



# 如何编写结构清晰的代码

---



张华

64174234@qq.com

# 内容

---

1.1 问题的提出

1.2 问题的抽象与量化

1.3 求解流程

1.4 实现

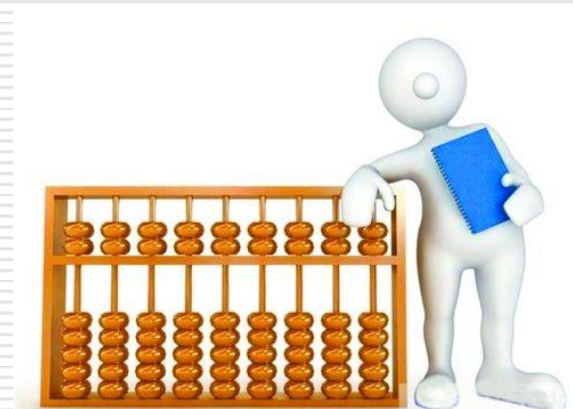
1.5 编译

1.6 总结

1.7 思考

# 1.1 问题的提出

- 婚庆酒杯塔布置，计算所需酒杯的数量
- 已知单根钢管的重量，计算每捆钢管的重量
- 已知某市2015年用于“校校通”工程的经费为500万元，为了保证工程的顺利实施，计划到2025，每年比上1年多投入50万元，计算这10年的总投入经费是多少？



