主題: Prime numbers

- 基礎
- 應用
- 作業與自我挑戰

基礎

- Prime numbers
- Determination of prime numbers
- Constructing a table of prime numbers

Prime numbers

- 一個大於等於 2 的數，除了 1 及本身，沒有其它因數，就是質數

- 假設無需處理「大數」

Determination of prime numbers: Method 1

- 2 到 n - 1 間的整數都不是 n 的因數，則 n 是質數
  - 最慢，省記憶體

```
is_prime1(int n) {
        for (i = 2 ; i < n ; i++)
                if ((n % i) == 0) return false;

        return true;
}
```

Time complexity???

Method 2

- 2 到 sqrt(n) 間的整數都不是 n 的因數，則 n 是質數
  - 次慢，省記憶體

why ???

```
is_prime2(int n) {
        int m = (int) (sqrt(n)+0.001);
        for (i = 2 ; i <= m ; i++)
                if ((n % i) == 0) return false;

        return true;
}
```

Time complexity???

Method 3

- 假設已建好一範圍在 2 到 n - 1 $(n \geq 3)$ 之間的質數表 prime，prime[i] 存第 i + 1 個質數，$0 \leq i < p\_num$

Example: n = 22, p_num = 8

	0	1	2	3	4	5	6	7
prime	2	3	5	7	11	13	17	19

Method 3 (cont.)

- 2 到 sqrt(n) 間的所有質數都不是 n 的因數，則 n 是質數 $(n \geq 3)$
- 最快，浪費記憶體

why ???

```
is_prime3(int n) {
        int m = (int) (sqrt(n)+0.001);
        for (i = 0 ; prime[i] <= m && i < p_num; i++)
                if ((n % prime[i]) == 0) return false;

        return true;
}
```

Time complexity???

Constructing a table of prime numbers

- 建造一個表儲存 2 ~ range 範圍內的所有質數
  - 基本法
  - 6N ± 1 法
  - The sieve method (篩去法)

基本法

- 利用判斷質數的方法造表
  - //prime[i] is used to store the (i+1)-th prime number

```
p_num = 1;
prime[0] = 2;
for (i = 3 ; i <= range ; i++) {
        if (is_prime3(i) == true) {
                prime[p_num] = i;
                p_num++;
        }
}
```


6N ± 1 法

- 只測試 6N ± 1 ，其它都是 2 or 3 的倍數

(i%2 ≠ 0) & (i%3 ≠ 0)

```
p_num = 2;
prime[0] = 2, prime[1] = 3;
for (i = 5 ; i <= range ; i++) {
        if (i%6 == 1 || i%6 == 5) {
                if (is_prime3(i) == true) {
                        prime[p_num] = i;
                        p_num++;
                }
        }
}
```


The sieve method (篩去法)

- 建立一個大小是 range 的陣列 is_prime (bit map)
- 方法: 將每一個質數的所有倍數拿掉，剩下的就是質數

Example: range = 29

is_prime[k] = { 1: k is a prime; 0: k is not }

不使用

0 1	1 1	2 1	3 1	4 ~~1~~ 0	5 1	6 ~~1~~ 0	7 1	8 ~~1~~ 0	9 ~~1~~ 0
10 ~~1~~ 0	11 1	12 ~~1~~ 0	13 1	14 ~~1~~ 0	15 ~~1~~ 0	16 ~~1~~ 0	17 1	18 ~~1~~ 0	19 1
20 ~~1~~ 0	21 ~~1~~ 0	22 ~~1~~ 0	23 1	24 ~~1~~ 0	25 ~~1~~ 0	26 ~~1~~ 0	27 ~~1~~ 0	28 ~~1~~ 0	29 1

int m = (int) (sqrt(n)+0.001);

i <= m

overflow !!!

k = i*i

```
char is_prime[range+1];
memset(is_prime, 1, range+1);
    //初始設定 is_prime 中所有 element 為 1

for (i = 2; i <= range; i++) {
        if (is_prime[i] == 1) {
                for (k = i*2 ; k <= range ; k = k + i) {
                        is_prime[k] = 0; //篩去所有倍數
                }
        }
}
```

- 優點：快速
  - $O(n/2 + n/3 + n/4 + ...) = O(n \lg n)$
  - 沒有使用除法
- 缺點：larger storage (range 不可以太大)
- **Remark:** As binary search, the sieve method should be considered as a technique for solving problems !

