

## 12章 数值类

作者 bluetea

网站 <https://github.com/bluetea>

数值类 Numeric

Numeric类的分类：

1.Integer 整数类，分为2个子类

(1)Fixnum 普通整数

(2)Bignum 大整数

2.Float 浮点小数类

3.Rational 有理数类

4.Complex 复数类

有理数的支持：

也可以用to\_f 函数将有理数转换为float对象

```
[7] pry(main)> a = Rational(4,10)
=> (2/5)
[8] pry(main)> b = Rational(1, 3)
=> (1/3)
[9] pry(main)> c= a + b
=> (11/15)
[10] pry(main)> c.to_f
=> 0.7333333333333333
```

数值对象的字面量,字面量的意思就是各种显示的方式

```
1 123      #10进制
2 0123     #8进制, 转换为10进制为83
3 0o123    #8进制, 转换为10进制为83
4 0d123    #10进制
5 0x123    #16进制, 转换为10进制 291
6 0b1111011 #2进制, 转换为10进制为 123
7 123.45   #浮点小数
8 1.23e4   #浮点小数的指数表示法, 1.23*10的4次幂 = 12300.0
9 1.23e-4  #浮点小数的指数表示法, 1.23*10的-4次幂 = 0.000123
```

一些运算的方法

1.x.div(y)

```
[14] pry(main)> 5.div(2)
=> 2
[15] pry(main)> 5.div(1)
=> 5
```

2.x.quo(y) 如果x和y都是整数，那么返回有理数 Rational，否则返回浮点数

```
[18] pry(main)> 5.quo(2)
=> (5/2)
[19] pry(main)> 5.quo(2.2)
=> 2.2727272727272725
```

3.x.modulo(y) 取模运算

```
[20] pry(main)> 5.modulo(2)
=> 1
[21] pry(main)> 5.modulo(3)
=> 2
```

4.x.divmod(y) 将x除以y后的商和余数最为数组返回

```
[23] pry(main)> 10.divmod(3)
=> [3, 1]
[24] pry(main)> 10.divmod(3.5)
=> [2, 3.0]
[25] pry(main)> 10.divmod(-3.5)
=> [-3, -0.5]
```

12.4 Math模块提供了三角幻术，对数函数等常用的函数运算 书 155页

12.5 整数类型的转换

to\_f 转换为浮点数

to\_i 转换为整数

#round 对小数进行四舍五入

```
[34] pry(main)> 22.round
=> 22
[35] pry(main)> 22.33.round
=> 22
[36] pry(main)> 22.33555.round
=> 22
```

#ceil 返回比接收大的最大整数

#floor 返回比接收小的最大整数

```
[40] pry(main)> 5.2.ceil  
=> 6  
[41] pry(main)> 5.2.floor  
=> 5
```

## 12.6位运算

1. ~ 按位取反
2. & 按位与
3. | 按位或
4. ^ 按位异或 ((a&~b | -a&b))
5. >> 位右移
6. << 位左移

## 12.7随机数

Random.rand 不指定参数返回比1小的浮点小数，指定参数后，返回0到这个正整数之间的整数

```
[80] pry(main)> Random.rand  
=> 0.6381276871722726  
[81] pry(main)> Random.rand(5)  
=> 2  
[82] pry(main)> Random.rand(5.3)  
=> 3.8233837780643394
```

模拟随机数是一个种子来生成随机的，只要随机数的种子一样的，得到的值很有可能是重复的

```
random.rb  
1 r1 = Random.new(1)  
2 p [r1.rand, r1.rand]  
3 r2 = Random.new(1)  
4 p [r2.rand, r2.rand]  
  
1. bash  
wangmjcdMacBook-Pro:ruby wangmjc$ ruby random.rb  
[0.417022004702574, 0.7203244934421581]  
[0.417022004702574, 0.7203244934421581]
```

所以要得到优质随机数，就能指定种子值，直接就用Random.rand 来实现

```
random.rb  
1  
2 p [Random.rand, Random.rand]  
3  
4 p [Random.rand, Random.rand]  
  
1. bash  
wangmjcdMacBook-Pro:ruby wangmjc$ ruby random.rb  
[0.8517287838793666, 0.1585007140375574]  
[0.90064459501508, 0.9524770672790569]  
wangmjcdMacBook-Pro:ruby wangmjc$
```

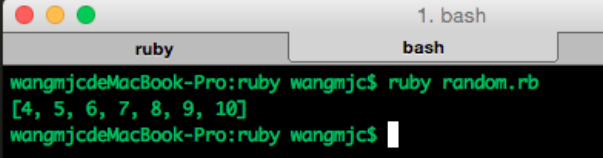
## 12.8计数

### 1. n.times{|i| ...}

```
random.rb  
1 ary = []  
2 10.times do |i|  
3   ary << i  
4 end  
5 p ary  
  
1. bash  
wangmjcdMacBook-Pro:ruby wangmjc$ ruby random.rb  
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]  
wangmjcdMacBook-Pro:ruby wangmjc$
```

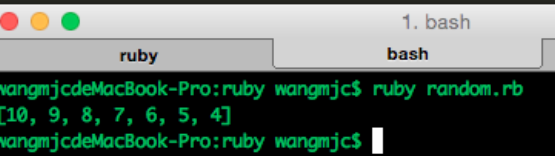
### 2.from.upto(to) {|i| ...}

```
1 ary = []
2 4.upto(10) do |i|
3   ary << i
4 end
5 p ary
```



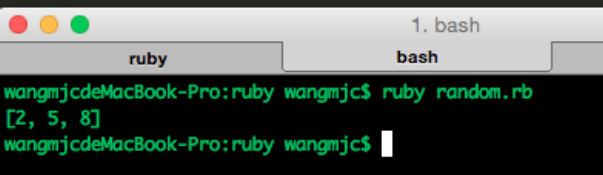
3.from.downto(to) {|i| ...}

```
1 ary = []
2 10.downto(4) do |i|
3   ary << i
4 end
5 p ary
```



4.from.step(to, step) {...}

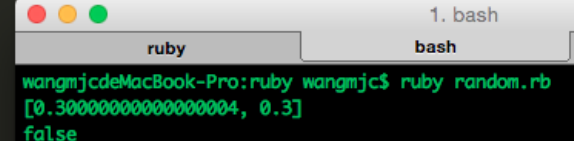
```
1 ary = []
2 2.step(10, 3) do |i|
3   ary << i
4 end
5 p ary
```



## 12.9近似值的误差

在处理浮点小数时，很容易因为误差产生问题

```
1 a = 0.1 + 0.2
2 b = 0.3
3 p [a, b]
4 p a == b
```



如果把小数转换为2个整数相除的形式，那么通过使用Rational的类进行计算，就会避免误差

```
=> 0.30000000000000004
[97] pry(main)> a = Rational(1, 10) + Rational(2, 10)
=> (3/10)
[98] pry(main)> b = 0.3
=> 0.3
[99] pry(main)> a == b
=> true
```

## Comparable 模块

Ruby的比较运算符(== <= 等等)实际上都是方法，comparable模块里面封装了比较运算符，将其Mix-in 到类后，就可以实现实例间互相比较的方法，comparable 是可比较的意思

comparable模块中各个运算符都会使用<=>运算符的结果