# 1.1 问题的提出(续)

---

□ 你能说出这是一类什么问题吗？即问题的本质是什么？

■ 等差数列求和



# 1.2 问题的抽象与量化

---

## □ 量化

- 首项为 $a$ ，数据类型为整型
- 项数为 $n$ ，数据类型为整型
- 公差为 $d$ ，数据类型为整型
- 结果为 $s$ ，数据类型为整型

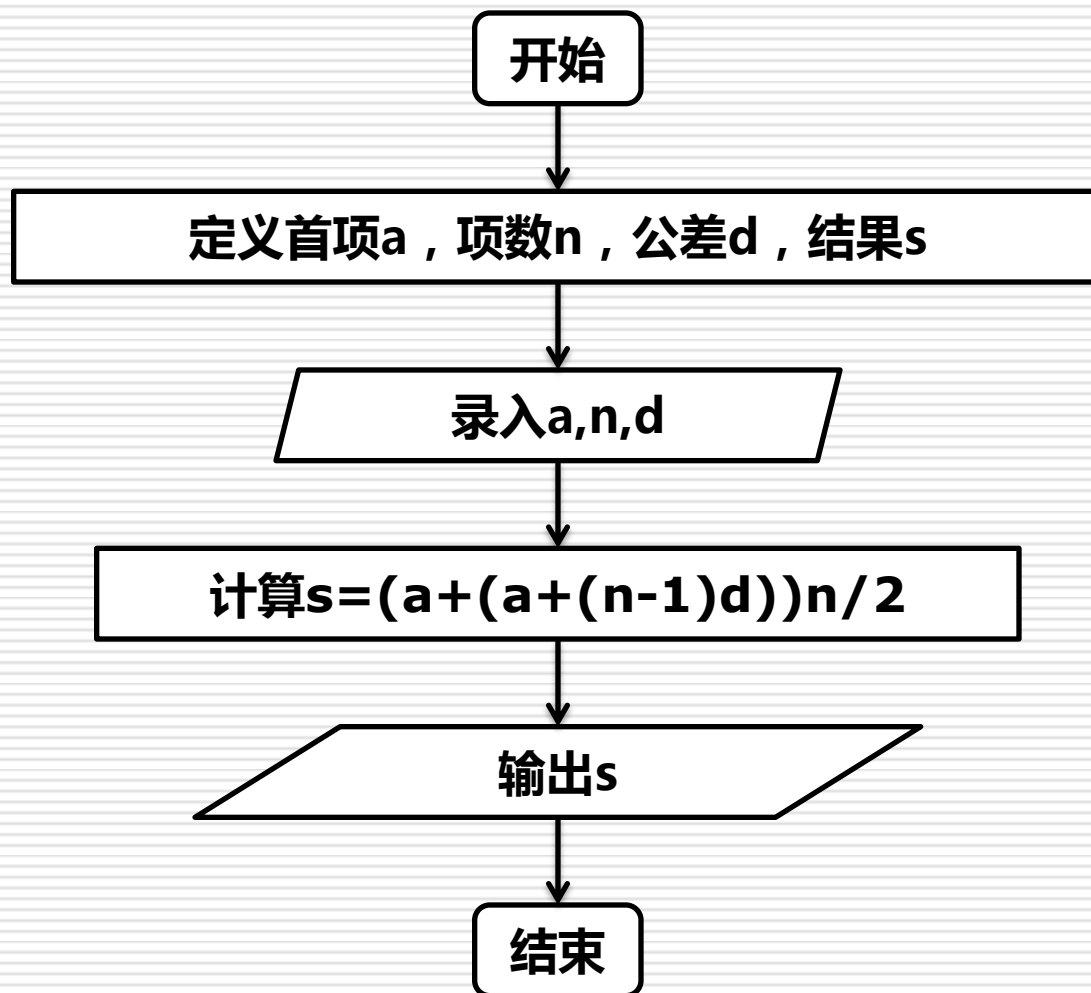
## □ 抽象

- 等差数列求和公式 $n*a + n*(n-1)*d/2$

## 1.3 求解流程(续)

### □ 一体化求解流程

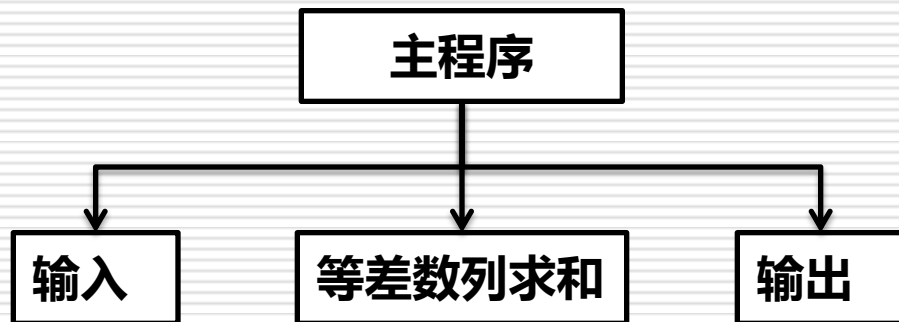
- 不便于复用，不适合于复杂问题求解



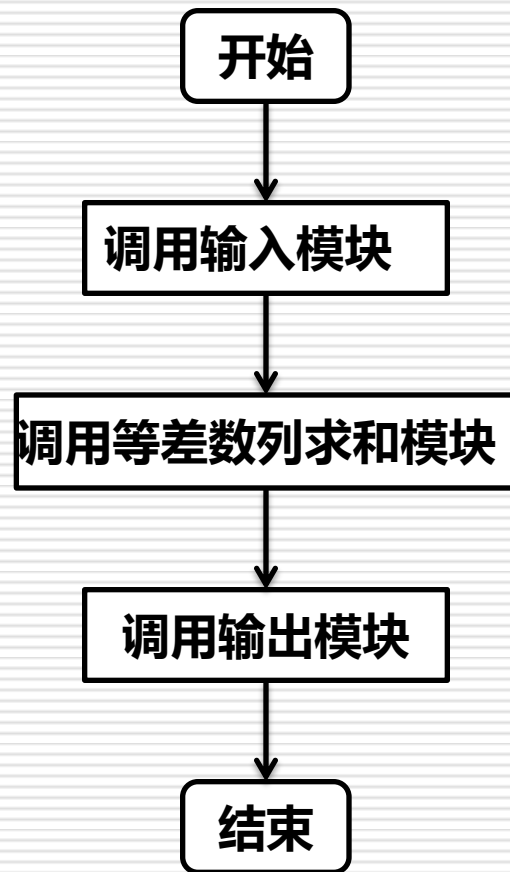
## 1.3 求解流程(续)

### □ 基于模块化编程思维的流程

#### ■ 代码结构清晰



功能结构

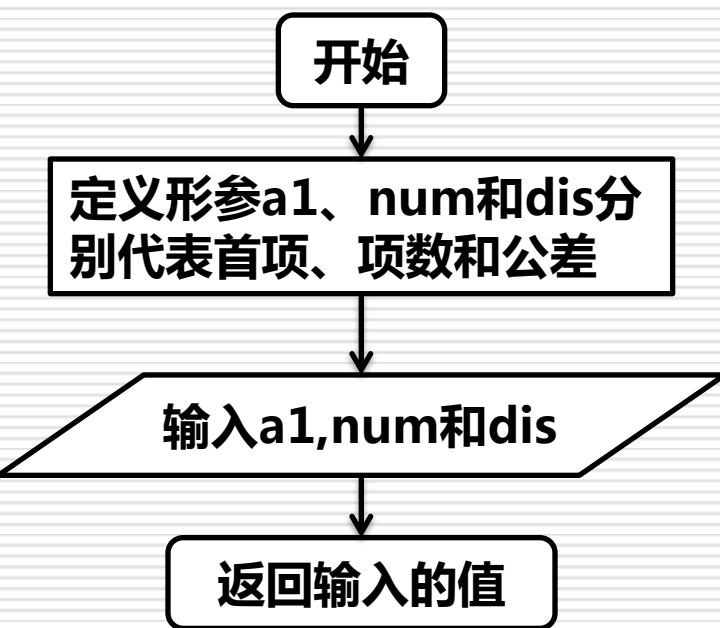


主程序流程

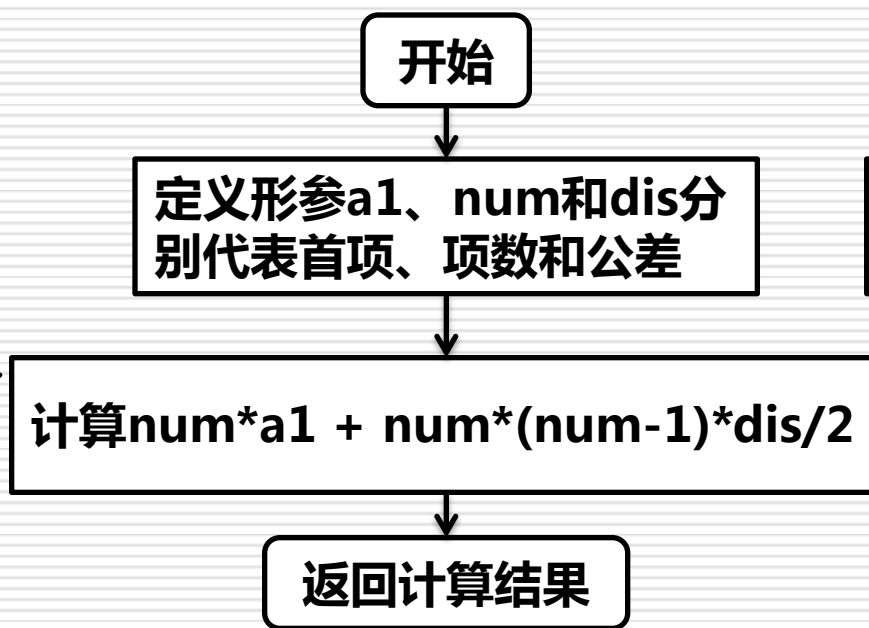
# 1.3 求解流程(续)

