

## 指针

Go虽然保留了指针，但与其它编程语言不同的是，在Go当中不支持指针运算以及“->”运算符，而直接采用“.”选择符来操作指针目标对象的成员

- 操作符“&”取变量地址，使用“\*”通过指针间接访问目标对象
- 默认值为 nil 而非 NULL

## 递增递减语句

在Go当中，++ 与 -- 是作为语句而并不是作为表达式

列子1：

```
1. package main
2.
3. import (
4.     "fmt"
5. )
6.
7. func main() {
8.     a := 1
9.     var p *int = &a
10.    fmt.Println(p)
11.    fmt.Println(*p)
12. }
```

列子2：

```
1. package main
2.
3. import (
4.     "fmt"
5. )
6.
7. func main() {
8.     a := 1
9.     a++
10.    var p *int = &a
11.    b := 1
12.    b--
13.    var f *int = &b
14.    fmt.Println(p)
15.    fmt.Println(*p)
16.    fmt.Println(f)
```

```
17.     fmt.Println(*f)
18. }
```

## 判断语句if

- 条件表达式没有括号
- 支持一个初始化表达式（可以是并行方式）
- 左大括号必须和条件语句或else在同一行
- 支持单行模式
- 初始化语句中的变量为block级别，同时隐藏外部同名变量
- 1.0.3版本中的编译器BUG

```
func main() {
    a := true
    if a, b, c := 1, 2, 3; a+b+c > 6 {
        fmt.Println("大于6")
    } else {
        fmt.Println("小于等于6")
        fmt.Println(a)
    }
    fmt.Println(a)
}
```

if 语句列子1：

```
1.  package main
2.
3.  import (
4.      "fmt"
5.  )
6.
7.  func main(){
8.      a := 10
9.      if a := 1; a > 0 {
10.         fmt.Println(a)
11.     }
12.     fmt.Println(a)
13. }
```

if 语句列子2：

```
1.  package main
2.
3.  import (
4.      "fmt"
5.  )
6.
7.  func main() {
8.      if 7 % 2 == 0 {
9.         fmt.Println("7 is even.")
10.     } else {
11.         fmt.Println("7 is odd.")

```

```

12.     }
13.
14.     if 8 % 4 == 0 {
15.         fmt.Println("8 is divisible by 4")
16.     }
17.
18.     if num := 9; num < 0 {
19.         fmt.Println(num, "is negative")
20.     } else if num < 10 {
21.         fmt.Println(num, "has 1 digit")
22.     } else {
23.         fmt.Println(num, "has multiple digits")
24.     }
25. }

```

## 循环语句for

- Go只有for一个循环语句关键字，但支持3种形式
- 初始化和步进表达式可以是多个值
- 条件语句每次循环都会被重新检查，因此不建议在条件语句中使用函数，尽量提前计算好条件并以变量或常量代替
- 左大括号必须和条件语句在同一行

```

func main() {
    a := 1
    for {
        a++
        if a > 3 {
            break
        }
    }
    fmt.Println(a)
}

```

```

func main() {
    a := 1
    for a <= 3 {
        a++
    }
    fmt.Println(a)
}

```

```

func main() {
    a := 1
    for i := 0; i < 3; i++ {
        a++
    }
    fmt.Println(a)
}

```

for 语句列子1:

```

1. package main
2.
3. import (
4.     "fmt"
5. )
6.
7. func main() {
8.     a := 1
9.     for {
10.         a++
11.         if a > 3 {
12.             break
13.         }
14.         fmt.Println(a)
15.     }
16.     fmt.Println("Over")
17. }

```

for 语句列子2：

```
1. package main
2.
3. import (
4.     "fmt"
5. )
6.
7. func main() {
8.     a := 1
9.     for a <= 3 {
10.         a++
11.         fmt.Println(a)
12.     }
13.     fmt.Println("Over")
14. }
```

for 语句列子3：

```
1. package main
2.
3. import (
4.     "fmt"
5. )
6.
7. func main() {
8.     a := 1
9.     for i := 0; i < 3; i++ {
10.         a++
11.         fmt.Println(a)
12.     }
13.     fmt.Println("Over")
14. }
```

for 语句列子4：

```
1. package main
2.
3. import (
4.     "fmt"
5. )
6.
7. func main() {
8.     a := "string"
9.     l := len(a)
10.    for i := 0; i < l; i++ {
11.        // a++
12.        fmt.Println(a)
13.    }
14.    fmt.Println("Over")
15. }
```

## 选择语句switch

- 可以使用任何类型或表达式作为条件语句
- 不需要写break，一旦条件符合自动终止
- 如希望继续执行下一个case，需使用fallthrough语句
- 支持一个初始化表达式（可以是并行方式），右侧需跟分号
- 左大括号必须和条件语句在同一行

```
func main() {  
    a := 1  
    switch a {  
    case 0:  
        fmt.Println("a=0")  
    case 1:  
        fmt.Println("a=1")  
    }  
    fmt.Println(a)  
}
```

```
func main() {  
    a := 1  
    switch {  
    case a >= 0:  
        fmt.Println("a=0")  
        fallthrough  
    case a >= 1:  
        fmt.Println("a=1")  
    }  
    fmt.Println(a)  
}
```

```
func main() {  
    switch a := 1; {  
    case a >= 0:  
        fmt.Println("a=0")  
        fallthrough  
    case a >= 1:  
        fmt.Println("a=1")  
    }  
}
```

switch 语句例子1：

```
1. package main  
2.  
3. import (  
4.     "fmt"  
5. )  
6.  
7. func main() {  
8.     a := 1  
9.     switch a {  
10.    case 0:  
11.        fmt.Println("a == 0")  
12.    case 1:  
13.        fmt.Println("a == 1")  
14.    default:  
15.        fmt.Println("None")  
16.    }  
17. }
```

switch 语句例子2：

```
1. package main  
2.  
3. import (  
4.     "fmt"  
5. )  
6.  
7. func main() {  
8.     a := 1  
9.     switch {  
10.    case a >= 0:  
11.        fmt.Println("a == 0")
```

```

12.     case a >= 1:
13.         fmt.Println("a == 1")
14.     default:
15.         fmt.Println("None")
16.     }
17. }

