# 作业参考

1、打印九九乘法表。如果可以要求间隔均匀。

```
1*1=1
1*2=2 2*2=4
1*3=3 2*3=6  3*3=9
1*4=4 2*4=8  3*4=12 4*4=16
1*5=5 2*5=10 3*5=15 4*5=20 5*5=25
1*6=6 2*6=12 3*6=18 4*6=24 5*6=30 6*6=36
1*7=7 2*7=14 3*7=21 4*7=28 5*7=35 6*7=42 7*7=49
1*8=8 2*8=16 3*8=24 4*8=32 5*8=40 6*8=48 7*8=56 8*8=64
1*9=9 2*9=18 3*9=27 4*9=36 5*9=45 6*9=54 7*9=63 8*9=72 9*9=81
```

```go
func multiplyTable() {
    var width int
    for i := 1; i < 10; i++ {
        for j := 1; j <= i; j++ {
            if j == 1 {
                width = 2
            } else {
                width = 3
            }
            // fmt.Printf("%d*%d=%-[4]*[3]d", j, i, i*j, width)
            fmt.Printf("%d*%d=%-[3]*d", j, i, width, j*i) // 靠近d就是值索引，宽度和精度要在后面加*，如果用[3]，那么，后面就直接表示3+1
        }
        fmt.Println()
    }
}
```

```go
func multiplyTable () {
    var width int
    for i := 1; i < 10; i++ {
        for j := 1; j <= i; j++ {
            if j == 1 {
                width = 2
            } else {
                width = 3
            }
            fmt.Printf("%d*%d=%-[4]*[3]d", j, i, i*j, width)
            if i == j {
                fmt.Println()
            }
        }
    }
}
```

2、随机生成100以内的20个非0正整数，打印出来。对生成的数值，第单数个（不是索引）累加求和，第偶数个累乘求积。打印结果

```go
package main

import (
    "fmt"
    "math/rand"
    "time"
)

func main() {
    r := rand.New(rand.NewSource(time.Now().UnixNano()))
    var (
        sum             = 0
        product uint64 = 1 // 极小情况下，可能超过int上界，使用uint64合适
    )

    for i := 0; i < 20; i++ {
        v := r.Intn(99) + 1
        fmt.Println(v)
        if i&1 == 0 {
            sum += v
        } else {
            product *= uint64(v)
        }
    }
    fmt.Println(sum, product)
}
```

3、打印100以内的斐波那契数列

```go
// 100以内斐波那契数列
a, b := 1, 1
limit := 100
fmt.Println(a, b)
for {
    b, a = a+b, b
    if b >= limit {
        break
    }
    fmt.Println(b)
}
```