How to handle $n > 10^7$ (≈ 32M)?

Assume only 32M RAM is available

(1) Save storage of the sieve method

- bit-wise: save 7/8 storage
  - hard to implement
  - can not handle $n > 10^9$
  - How to get/set/reset a bit of a byte ???
    - get(x, i)
    - set(x, i)
    - reset(x, i)

How to handle $n > 10^7$? (cont.)

(2) Use a table to store all prime numbers $\leq \sqrt{n}$

- check whether a number $k \leq \sqrt{n}$ is prime in $O(\lg n)$ time (How ???)
- check whether a number $k > \sqrt{n}$ is prime in $O(\sqrt{n}/\lg n)$ time
- storage: $O(\sqrt{n}/\lg n)$

How to handle $n > 10^7$? (cont.)

(3) Use the sieve method for all numbers $\leq \sqrt{n}$

- check whether a number $k \leq \sqrt{n}$ is prime in $O(1)$ time
- check whether a number $k > \sqrt{n}$ is prime in $O(\sqrt{n})$ time
- storage: $O(\sqrt{n})$

應用

- 應用一: A.583 Prime Factors
- 應用二: A.294 Divisors
- 應用三: A.10622 Perfect Pth Powers
- 應用四: A.369 Combinations
- 應用五: A.10168 Summation of Four Primes
- 應用六: 韓信點兵

應用一: A. 583 Prime Factors

- 給一個整數 g ，作質因數分解 ($-2^{31} < g < 2^{31}$)
- Solution:
  - 找 ≤ sqrt(g) 的質因數:
    先建質數表，從質數表的第一個質數往後看到超過 sqrt(g)為止，只要除的盡就不斷的除以該質數
  - 找 > sqrt(g) 的質因數: 最多只有一個
- 建一個 plist array 存所有質因數
- 建一個 ppow array 存每個質因數的次數

Example

	0	1	2	3	4	5	6	7	8
prime	2	3	5	7	11	13	17	19	23

- $660 = 2^2 \times 3^1 \times 5^1 \times 11^1$

plist	2	3	5	11
ppow	2	1	1	1


```
num = 0; //質因數的個數
m = (int) (sqrt(g) + 0.001);
x = g;
for (i = 2 ; i <= m ; i++) {
  if ((is_prime[i] == 1) && (x % i == 0)) {  //用篩去法的質數表
     plist[num] = i;  // 新質因數, 放入質因數表
     ppow[num] = 0;
     do {  x = x / i;          // 抽掉一個
           ppow[num]++;  // 次數 +1
      } while (x % i != 0)
      num++;  // 質數增加一個
   }
}
if (x > 1) {  //x 是唯一大於 m 的質因數
  plist[num] = x;
  ppow[num] = 1;
  num++;
}
```

抽掉所有新質因數

m = (int) (sqrt(x)+0.001);

Time complexity ???

應用二: A.294 Divisors

- 給 L 和 U，求 L 到 U 之間因數個數最多的整數 P
- $1 \le L \le U \le 10^9$，$U - L \le 10^4$
- Solution: 先做因數分解，再代入求因數個數的公式

$$n = p_1^{q_1} \times p_2^{q_2} \times \ldots \times p_m^{q_m}$$

因數個數

$$(q_1+1) \times (q_2+1) \times \ldots \times (q_m+1)$$

- Time complexity ???

