

一元运算符：

```go
package main

import (
    "fmt"
)

/*


 */

func main() {
    fmt.Println(^2)
}
```

```
/home/jiemin/code/GOlang/go/src/temp/temp [/home/jiemin/code/GOlang/go/src/temp]
-3
成功: 进程退出代码 0.
```

```go
package main

import (
    "fmt"
)

/*


 */

func main() {
    fmt.Println(!true)
}
```

```
/home/jiemin/code/GOlang/go/src/temp/temp [/home/jiemin/code/GOlang/go/src/temp]
false
成功: 进程退出代码 0.
```

```go
package main

import (
    "fmt"
)

/*


 */

func main() {
    fmt.Println(!false)
}
```

```
/home/jiemin/code/GOlang/go/src/temp/temp [/home/jiemin/code/GOlang/go/src/temp]
true
成功: 进程退出代码 0.
```

乘除加减取余，左移和右移

左移

```go
package main

import (
    "fmt"
)

/*


 */

func main() {
    fmt.Println(1 << 10)
}
```

```
/home/jiemin/code/GOlang/go/src/temp/temp [/home/jiemin/code/GOlang/go/src/temp]
1024
成功: 进程退出代码 0.
```

右移

```go
package main

```

```go
import (
    "fmt"
)

/*


 */

func main() {
    fmt.Println(1 >> 10)
}
```

```go
package main

import (
    "fmt"
)

/*


 */

func main() {
    fmt.Println(1 << 10 >> 10)
}
```

```
/home/jiemin/code/GOlang/go/src/temp/temp  [/home/jiemin/code/GOlang/go/src/temp]
1
成功: 进程退出代码 0.
```

```go
package main

import (
    "fmt"
)

/*

四个位运算符
6:  0110
11: 1011
-------------
&   0010        =2      //和运算——如果两个都是1的话，那就是1
|   1111        =15     //或运算——如果有一个是1的话，那就是1
^   1101        =13     //对比过程中俩位只有是一个是1的时候，它才是1
&^  0100        =4      //对比过程中，第二位数字这一位是1的话，那就把第一位数字同样这一位

*/

```

```
20. func main() {
21.     fmt.Println(6 & 11)
22.     fmt.Println(6 | 11)
23.     fmt.Println(6 ^ 11)
24.     fmt.Println(6 &^ 11)
25. }
```

```
/home/jiemin/code/GOlang/go/src/temp/temp  [/home/jiemin/code/GOlang/go/src/temp]
2
15
13
4
成功: 进程退出代码 0.
```

二元运算符

```
1.  package main
2.
3.  import (
4.      "fmt"
5.  )
6.
7.  /*
8.
9.
10.  */
11.
12. func main() {
13.     fmt.Println(1 ^ 2)
14. }
```

```
/home/jiemin/code/GOlang/go/src/temp/temp  [/home/jiemin/code/GOlang/go/src/temp]
3
成功: 进程退出代码 0.
```

&& 和 ||

&&——AND，需要满足全部条件才能执行，只要一个条件不满足，那就会跳出语句块。执行下面一语句

```
1.  package main
2.
3.  import (
4.      "fmt"
5.  )
6.
7.  func main() {
8.      a := 1
9.      if (10 / a) > 1 {
10.         fmt.Println("OK")
11.     }
```

```
12.
13. }
```

```
/home/jiemin/code/GOlang/go/src/temp/temp  [/home/jiemin/code/GOlang/go/src/temp]
OK
成功: 进程退出代码 0.
```

```
1.  package main
2.
3.  import (
4.      "fmt"
5.  )
6.
7.  func main() {
8.      a := 1
9.      if a > 0 && (10/a) > 1 {
10.         fmt.Println("OK")
11.     }
12. }
```

```
/home/jiemin/code/GOlang/go/src/temp/temp  [/home/jiemin/code/GOlang/go/src/temp]
OK
成功: 进程退出代码 0.
```

如果a小于0

```
1.  package main
2.
3.  import (
4.      "fmt"
5.  )
6.
7.  func main() {
8.      a := 1
9.      if a < 0 && (10/a) > 1 {
10.         fmt.Println("OK")
11.     }
12. }
```

就没有执行后面的条件，直接退出了

```
/usr/local/go/bin/go build -i [/home/jiemin/code/GOlang/go/src/temp]
成功: 进程退出代码 0.
/home/jiemin/code/GOlang/go/src/temp/temp  [/home/jiemin/code/GOlang/go/src/temp]
成功: 进程退出代码 0.
```

||——OR，只要条件满足一个，就可以执行语句块

```
1.  package main
2.
3.  import (
```

```
    "fmt"
)

func main() {
    a := 1
    if a < 0 || (10/a) > 1 {
        fmt.Println("OK")
    }
}
```

```
/usr/local/go/bin/go build -i [/home/jiemin/code/GOlang/go/src/temp]
成功: 进程退出代码 0.
/home/jiemin/code/GOlang/go/src/temp/temp  [/home/jiemin/code/GOlang/go/src/temp]
OK
成功: 进程退出代码 0.
```

```
package main

import (
    "fmt"
)

func main() {
    a := 1
    if a > 0 || (10/a) > 1 {
        fmt.Println("OK")
    }
}
```

```
/usr/local/go/bin/go build -i [/home/jiemin/code/GOlang/go/src/temp]
成功: 进程退出代码 0.
/home/jiemin/code/GOlang/go/src/temp/temp  [/home/jiemin/code/GOlang/go/src/temp]
OK
成功: 进程退出代码 0.
```

课堂作业

- 请尝试结合常量的iota与<<运算符实现计算机储存单位的枚举

```
const (
    _           = iota
    KB float64 = 1 << (iota * 10)
    MB
    GB
    TB
    PB
    EB
    ZB
    YB
)
```

```
1024
1.048576e+06
1.073741824e+09
1.099511627776e+12
1.125899906842624e+15
1.152921504606847e+18
1.1805916207174113e+21
1.2089258196146292e+24
```

作业：

```go
package main

import (
    "fmt"
)

const (
    B float64 = 1 << (iota * 10)
    KB
    MB
    GB
    TB
    PB
    EB
    ZB
    YB
)

func main() {
    fmt.Println(B)
    fmt.Println(KB)
    fmt.Println(MB)
    fmt.Println(GB)
    fmt.Println(TB)
    fmt.Println(PB)
    fmt.Println(EB)
    fmt.Println(ZB)
    fmt.Println(YB)
}
```

```
/home/jiemin/code/GOlang/go/src/temp/temp  [/home/jiemin/code/GOlang/go/src/temp]
1
1024
1.048576e+06
1.073741824e+09
1.099511627776e+12
1.125899906842624e+15
1.152921504606847e+18
1.1805916207174113e+21
1.2089258196146292e+24
成功: 进程退出代码 0.
```