```

switch 语句列子3：

```

1.  package main
2.
3.  import (
4.      "fmt"
5.  )
6.
7.  func main() {
8.      a := 1
9.      switch {
10.     case a >= 0:
11.         fmt.Println("a == 0")
12.         fallthrough
13.     case a >= 1:
14.         fmt.Println("a == 1")
15.     default:
16.         fmt.Println("None")
17.     }
18. }

```

switch 语句列子4：

```

1.  package main
2.
3.  import (
4.      "fmt"
5.  )
6.
7.  func main() {
8.      // a := 1
9.      switch a := 1; {
10.     case a >= 0:
11.         fmt.Println("a == 0")
12.         fallthrough
13.     case a >= 1:
14.         fmt.Println("a == 1")
15.     default:
16.         fmt.Println("None")
17.     }
18. }

```

switch 语句列子5：

```

1.  package main
2.

```

```
3. import (
4.     "fmt"
5. )
6.
7. func main() {
8.     a := 10
9.     switch a := 1; {
10.    case a >= 0:
11.        fmt.Println("a == 0")
12.        fallthrough
13.    case a >= 1:
14.        fmt.Println("a == 1")
15.    default:
16.        fmt.Println("None")
17.    }
18.    fmt.Println(a)
19. }
```

```
1. package main
2.
3. import (
4.     "fmt"
5. )
6.
7. func main() {
8.     // a := 10
9.     switch a := 1; {
10.    case a >= 0:
11.        fmt.Println("a == 0")
12.        fallthrough
13.    case a >= 1:
14.        fmt.Println("a == 1")
15.    default:
16.        fmt.Println("None")
17.    }
18.    // fmt.Println(a)
19. }
```

## 跳转语句goto, break, continue

- 三个语法都可以配合标签使用
- 标签名区分大小写，若不使用会造成编译错误
- Break与continue配合标签可用于多层循环的跳出
- Goto是调整执行位置，与其它2个语句配合标签的结果并不相同

```
func main() {
    LABEL:
    for {
        for i := 0; i < 10; i++ {
            if i > 2 {
                break LABEL
            } else {
                fmt.Println(i)
            }
        }
    }
}
```

```
func main() {
    LABEL:
    for i := 0; i < 10; i++ {
        for {
            fmt.Println(i)
            continue LABEL
        }
    }
}
```

### break 语句

break 语句例子1：

```
1. package main
2.
3. import (
4.     "fmt"
5. )
6.
7. func main() {
8.     LABEL1:
9.     for {
10.        for i := 0; i < 10; i++ {
11.            if i > 3 {
12.                break LABEL1
13.            }
14.        }
15.    }
16.    fmt.Println("OK")
17. }
```

### goto语句

goto 语句例子1：

```
1. package main
2.
3. import (
4.     "fmt"
5. )
6.
7. func main() {
8.     for {
9.        for i := 0; i < 10; i++ {
```



```

10.         if i > 3 {
11.             goto LABEL1
12.         }
13.     }
14. }
15. LABEL1:
16.     fmt.Println("OK")
17. }

```

## continue 语句

continue 语句例子1：

```

1. package main
2.
3. import (
4.     "fmt"
5. )
6.
7. func main() {
8. LABEL1:
9.     for i := 0; i < 10; i++ {
10.        for {
11.            continue LABEL1
12.            fmt.Println(i)
13.        }
14.    }
15.    fmt.Println("OK")
16. }

```

## 课堂作业

- 将下图中的continue替换成goto，程序运行的结果还一样吗？
- 请尝试并思考为什么。

```

func main() {
LABEL:
    for i := 0; i < 10; i++ {
        for {
            fmt.Println(i)
            continue LABEL
        }
    }
}

```

Goto是调整执行位置

作业：

```

1. package main
2.
3. import (

```

```

4.     "fmt"
5. )
6.
7. func main() {
8. LABEL1:
9.     for i := 0; i < 10; i++ {
10.        for {
11.            fmt.Println(i)
12.            continue LABEL1
13.        }
14.    }
15. }

```

**/home/jiemin/code/GOlang/go/src/break/break** **[/home/jiemin/code/GOlang/go/src/break]**

```

0
1
2
3
4
5
6
7
8
9

```

**成功: 进程退出代码 0.**

把continue更换为goto

```

1. package main
2.
3. import (
4.     "fmt"
5. )
6.
7. func main() {
8. LABEL1:
9.     for i := 0; i < 10; i++ {
10.        for {
11.            fmt.Println(i)
12.            goto LABEL1
13.        }
14.    }
15. }

```

结果是死循环，一直打印0数值

把标签放到下面

```

1. package main
2.
3. import (
4.     "fmt"
5. )
6.
7. func main() {

```

```
8.     for i := 0; i < 10; i++ {  
9.         for {  
10.             fmt.Println(i)  
11.             goto LABEL1  
12.         }  
13.     }  
14. LABEL1:  
15.     fmt.Println("OK")  
16. }
```

**/home/jiemin/code/GOlang/go/src/break/break** [/home/jiemin/code/GOlang/go/src/break]

0

OK

**成功: 进程退出代码 0.**