## □ 基于模块化编程思维的流程

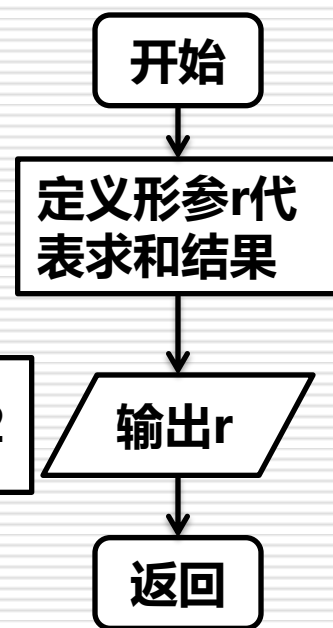
### ■ 结构清晰



输入模块流程



等差数列求和模块流程

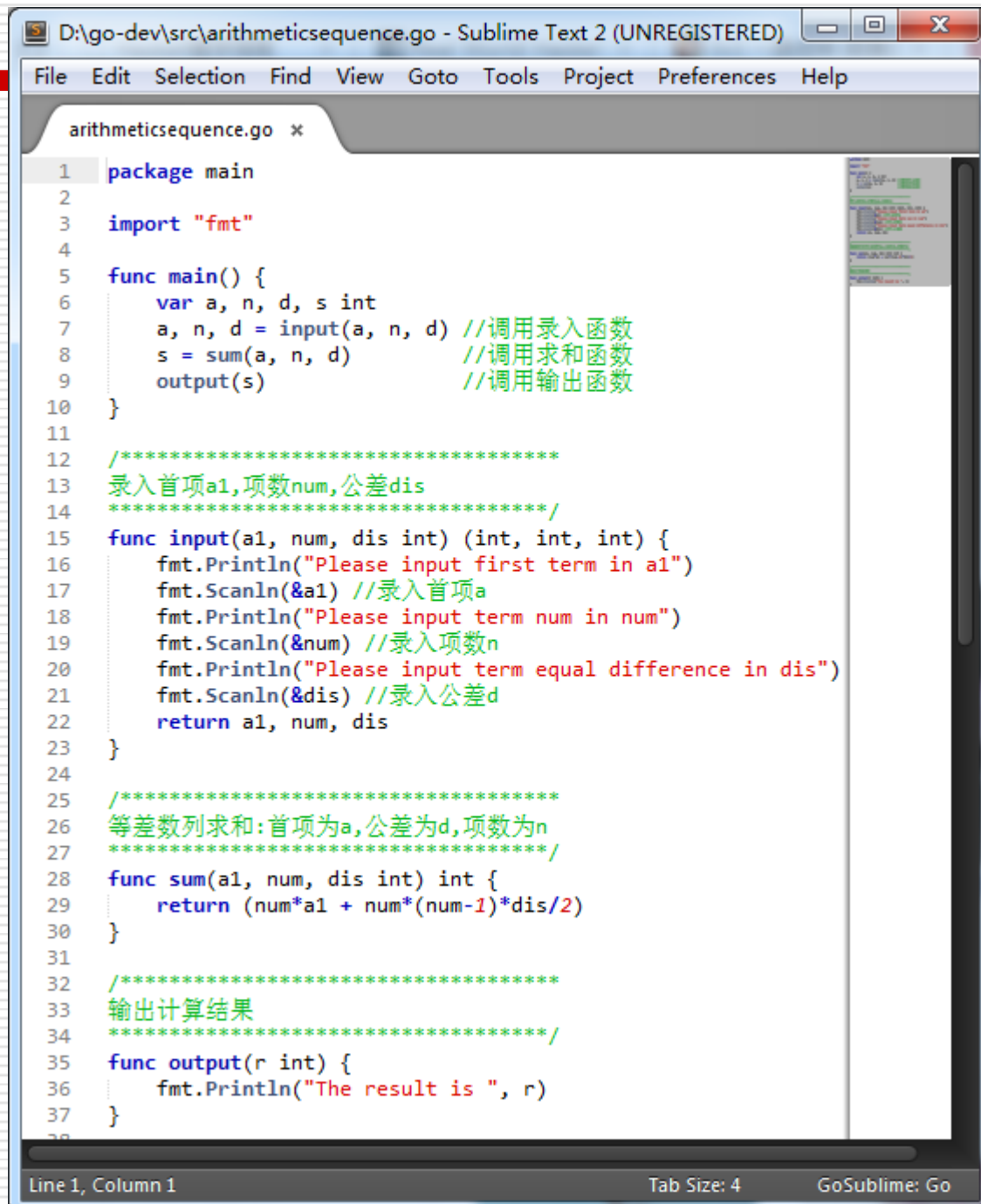


输出模块流程



# 1.4 实现

## □ 代码全貌



```
D:\go-dev\src\arithmeticsequence.go - Sublime Text 2 (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

arithmeticsequence.go x
1 package main
2
3 import "fmt"
4
5 func main() {
6     var a, n, d, s int
7     a, n, d = input(a, n, d) //调用录入函数
8     s = sum(a, n, d)         //调用求和函数
9     output(s)               //调用输出函数
10 }
11
12 /*****
13 录入首项a1,项数num,公差dis
14 *****/
15 func input(a1, num, dis int) (int, int, int) {
16     fmt.Println("Please input first term in a1")
17     fmt.Scanln(&a1) //录入首项a
18     fmt.Println("Please input term num in num")
19     fmt.Scanln(&num) //录入项数n
20     fmt.Println("Please input term equal difference in dis")
21     fmt.Scanln(&dis) //录入公差d
22     return a1, num, dis
23 }
24
25 /*****
26 等差数列求和:首项为a,公差为d,项数为n
27 *****/
28 func sum(a1, num, dis int) int {
29     return (num*a1 + num*(num-1)*dis/2)
30 }
31
32 /*****
33 输出计算结果
