1　ユークリッドの互除法

「ラメの定理」は、最大公約数を求めるのに、なぜ小学校で最初に学ぶような素因数分解よりも、互除法が優れているかを、はっきりと教えてくれる。扱う対象がせいぜい数百か数千までの数ならば、素因数分解したほうが速いこともある。しかし、数十桁の数になると、素因数分解は、とてつもなく時間がかかる。世界最高速の電子計算機でも長時間かかる。これに対して、互除法なら、たとえば 100 桁の数でも最大 500 回の除法ですむ。これは現代の高速計算機なら、1 秒の何百分の 1 以下の時間でできる P 問題である。この劇的な所要時間の差が、RSA 暗号体系の基礎の一つでもある。

暗号の数理　一松信著　ブルーバックス　169 頁

互除法

二つの正の整数 m、n の最大公約数を求めるユークリッドの互除法は、一部の高校の教科書にも載っており、かなりよく知られている。それは次のような指令の集まりからなる。

1　m と n を比較し、必要なら入れかえて $m \geq n$ とせよ。

2　m を n で割り、商を q、余りを r とせよ。

3　余り r が 0(割り切れた) ならば、そのとき除数 n が最大公約数である。これで完了。

4　余り r が 0 でなければ、除数 n と r を m、n と置き直して、2 へ戻れ。

暗号の数理　一松信著　ブルーバックス　166 頁

練習問題

m と n の最大公約数を求めよ。

1　$m = 187$　$n = 143$

2　$m = 561$　$n = 231$

3　$m = 7564$　$n = 588$

4　$m = 119790$　$n = 42900$

5　$m = 919199$　$n = 830407$

初等整数論　H. スターク著　現代数学社　31 頁

1.1 暗号の解読 euclid.c

```
/*
  ユークリッドの互除法
  euclid.c
*/
#include <stdio.h>

int main()
{
  int a;
  int b;
  int m;
  int n;
  int r;

  printf("二つの整数を入力してください。");
  scanf("%d %d", &a, &b);

  m = a;
  n = b;

  while(m != n){
    if(m > n){
      m = m - n;
    }else{
      n = n - m;
    }
  }

  printf("最大公約数 = %d\n", m);

  /*  別解
  do{
     r = m % n;
     m = n;
     n = r;
  }while(r != 0);
```

```
    printf("最大公約数 = %d\n", m);
    */

    return 0;
}
```

C 言語によるはじめてのアルゴリズム　河西朝雄著　技術評論社　36 頁