Homework Chapter 1

1. Go 语言的变量命名规范是什么？

   - 常量: 大写字母使用下划线分词
   - 变量: 驼峰命名. 数字, 字母, 下划线等, 不推荐下划线, 数字不能在变量前

     首字母大写: 包外可见

     首字母小写: 包内可见

2. 如何获取变量的地址？ 如何取到地址中的值？

   取地址: `&variable`

   取地址值: `*variable`

3. 变量的生命周期是什么？作用域是什么？

   变量所在的代码块为此变量的作用域, 变量进入代码块的开始到退出为此变量的生命周期.

4. 创建变量有哪几种方式

   i. 长声明 `var variable TYPE` 初始化 `variable = value`
   ii. 长声明 + 初始化 `var variable = value`
   iii. 短声明 + 初始化 `variable := value`
   iv. 多个变量声明...

```
var(
variable1 = value1
variable2 = value2
)
```

   v. 多个常量声明...

```
const(
VARIABLE1 = value1
```

```
VARIABLE2 = value2
)
```

vi. 对于引用类型变量 map, slice, chan

```
m := make(map([string]sting), [cap])
var s = make([]sting, [len, cap])
var c chan int= make(chan int, [cap])
```

vii. new

5. Go 语言简单数据类型都有哪些？
   - 整型

     有符号: int, int8, int16, int32, int64

     无符号: uint, uint8, uint16, uint32, uint64, byte
   - 浮点型

     float32, float64
   - 字符型

     byte
   - 字符串

     string
   - 布尔型

     bool
6. 初始化数组的几种方式？
   i. 指定大小+赋值 `variable := [3]int{1, 2, 3}`

   ii. 指定大小+不赋值 `variable := [3]int{}`

   iii. 推断大小 `variable := [...]int{1, 2, 3}`
7. 遍历数组
   i. for

```
for i := 0; i < len(array); i++ {
    // do something
}
```

ii. for range

```
for idx, value := range array {
    // do something
}
```

8. 初始化切片的几种方式？

   i. 变量声明 `var sclice []int`

   ii. 字面量 `var slice = []int{1, 2, 3}`

   iii. 从数组初始化 `slice := array[start:end[:max]]` cap = max - start

   iv. make `slice := make([]int, [len, cap])`

9. 遍历切片

   i. for

```
for i := 0; i < len(slice); i++ {
    // do something
}
```

   ii. for range

```
for idx, value := range slice {
    // do something
}
```

10. 如何复制切片

    `copy(destnation, source)`

11. 实现切片的增删改查

    - 增 `slice := append(slice, interface...)`
    - 删 `slice := append(start:targetIdx, targetIdx+1:end...)`

- 改 `slice[targetIdx]` = value
- 查 `slice[targetIdx]`

12. 面代码是否有问题？并说出为什么？ 如果有问题，如何修正？

```
s :=[]string{"炭烤生蚝","麻辣小龙虾","干锅鸭"}
s2 := make([]*string,len(s))
for i,v:=range s {
    s2[i]=&v
}
```

有问题, s2 中存储均为同一个地址. 原因是 for 语句中, 会将每一个值复制一份给 v, 而 v 的地址在 for 语句开始到结束都是不变的.

解决方法:

```
s :=[]string{"炭烤生蚝","麻辣小龙虾","干锅鸭"}
s2 := make([]*string,len(s))
for i,v:=range s {
    v := v
    s2[i]=&v
}
```

13. 分别写一个 if 和 switch、枚举 的例子

- if

```
if mark >= 90 {
    fmt.Println("Excellent")
} else if mark > 60 && mark < 90 {
    fmt.Println("Good")
} else {
    fmt.Println("failed")
}
```

- switch

```
swich {
case mark >= 90:
    fmt.Println("Excellent")
case mark >= 60 && mark < 90:
    fmt.Println("Good")
```

```
case mark < 60:
    fmt.Println("failed")
}
```

- enum

```
const(
    MONDAY      = iota + 1
    TUESDAY
    WEDNESDAY
    THURSDAY
    FRIDAY
    SATURDAY
    SUNDAY
)
```

14. map 有什么特点？

    i. 无序

    ii. 不可比较

    iii. k-v 对 不可寻址

    iv. 使用 make 初始化

15. 什么样的类型可以做 map 的 key

    可比较的类型都可以做 map 的 key eg. 基本值类型, struct, array, channel

16. 写一个 map 的增删改查

    - 增 `m[key] = value`
    - 删 `delete(m, key)`
    - 改 `m[key] = newValue`
    - 查

```
if _, ok := m[key]; !ok {
    // not founded
    return
} // founded
```

17. 函数的定义

```
func funcName(parametersList) (returnList) {
    // function body
}
```

18. 函数传参，传值还是传引用？

    i. 取决于实参类型, 实参为基本类型, struct 等时, 传值. 实参为引用类型, 如 map, slice, chan 等, 传引用

    ii. 取决于 parametersList 的要求, go 语言会自动解引用与引用

19. 定义函数的多返回值？

```
func MutiReturns(a int, b int) (sum int, diff int) {
    return a + b, a - b
}
```

20. 举例说明 函数变量、匿名函数、闭包、变长函数？

    - 函数变量

        i. 变量可以存储函数的地址

```
f := MutiReturns
fmt.Println(f(1, 2))
// output: 3 -1
```

        ii. 可以当作函数的参数

```
sum, diff := MutiReturns(
    func() int { return 1 }(),
    func() int { return 2 }())

fmt.Println(sum, diff)
// output: 3 -1
```

    - 匿名函数

        - 声明

```
func (parametersList) (returnList) {
    // function body
```

```
}
```

- 调用

```
func (parametersList) (returnList) {
    // function body
}()
```

- 闭包

一个函数的返回值是一个函数, 返回的函数使用了非自己函数块定义的变量.

```
func makeSuffix(suffix string) func(str string) string {
    return func(str string) string {
        if !strings.HasSuffix(str, suffix) {
            return str + suffix
        }
        return str
    }
}

func main() {
    checkSuffix := makeSuffix(".txt")
    file := checkSuffix("rey")
    fmt.Println(file)
    // output: rey.txt
}
```

- 变长函数

```


func MutiSummation(nums ...int) (sum int) {
    for _, v := range nums {
        sum += v
    }
    return
}

func main() {
    sum := MutiSummation(1, 2, 3)
    fmt.Println(sum)
    // output: 6
}
```

21. 说一下面向对象设计的好处？

    高聚合, 低耦合, 高复用, 隐藏实现细节 user-friendly

22. 方法的定义

```
func (receiver) funcName(parametersList) (returnList) {
    // function body
}
```

23. 指针接收者和值接收者有何不同?

    值接收者, 传值. 指针接收者, 传引用.

Coding By Rey

mail: 3065588496@qq.com