34 *****/
35 func output(r int) {
36     fmt.Println("The result is ", r)
37 }
38 }
```

Line 1, Column 1 Tab Size: 4 GoSublime: Go

## 1.4 实现(续)

### □ 主函数

```
1  package main
2
3  import "fmt"
4
5  func main() {
6      var a, n, d, s int
7      a, n, d = input(a, n, d) //调用录入函数
8      s = sum(a, n, d)         //调用求和函数
9      output(s)               //调用输出函数
10 }
```

## 1.4 实现(续)

### □ 录入函数

```
12  /*****
13  录入首项a1,项数num,公差dis
14  *****/
15  func input(a1, num, dis int) (int, int, int) {
16      fmt.Println("Please input first term in a1")
17      fmt.Scanln(&a1) //录入首项a
18      fmt.Println("Please input term num in num")
19      fmt.Scanln(&num) //录入项数n
20      fmt.Println("Please input term equal difference in dis")
21      fmt.Scanln(&dis) //录入公差d
22      return a1, num, dis
23  }
```

## 1.4 实现(续)

### □ 等差数列求和函数

```
25  /*******  
26  等差数列求和:首项为a,公差为d,项数为n  
27  *****/  
28  func sum(a1, num, dis int) int {  
29      return (num*a1 + num*(num-1)*dis/2)  
30  }
```

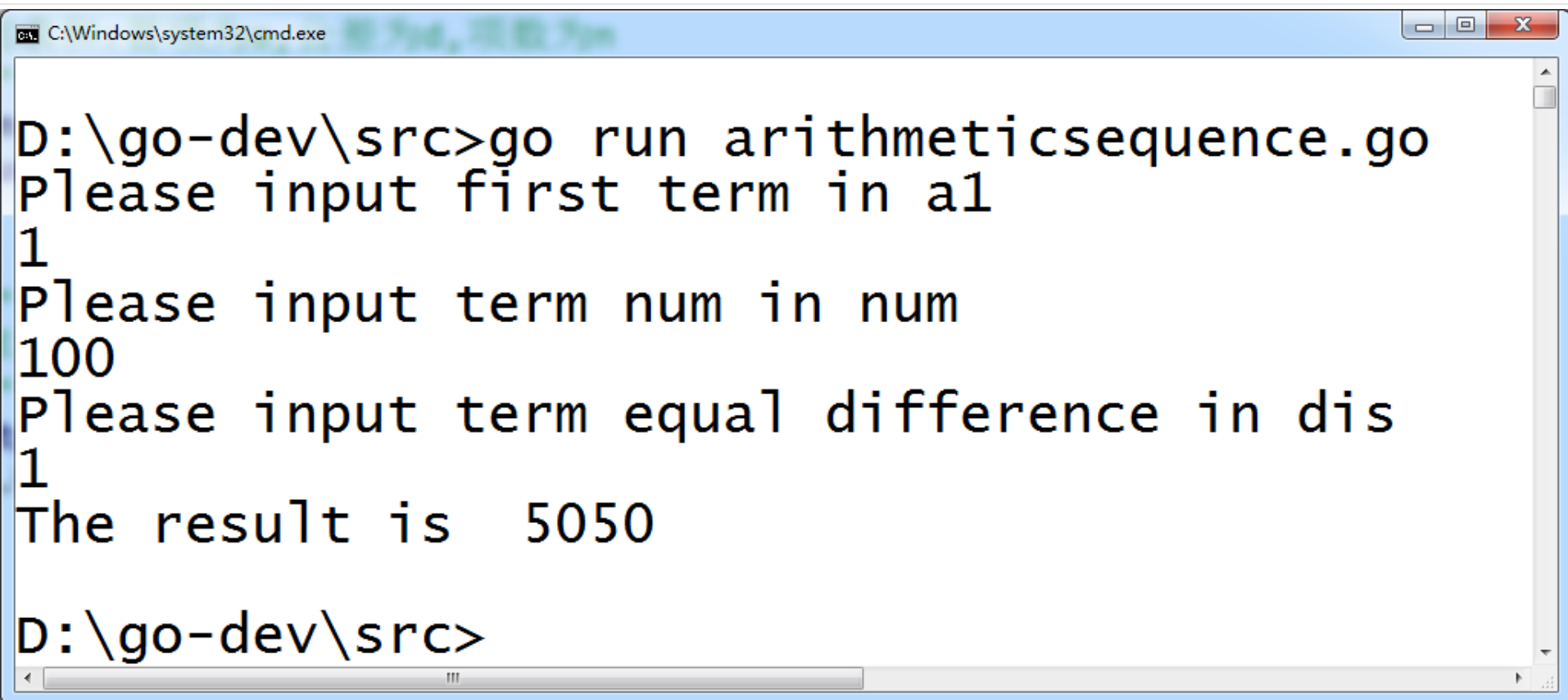
## 1.4 实现(续)

