第五章 整除性与最大公因数

> **欧几里得算法**：要计算两个整数a与b的最大公因数，先令$r_{-1}=a$且$r_0=b$，然后计算相继的商和余数
>
> $$r_{i-1}=q_{i+1}\times r_i+r_{i+1}(i=0,1,2,\dots)$$
>
> 直到某余数r_{n+1}为0。最后的非零余数r_n就是a与b的最大公因数。

> **习题**

- *5.1* 应用欧几里得算法计算下述最大公因数。
  - *(a)* $gcd(12345, 67890)$
  - *(b)* gcd(54321, 9876)

(a) $67890=5\times12345+6165$	(b) $54321=5\times9876+4941$
$12345=2\times6165+15$	$9876=1\times4941+4935$
$6165=411\times15+0$	$4941=1\times4935+6$
$gcd(12345,67890)=15$	$4935=822\times6+3$
	$6=2\times3+0$
	$gcd(54321,9816)=3$

- *5.2* 编写程序计算两个整数a与b的最大公因数$gcd(a, b)$。即使a与b中的一个等于零，程序也应该运行，但要确保a与b都是0时不会出现死循环。

```go
package main

import (
	"fmt"
)

func gcd(a, b int) int {
	if a == 0 {
		return abs(b)
	}
	if b == 0 {
		return abs(a)
	}

	for b != 0 {
		a, b = b, a%b
	}
	return abs(a)
}

func abs(x int) int {
```

```go
    if x < 0 {
        return -x
    }
    return x
}

func main() {
    var a, b int
    fmt.Println("请输入两个整数:")
    _, err := fmt.Scan(&a, &b)
    if err != nil {
        return
    }

    result := gcd(a, b)
    fmt.Printf("最大公因数是: %d\n", result)
}
```

- *5.3* 设$b = r_0, r_1, r_2, ...$是将欧几里得算法应用于*a*与*b*得到的相继余数，证明每两步会缩小余数至少一半。换句话说，验证

$$r_{i+2} < \frac{1}{2} r_i (i = 0, 1, 2, ...)$$

5.3. $r_i = q_{i+2} \times r_{i+1} + r_{i+2} \Rightarrow r_{i+2} = r_i - q_{i+2} \times r_{i+1}$

① 当$r_{i+1} < \frac{1}{2} r_i$时 $r_{i+2} < \frac{1}{2} r_i$ 恒成立

② 当$r_{i+1} \geq \frac{1}{2} r_i$时，$r_{i+2} \leq r_i - \frac{1}{2} q_{i+2} \times r_i = (1 - \frac{1}{2} q_{i+2}) \times r_i$

因为$q_{i+2} > 1$，所以$r_{i+2} < \frac{1}{2} r_i$ $(i = 0, 1, 2, ...)$ 得证

- *5.4*
  - *(a)* 求下述最小公倍数：*(i)* $LCM(8, 12)$，*(ii)* $LCM(20, 30)$，*(iii)* $LCM(51, 68)$，*(iv)* $LCM(23, 18)$。
  - *(b)* 对于在*(a)*中计算的每个LCM，比较$LCM(m, n)$与m，n，$gcd(m, n)$的值，试找出它们之间的关系。
  - *(c)* 证明你所发现的关系对所有m与n成立。
  - *(d)* 用*(b)*中的结果计算$LCM(301337, 307829)$。
  - *(e)* 假设$gcd(m, n) = 18$且$LCM(m, n) = 720$，求m与n。存在一种以上的可能性吗？如果存在，求出所有的m与n。

(a) (i) $LCM(8,12)=24$ (ii) $LCM(20,30)=60$ (iii) $LCM(51,68)=204$

(iv) $LCM(23,18)=414$

(b)

m	n	LCM(m,n)	gcd(m,n)
8	12	24	4
20	30	60	10
51	68	204	17
23	18	414	1

$LCM(m,n)gcd(m,n)=mn$

(c) 设$m=x\,gcd(m,n)$，$n=y\,gcd(m,n)$

$LCM(m,n)=xy\,gcd(m,n)$，所以 $LCM(m,n)gcd(m,n)=xy\,gcd(m,n)gcd(m,n)$

$=mn$ 得证

(d) $307829=1\times301337+6492$

$301337=46\times6492+2705$

$6492=2\times2705+1082$

$2705=2\times1082+541$

$1082=2\times541+0$

所以$gcd(301337,307829)=541$

根据(b)得 $LCM(301337,307829)=301337\times307829/541=171460753$

(e) 根据(c) $LCM(m,n)/gcd(m,n)=xy=40$

又因为$gcd(x,y)=1$，所以$(m,n)=(18,720),(90,144),(144,90),(720,18)$

- 5.5
    - (a) 对n的下述每一个开始值，求$3n+1$算法的长度与终止值：(i) $n=21$，(ii) $n=13$，(iii) $n=31$。
    - (b) 做进一步的实验，试确定$3n+1$算法是否总是终止，如果终止的话，它在何值处终止？
    - (c) 假定算法在1处终止，设$L(n)$是开始值为n的算法长度。证明$n=8k+4(k\geq 1)$时$L(n)=L(n+1)$。

- (d) 证明$n = 128k + 28$时$L(n) = L(n + 1) = L(n + 2)$。

(a) (i) 21, 64, 32, 16, 8, 4, 2, 1, 4, 2, 1, ... 长度为8，终止值为1

(ii) 13, 40, 20, 10, 5, 16, 8, 4, 2, 1, ... 长度为10，终止值为1

(iii) 31, 94, 47, 142, 71, 214, 107, 322, 161, 484, 242, 121, 364, 182, 91 ... 4, 2, 1

长度为107，终止值为1

(b) 总是终止，在1处终止。

(c) $n=8k+4$时，$4k+2$，$2k+1$，$6k+4$，$3k+2$，...，4，2，1

$n=8k+5$时，$24k+16$，$12k+8$，$6k+4$，$3k+2$，...，4，2，1

所以长度相同，$L(n)=L(n+1)$

(d) $n=128k+28$时，$64k+14$，$32k+7$，$96k+22$，$48k+11$，$144k+34$，$72k+17$，$216k+52$，$108k+26$，$54k+13$，$162k+40$，$81k+20$，...，4，2，1

$n=128k+29$时，$384k+88$，$192k+44$，$96k+22$，$48k+11$，$144k+34$，$72k+17$，$216k+52$，$108k+26$，$54k+13$，$162k+40$，$81k+20$，...，4，2，1

$n=128k+30$时，$64k+15$，$192k+46$，$96k+23$，$288k+70$，$144k+35$，$432k+106$，$216k+53$，$648k+160$，$324k+80$，$162k+40$，$81k+20$，...，4，2，1

- 5.6 编写程序来运行上题叙述的$3n + 1$算法。用户输入n，程序就会给出$3n + 1$算法的长度$L(n)$与终止值$T(n)$。用你的程序制作一个表格，给出所有开始值$1 \le n \le 100$的算法长度与终止值。

```go
package main

import (
	"fmt"
)

func calculation(n int) (int, int) {
	length := 1
	for n != 1 {
		if n%2 == 0 {
			n /= 2
		} else {
			n = 3*n + 1
		}
		length++
	}

	return length, 1
}

func main() {
	fmt.Printf("|  n  | Length L(n) | Termination T(n) |\n")
	fmt.Printf("|-----|-------------|------------------|\n")

	for n := 1; n <= 100; n++ {
		length, termination := calculation(n)
		fmt.Printf("| %3d | %11d | %16d |\n", n, length, termination)
```

```
    }
}
```