Fast Power :

第一個子題：只要迴圈跑一次，最後除以p就好，屬於基本迴圈題。

第二個子題：首先要知道整數的性質「同餘」。

a * b mod m = ((a mod m) * (b mod m)) mod m

知道這個性質之後，就邊乘邊取餘數就可以啦！

第三個子題：跟第二個子題一樣，只是會overflow，記得開 long long。

第四個子題：

快速冪，本題很重要的演算法！也是下學期上課的內容，利用二進制拆解去實作次方，例如：

5 次方 可以拆成 $2^2 + 2^0$ 次方，3^5 就可以寫成 $3^{2\text{^}2} \times 3^{2\text{^}0}$ 。

快速冪的操作就是：

起始底數：3 (x) , 指數：5 (y) , 第幾次：0 (num) , 答案：1 (res)

1. 問當前指數除以二是否餘 1？
2. 若是，表示 res 應該乘上當前 x ($3^{2\text{^}num}$)
3. 每次結束 指數應除二取下高斯 (y/=2)，底數要平方(x*=x)。

重複 1~3 值到 y等於0為止

時間複雜度 O($\log_2 y$)

上面的快速冪算法，配上同餘，每次乘完答案取餘，x平方完也取餘，結束。

第五個子題：

快速冪＋費馬小定理。

先利用同餘性質，把 x 用字串讀進來後，對於每個字元，把結果先乘10+該位數的數字後，取餘數。這樣就可以讀入 x 了。

同理 y 也是，但是 y 是對 (p-1) 取餘數而不是 p，為什麼呢？

根據費馬小定理當p是質數時 $a^{p-1} \equiv 1 \pmod p$，

也就是說，每 p-1 次，就會餘 1，所以次方就對 p-1 取餘囉！。

找到 x y 之後，就跟第四個子題一樣囉！

Is Prime :

第一個子題：

迴圈從 2~n-1跑一次，看看有沒有餘數是0的囉！

第二個子題：

迴圈從 2 ~√n 跑一次，看看有沒有餘數是0的囉！

Why?如果n有非1跟本身的因數，那必定存在任意兩因數 p,q 使得 $p \times q = n$，其中 $p \le q$。根據這個性質可推得，若n不是質數，必存在一個因數 f 且 $1 < f \le \sqrt{n}$。

第三個子題：

從測資的大小來看，第二個子題的方法似乎不太行。

$10^3 \times 10^6 = 10^9$ 。有點大（我的judge比較菜一點），不過就算judge比較快，題目只要多加一個 0 還是一樣的。

那這題要怎麼做？ 因為是判斷多個質數，所以當然是先建質數表囉！

因為輸入只有10^{12} 所以質數表建到10^6 就可以了。

問題是，怎麼建質數表？Sieve of Eratosthenes Algorithm

詳見AC Code或是自行 Google

找到之後 根據質數定理 $\pi(x) \approx \frac{x}{ln\,x}$

每找一個質數時間就可以降成 $\frac{\sqrt{p}}{ln\sqrt{p}} \approx 7.2 \times 10^4$

找1000個就是 7.2×10^7

實際上時間複雜度是 7.2×10^7 + 建質數表的時間，如果質數表建的不夠快還是不行的。

第四個子題：

從測資的大小來看，第三個子題的方法似乎也不行了。

怎麼辦呢？

使用 Miller - Rabin 演算法，由於這個演算法講解起來過於複雜，請大家自行Google不過建議數學好的人再去看就好，因為網路上的一些術語對於沒有數學底子的墳誒長難以理解。

這邊只要注意一件事，因為測資很大，不能直接乘，會OverFlow，可以利用類似快速冪的方式去做快速乘法，這樣就不會Overflow了。

或是你可以研究一下 C++的 int_128。

Sort :

第一個子題：

只有五個，可以if else 解決。

第二個子題：

可以用bubble sort , selection sort , insert sort 解決。

第三個子題：

可以用C++ sort , 或是 qsort 解決。

第四個子題：

問題來了，這題看似 O(nlogn)可以解決的測資大小，為何會TLE呢？

原因是因為，輸入輸出本身很慢，而且慢很多（大三上作業系統），可以自己實測，sort本身花了約0.5~0.6秒，但是輸入輸出總共花了約0.6~0.8秒，就因為這個輸入輸出導致最後TLE。

想要解決這個問題，兩個方法，一個改進IO一個改進演算法。

改進演算法：這題 a_i 的範圍介在1~100直接開一個大小為101的陣列去計算數量就好了啊！這樣時間複雜度從O(nlogn)變成O(n)了！（題目原本的正解）。

改進 IO，用Line buffer IO 讀入（fgets + strtok），快約0.2~0.4秒（大二下系統程式）。

PS. 當n越大sort影響力會比IO來的大。雖然看起來，好像是第四個子題才出現IO的問題， 其實在前幾個中，IO影響力比sort還大（常數比logn帶來的影響多），只是n都太小，看不出來。當n越大的時候，logn的影響力就會越來越明顯。在第四個子題時，只是兩者同時體現出來而已，並不代表IO影響力比sort大。