### □ 输出函数

```
32  /*******  
33  输出计算结果  
34  *****/  
35  func output(r int) {  
36      fmt.Println("The result is ", r)  
37  }
```

# 1.5 编译

---



A screenshot of a Windows command prompt window. The title bar shows the path 'C:\Windows\system32\cmd.exe'. The command prompt displays the following text:

```
D:\go-dev\src>go run arithmeticsequence.go
Please input first term in a1
1
Please input term num in num
100
Please input term equal difference in dis
1
The result is 5050

D:\go-dev\src>
```

# 1.5 总结

---

```
□ func sum(a1, num, dis int) int {  
    return (num*a1 + num*(num-1)*dis/2)  
}
```

## ■ 函数定义的模式

```
□ func 函数名(参数1 类型, 参数2 类型,...,参数n 类型) (返回值1,返回值2,...,返回值n){  
    函数体  
}
```

■ func 为函数定义关键字，sum 为函数名，  
a1,num,dis int为函数参数，最后一个int表明返回一个整数值。

■ 函数可以接受0个或多个参数。a1,num,dis int 实为  
a1 int,num int,dis int的简写

■ return (num\*a1 + num\*(num-1)\*dis/2) 函数  
执行结束语句，将表达式的值返回给主调函数

## 1.5 总结(续)

### □ var a, n, d, s int

- var 语句定义了一个变量的列表；跟函数的参数列表一样，类型在后面。
- var 语句可以定义在包或函数级别
- 注意参数类型写在变量名之后。

```
var.go x
1 package main
2
3 import "fmt"
4
5 var c, python, java bool
6
7 func main() {
8     var i int
9     fmt.Println(i, c, python, java)
10 }
```

作用域为整个包

作用域为main函数



# 1.5 总结(续)

---

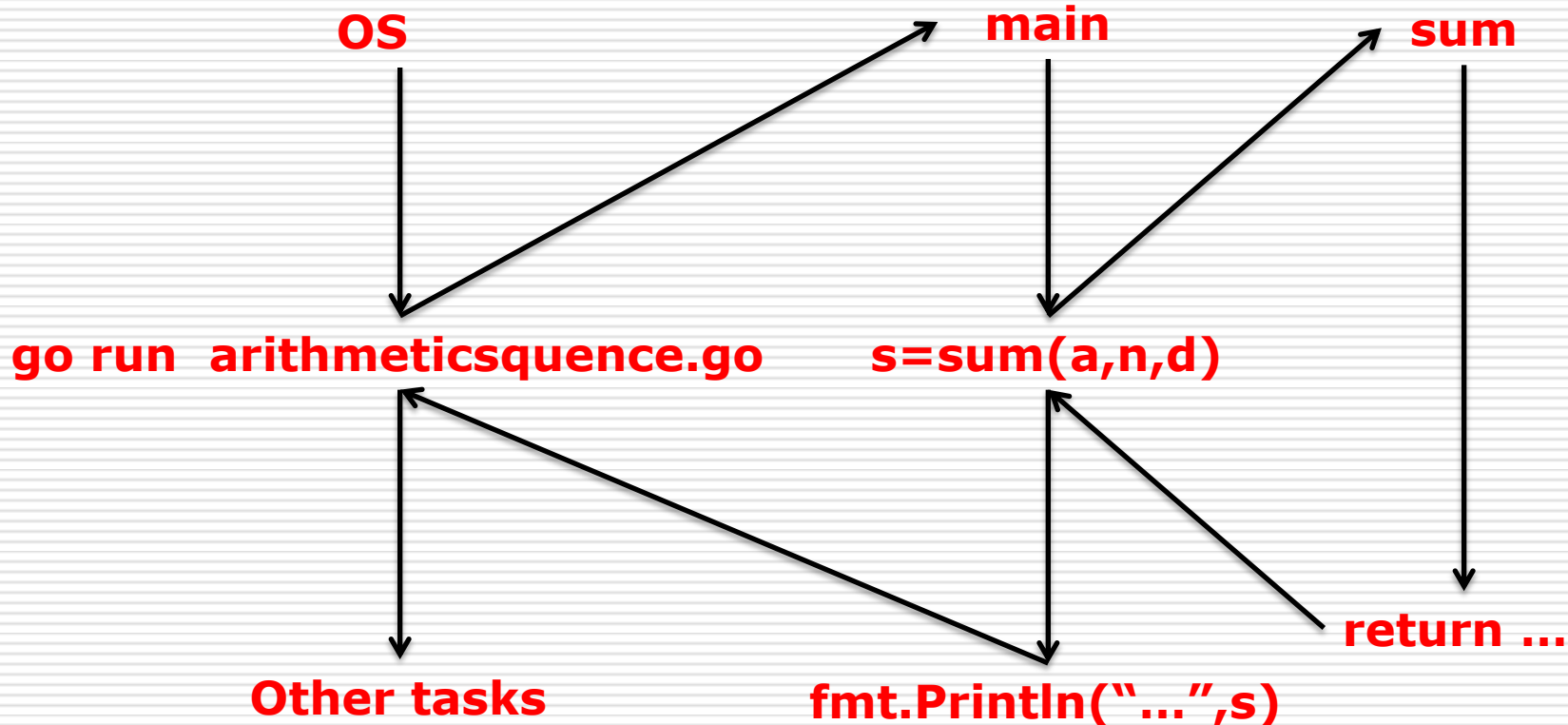
## □ fmt.Scanln(&a1)