應用三: A.10622 Perfect Pth Powers

- 給一個 32-bit integer n，且 $|n| \ge 2$
- 請找出最大的整數 p 滿足 $n = y^p$，其中 y 是整數

64 = 8^2 = 4^3 = 2^6 → **ans: 6**

Solution

- 質因數分解
- 決定最大次方數
  - $n = 129600 = 2^6 \times 3^4 \times 5^2 = (2^3 \times 3^2 \times 5)^2$
  - 最大次方數為所有的質因數次方數的最大公因數 g
- 注意 n 是負數的狀況
  - 只有奇數次方才有可能為負數
    - eg,. $n = -(2^{24} \times 3^{12}) = -((2^2 \times 3^1))^{12} = (-(2^2 \times 3^1)^4)^3$
  - 找最大"奇"公因數 g'
  - 將 g 不斷除以二，直到成為奇數為止
    - eg., $g = 336 \rightarrow 168 \rightarrow 84 \rightarrow 42 \rightarrow 21$

Finding gcd

- a, b 兩數求公因數 $(a, b \ge 1)$
  - 輾轉相除法

```
int gcd(int a, int b) {
        if (a < b) swap(a, b); //make a > b
        if (a%b != 0)
                return gcd(b, a%b);
        else
                return b;
}
```

Other solutions ???

應用四: A.369 Combinations

- 給 n, m，求 C_m^n (5≤ m ≤ n ≤ 100)
- 答案保證是 32-bit 整數
- 注意: 直接將分母分子算出，再做除法，會 overflow
- Solution: 先將所有質數代入約分，再把分子乘出
- 例: $C_4^{10} = \frac{10\times9\times8\times7}{4\times3\times2\times1} = \frac{2^4\times3^2\times5^1\times7^1}{2^3\times3^1} = \frac{2^1\times3^1\times5^1\times7^1}{1} = 210$

應用五: A.10168 Summation of Four Primes

- 給一個偶數 $n \le 10^7$ ，找四個質數 p_1，p_2，p_3，p_4，使得 $n = p_1 + p_2 + p_3 + p_4$
- 必備知識: 所有的偶數 n >2 都是兩個質數相加的和
  (Goldbach's conjecture, which is known to be true for very large n)
- Solution:

  Let p1=p2=2 and then find p3 + p4 = n - 4

應用六: 韓信點兵

- 有一數除以 3 餘 2，除以 5 餘 3，除以 7 餘 2，這個數字最小是多少?
- 類題: A.756 Biorhythms
- 中國餘式定理: 令 $n_1, n_2, \ldots, n_k$ 為兩兩互質的整數，若 $a_1, a_2, \ldots, a_k$ 為任意整數，則
  - 存在一個整數 x 使得
    $x \equiv a_i \pmod{n_i}$ for each i = 1 to k 且
  - 若 $y \equiv a_i \pmod{n_i}$ for each i = 1 to k 則
    $y \equiv x \pmod{n_1 n_2 n_3 \ldots n_k}$

有解

解的關係

Solution

$x \equiv 2 \pmod 3$ $\Rightarrow x = 3 \times q1 + 2$

test from 0 to 4

$x \equiv 3 \pmod 5$ $\Rightarrow x = 3 \times (5 \times q2 + r2) + 2$, $r2 = 2$

$\Rightarrow x = 15 \times q2 + 8$

test from 0 to 6

$x \equiv 2 \pmod 7$ $\Rightarrow x = 15 \times (7 \times q3 + r3) + 8$, $r3 = 1$

$\Rightarrow x = 105 \times q3 + 23$


```
// m[] 是除數，z[] 是餘數，num_inp 是組數
r = z[0];
q = m[0];
for (i = 1 ; i < num_inp ; i++) {
   for (j = 0 ; j < m[i] ; j++)
       if (((j*q+r) % m[i]) == z[i])
              break;
   r = r + j*q;
   q = q*m[i];
}
```

test from 0 to m[i]-1

作業與自我挑戰

- 作業
  - 練習題
    - A.294 Divisors
      http://uva.onlinejudge.org/external/2/294.html
  - 挑戰題
    - A.10140 Prime Distance (Hint: U-L<=10^6) (很值得作 !)
      http://uva.onlinejudge.org/external/101/10140.html
- 其它有趣的題目
  - A.10311 Goldbach and Euler (n<=10^8)
    http://uva.onlinejudge.org/external/103/10311.html
  - A.756 Biorhythms
    (Simulation is also ok, since the answer is small.)
  - A.606, A.686, A.516, A.530, A.160, A.543