Map类型

作者：少林之巅

目录

map声明和定义

1. map类型是一个key-value的数据结构。

```
//var a map[key的类型]value类型
var a map[string]int
var b map[int]string
var c map[float32]string
```

注意：*map*必须初始化才能使用，否则*panic*

map声明和定义

2. map类型的变量默认初始化为nil，需要使用make分配map内存

```go
package main

import (
    "fmt"
)

func main() {
    var a map[string]int
    if a == nil {
        fmt.Println("map is nil. Going to make one.")
        A = make(map[string]int)
    }
}
```

map基本操作

3. map插入操作

```go
package main

import (
    "fmt"
)

func main() {
    a := make(map[string]int)
    a["steve"] = 12000
    a["jamie"] = 15000
    a["mike"] = 9000
    fmt.Println(“a map contents:", a)
}
```

map基本操作

4. 声明时进行初始化

```
package main

import (
    "fmt"
)

func main() {
    a := map[string]int {
        "steve": 12000,
        "jamie": 15000,
    }
    a[“mike"] = 9000
    fmt.Println(“a map contents:", a)
}
```

map基本操作

5. 通过key访问map中的元素

```go
package main

import (
    "fmt"
)

func main() {
    a := map[string]int{
        "steve": 12000,
        "jamie": 15000,
    }
    a["mike"] = 9000
    b := "jamie"
    fmt.Println("Salary of", b, "is", a[b])
}
```

map基本操作

5. 通过key访问map中的元素

```
package main

import (
    "fmt"
)

func main() {
    a := map[string]int{
        "steve": 12000,
        "jamie": 15000,
    }
    a["mike"] = 9000
    b := “123”
    fmt.Println("Salary of", b, "is", a[b])
}
```

map基本操作

6. 如何判断map指定的key是否存在？ value, ok := map[key]

```go
package main

import (
    "fmt"
)

func main() {
    a := map[string]int{
        "steve": 12000,
        "jamie": 15000,
    }
    a["mike"] = 9000
    b := "joe"
    value, ok := a[b]
    if ok == true {
        fmt.Println("Salary of", b, "is", value)
    } else {
        fmt.Println(b,”not found")
    }
}
```

map基本操作

7. map遍历操作

```go
package main

import (
    "fmt"
)

func main() {
    a := map[string]int{
        "steve": 12000,
        "jamie": 15000,
    }
    a["mike"] = 9000
    fmt.Println("All items of a map")
    for key, value := range a {
        fmt.Printf("personSalary[%s] = %d\n", key, value)
    }

}
```

map基本操作

8. map删除元素

```go
package main

import (
    "fmt"
)

func main() {
    a := map[string]int{
        "steve": 12000,
        "jamie": 15000,
    }
    a["mike"] = 9000
    fmt.Println("map before deletion", a)
    delete(a, "steve")
    fmt.Println("map after deletion", a)

}
```

map基本操作

9. map的长度

```go
package main

import (
	"fmt"
)

func main() {
	a := map[string]int{
		"steve": 12000,
		"jamie": 15000,
	}
	a["mike"] = 9000
	fmt.Println("length is", len(a))

}
```

map基本操作

10.map是引用类型

```
package main

import (
    "fmt"
)

func main() {
    a := map[string]int{
        "steve": 12000,
        "jamie": 15000,
    }
    a["mike"] = 9000
    fmt.Println(“origin map", a)
    b := a
    b["mike"] = 18000
    fmt.Println(“a map changed", a)

}
```

map 进行排序

11.默认情况下，map并不是按照key有序进行遍历的

```
package main

import (
    "fmt"
)

func main() {
    var a map[string]int = make(map[string]int, 10)
    for i := 0; i < 10; i++ {
        key := fmt.Sprintf("key%d", i)
        a[key] = i
    }

    for key, value := range a {
        fmt.Printf("key:%s = %d\n", key, value)
    }
}
```

map 进行排序

12. map按照key进行排序，遍历

```
package main
import (
    "fmt"
    "sort"
)
func main() {
    var a map[string]int = make(map[string]int, 10)
    for i := 0; i < 10; i++ {
        key := fmt.Sprintf("key%d", i)
        a[key] = i
    }
    var keys []string
    for key, _ := range a {
        keys = append(keys, key)
    }
    sort.Strings(keys)
    for _, key := range keys {
        fmt.Printf("key:%s=%d\n", key, a[key])
    }
}
```

map 进行排序

13. map类型的切片

```go
package main

import (
	"fmt"
)

func main() {
	var mapSlice []map[string]int
	mapSlice = make([]map[string]int, 5)
	fmt.Println("before map init")
	for index, value := range mapSlice {
		fmt.Printf("index:%d value:%v\n", index, value)
	}

	fmt.Println()
	mapSlice[0] = make(map[string]int, 10)
	mapSlice[0]["a"] = 1000
	mapSlice[0]["b"] = 2000
	mapSlice[0]["c"] = 3000
	mapSlice[0]["d"] = 4000
	mapSlice[0]["e"] = 5000

	fmt.Println("after map init")
	for index, value := range mapSlice {
		fmt.Printf("index:%d value:%v\n", index, value)
	}
}
```

课后练习

1. 写一个程序，统计一个字符串每个单词出现的次数。比如： s = “how do you do”
   输出 how = 1 do = 2 you = 1

2. 写一个程序，实现学生信息的存储，学生有id、年龄、分数等信息。需要非常方便的通过id查找到对应学生的信息。