- fmt 包实现了格式化I/O函数，类似于C的 printf 和 scanf。格式“占位符”衍生自C，但比C更简单。
- 该语句的功能是录入一个值到变量a1的地址，即给变量a1赋值。并以回车结束录入
- &放在变量名前，是指取该变量的地址。这种情况称其为取地址符号。
- 关于fmt包请参看  
<https://github.com/polaris1119/The-Golang-Standard-Library-by-Example/blob/master/chapter01/01.3.md>

# 1.5 总结(续)

□ `s = sum(a, n, d)`

- 调用sum函数，a,n,d作为实际参数，分别传给sum函数的形参a1,num,d1s。
- 调用结束后，返回到主调函数，并将结果赋给变量s

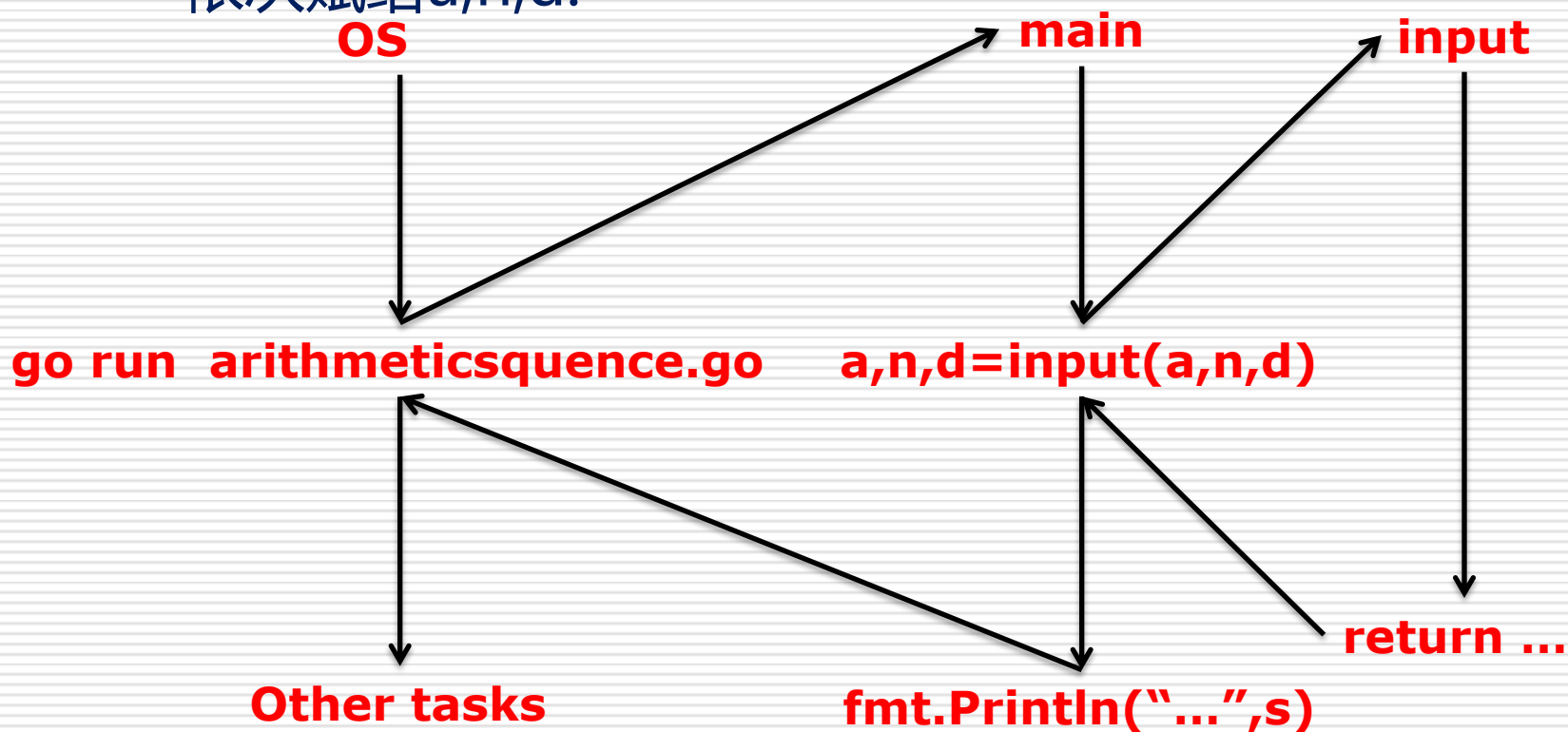


调用关系图

# 1.5 总结(续)

□ `a, n, d = input(a, n, d)`

- 调用`input`函数，`a,n,d`作为实际参数，分别传给`input`函数的形参`a1,num,dis`。
- 调用结束后，返回到主调函数，并将返回的三个值依次赋给`a,n,d`。



调用关系图

## 1.5 总结(续)

□ `fmt.Println("The result is ", s)`

- 双引号内为格式控制符，普通字符原样输出
- 逗号后面为输出项列表，用逗号间隔，这里只有s
- 因此，本条语句的执行结果为：**The result is s的值**并在末尾追加一个换行符
- 更多内请参看fmt包说明

□ 在Go语言中函数**不用先声明即可使用**

```
1 package main
2
3 import "fmt"
4
5 func main() {
6     var a, n, d, s int
7     a, n, d = input(a, n, d) //调用录入函数
8     s = sum(a, n, d)         //调用求和函数
9     output(s)                //调用输出函数
10 }
```

