平时练习Go的代码

1.go实现递归：

```
package main

import "fmt"

func main() {
    var result = fibonacci(10)
    fmt.Printf("fibonacci(%d) is: %d\n", 10, result)

}

func fibonacci(n int) (res int) {
    if n <= 0 {
        res = 0
    }else {
        res = n + fibonacci(n-1)
    }
    return
}
```

2.go实现回调

```
package main

import "fmt"

func main()  {
    callback(1,Add)
}

func Add(a,b int)  {
    fmt.Printf("The sum of %d and %d is: %d\n", a, b, a+b)
}

func callback(y int,f func(int,int))  {
    f(y,2)
}
```

3.函数调用

```
package main

import "fmt"

func main() {
    var f = Adder2()
    fmt.Print(f(1), " - ")
    fmt.Print(f(20), " - ")
    fmt.Print(f(300))
}

func Adder2() func(int) int {
    var x int
    return func(delta int) int {
        x += delta
        return x
    }
}
```

4.把函数付给变量

```
package main

import "fmt"

func main() {
    // make an Add2 function, give it a name p2, and call it:
    p2 := Add2()
    fmt.Printf("Call Add2 for 3 gives: %v\n", p2(3))
    // make a special Adder function, a gets value 3:
    TwoAdder := Adder(2)
    fmt.Printf("The result is: %v\n", TwoAdder(3))
}

func Add2() func(b int) int {
    return func(b int) int {
        return b + 2
    }
}

func Adder(a int) func(b int) int {
    return func(b int) int {
        return a + b
    }
}
```

5.切片实例

```
package main
import "fmt"

func main() {
    var arr1 [6]int
    var slice1 []int = arr1[2:5] // item at index 5 not included!

    // load the array with integers: 0,1,2,3,4,5
    for i := 0; i < len(arr1); i++ {
        arr1[i] = i
    }

    // print the slice
    for i := 0; i < len(slice1); i++ {
        fmt.Printf("Slice at %d is %d\n", i, slice1[i])
    }

    fmt.Printf("The length of arr1 is %d\n", len(arr1))
    fmt.Printf("The length of slice1 is %d\n", len(slice1))
    fmt.Printf("The capacity of slice1 is %d\n", cap(slice1))

    // grow the slice
    slice1 = slice1[0:4]
    for i := 0; i < len(slice1); i++ {
        fmt.Printf("Slice at %d is %d\n", i, slice1[i])
    }
    fmt.Printf("The length of slice1 is %d\n", len(slice1))
    fmt.Printf("The capacity of slice1 is %d\n", cap(slice1))

    // grow the slice beyond capacity
    //slice1 = slice1[0:7 ] // panic: runtime error: slice bound out of range
}
```