优化

1. 方法
2. 数据结构、流程
3. 指令

取整

- 向下取整
  - 不大于x的最大整数
- 向上取整
  - 不小于x的最小整数
- 向0取整
  - 大于0，向下取整
  - 小于0，向上取整

在C中，余数的符号跟被除数一致

```
10 % 3 = 1
10 % -3 = 1
-10 % 3 = -1
-10 % -3 = -1
```

右移属于向下取整，在负数里与C的向上取整有差异，故负数除法用右移代替会有误

```
10101 >> 3
在数学上：10101 >> 3 = 10.101
在c语言上：10101 >> 3 = 10
精度丢失，比真值小，属于向下取整

-10101 >> 3
在数学上：-10101 >> 3 = -10.101
在c语言上：10101 >> 3 = -10
精度丢失，比真值大，属于向上取整
```

循环与goto

- `while`

  ```
  cmp表达式计算真值，为真就执行循环体中的内容，否则退出
  while(cmp) {
        ...
  }
  ```

- `do-while`

```
先执行一次循环体的内容，然后判断cmp表达式的真值，为真就继续执行循环体的内容
do {
      ...
} while(cmp)
```

- `for`

```
第一步：执行init初始化部分
第二步：判断cmp表达式真值，为真就进行第三步，否则就到第六步
第三步：执行循环体
第四步：执行step，循环计数增加步长step
第五步：转而去第二步
第六步:结束循环
for (init; cmp; step) {
      ...
}
```

- `goto`
  - 多重循环跳出到最外层
  - 低级语言模拟异常处理
  - 标号属于常量，代表标号后一条语句的地址，标号一般大写

continue和break

- `continue`
  忽略之后循环体的内容，接着执行下此循环

```
while/do-while/for
// 循环体
{
      if(cmp) {
          continue; // 忽略后面 TODO 的内容，执行下一次的循环
      }
      // TODO:
      ...
}
```

- `break`
  跳出当前循环

```
while/do-while/for
// 循环体
{
       if(cmp) {
           break;;   // 直接跳出循环，即转去执行 TODO 后的内容
       }
       ...
}
// TODO:
```

作业

1. goto模拟三种循环
2. 猴子吃桃，等比数列
3. 判断质数