不用事先声明，  
直接调用

# 1.5 总结(续)

## □ 函数可以返回任意数量的返回值

- input函数有三个返回值
- sum函数有一个返回值
- output没有返回值

```
13 录入首项a1,项数num,公差dis
14 ******/
15 func input(a1, num, dis int) (int, int, int) {
16     fmt.Println("Please input first term in a1")
17     fmt.Scanln(&a1) //录入首项a
18     fmt.Println("Please input term num in num")
19     fmt.Scanln(&num) //录入项数n
20     fmt.Println("Please input term equal difference in dis")
21     fmt.Scanln(&dis) //录入公差d
22     return a1, num, dis
23 }
24 ******/
25 等差数列求和:首项为a,公差为d,项数为n
26 ******/
27 func sum(a1, num, dis int) int {
28     return (num*a1 + num*(num-1)*dis/2)
29 }
30 ******/
31 输出计算结果
32 ******/
33 func output(r int) {
34     fmt.Println("The result is ", r)
35 }
```

## 1.6 思考

---

- 请画出从os→main→input→sum→output的完整的函数调用关系图

## 1.6 思考(续)

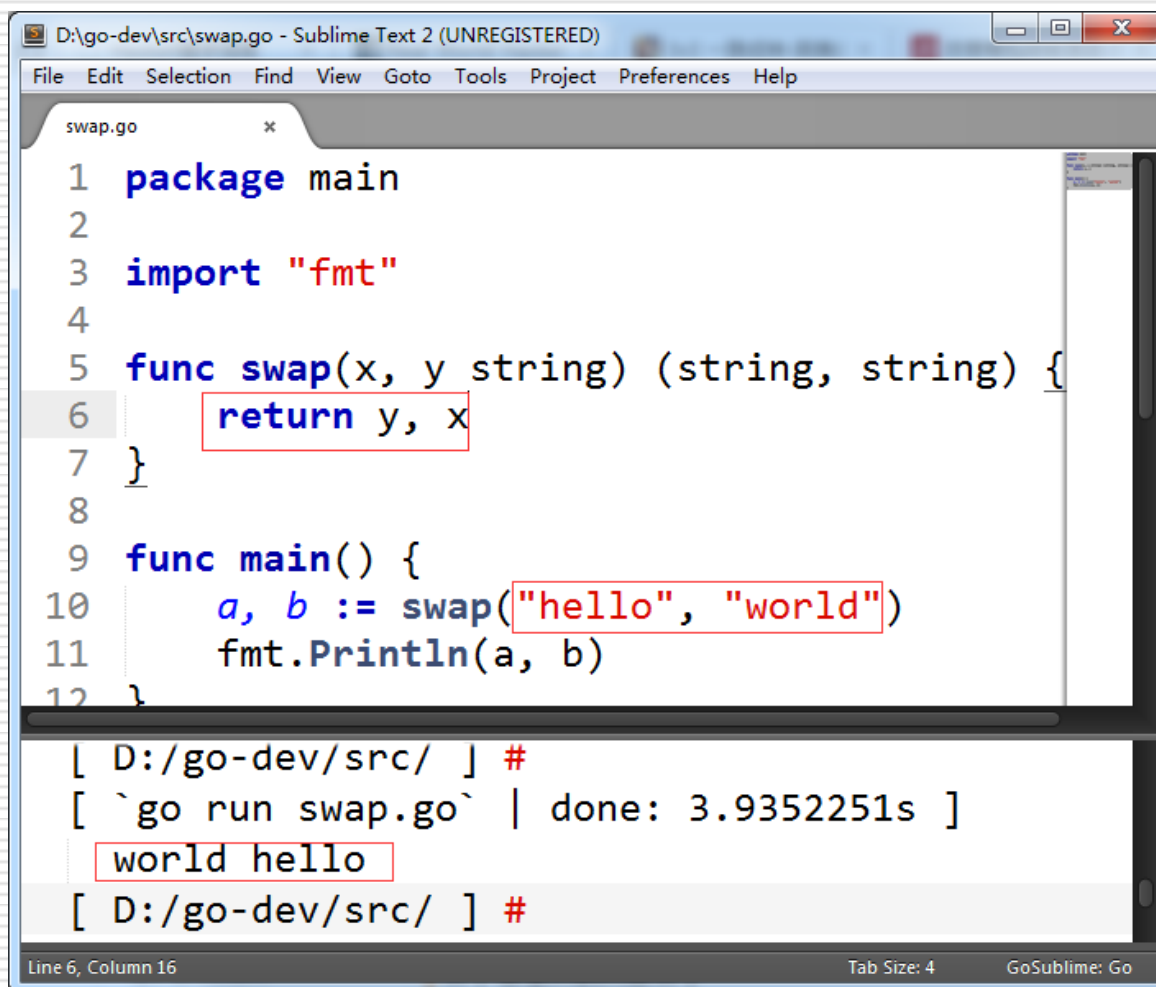
- 试着在Sublime的console下编译并执行等差数列求和程序，观察会出现什么情况？并解释说明为什么？如何解决这个问题呢？

```
▼ [ `go run arithmeticsequence.go` | done: 4.6302648s ]  
Please input first term in a1  
Please input term num in num  
Please input term equal difference in dis  
The result is 0  
[ D:/go-dev/src/ ] # |
```

- 出现的情况：运行时候，无法对Scanln等执行录入动作，也就是说对于应该输入的时候无法输入而直接继续进行后面的语句。
- 原因：ST的Console不支持terminal，这个是ST本身的问题
- 解决办法：不用ST自带的console，改用windows下的console来执行程序。

# 1.6 思考(续)

□ 请给出下面代码的结果



```
D:\go-dev\src\swap.go - Sublime Text 2 (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

swap.go
1 package main
2
3 import "fmt"
4
5 func swap(x, y string) (string, string) {
6     return y, x
7 }
8
9 func main() {
10     a, b := swap("hello", "world")
11     fmt.Println(a, b)
12 }

[ D:/go-dev/src/ ] #
[ `go run swap.go` | done: 3.9352251s ]
world hello
[ D:/go-dev/src/ ] #

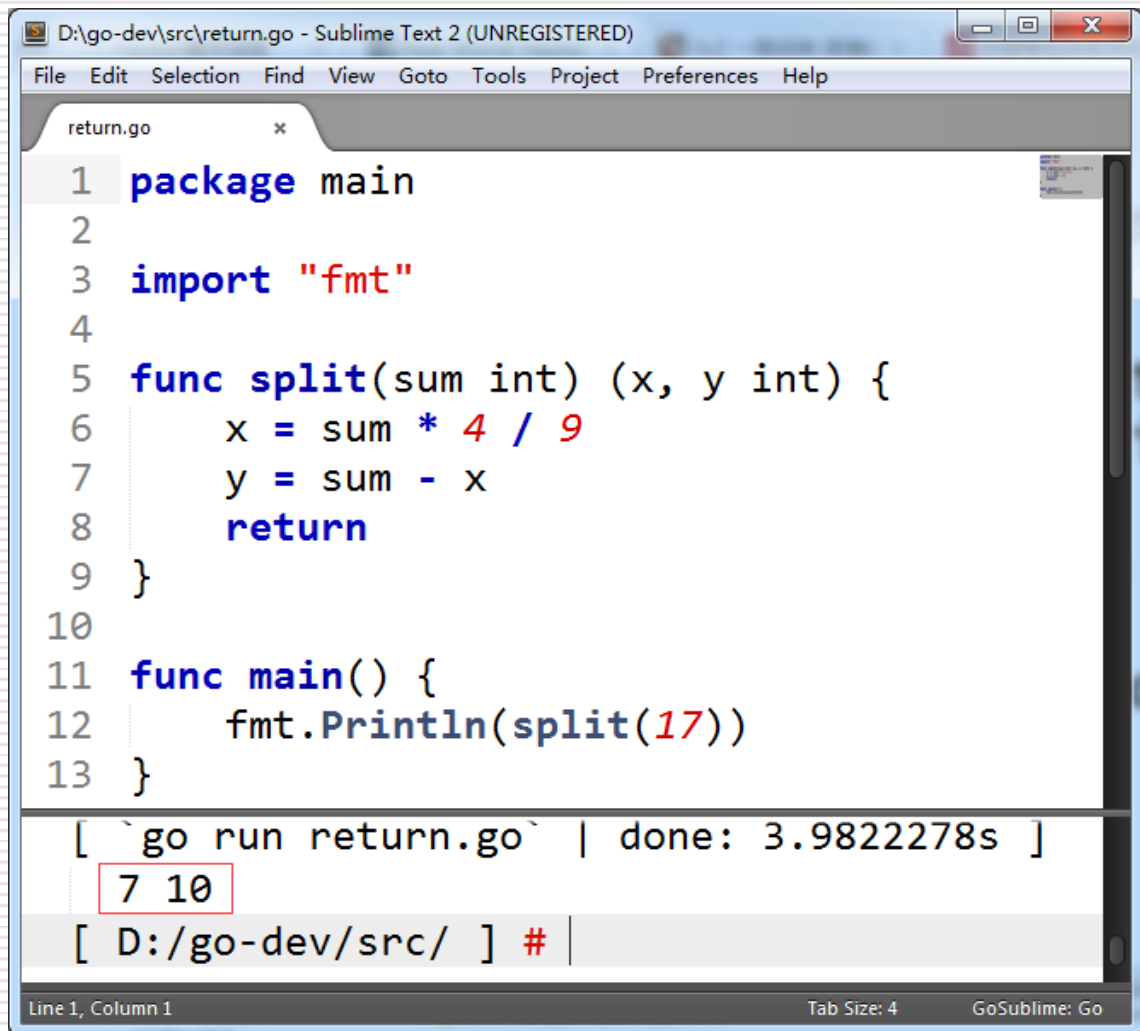
Line 6, Column 16 Tab Size: 4 GoSublime: Go
```

- swap函数实现了两个字符串x,y的交换
- (string,string)表明函数有两个字符串类型的返回值
- 通过return y , x实现交换功能。
- :=为短变量定义及赋值符号
  - 定义了两个变量a,b并赋值为swap函数的返回值。
  - 只能在函数内定义，作为临时或局部变量使用



## 1.6 思考(续)

□ 请给出下面代码的执行结果，并说明为什么会得出这样的结果？



```
D:\go-dev\src\return.go - Sublime Text 2 (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

return.go x
1 package main
2
3 import "fmt"
4
5 func split(sum int) (x, y int) {
6     x = sum * 4 / 9
7     y = sum - x
8     return
9 }
10
11 func main() {
12     fmt.Println(split(17))
13 }

[ `go run return.go` | done: 3.9822278s ]
7 10
[ D:/go-dev/src/ ] #

Line 1, Column 1 Tab Size: 4 GoSublime: Go
```

- 没有参数的 `return` 语句返回结果的当前值。也就是`直接`返回
- 直接返回语句可读性差
- 因此要给返回值指定名称为好，如 `return x,y`。即养成命名返回值的好习惯。

---

**Thank you very much**

*Any comments and suggestions  
are beyond welcome*