We will learn how to factorise a number in O(logN) time complexity . We know that any number can be represented as multiplication of primes . Let us suppose that we want to prime factorise a number N . And we have a array pf[] where pf[i] indicated any one prime factor of i . Now we have a prime factor of N i.e. pf[N] . Now we will keep on dividing N by pf[N] until it becomes indivisible by pf[N] . After all this division let the number N became some x where (x<N) . We will keep on dividing x by pf[x] i.e. repeat the same procedure that we did for N until we are left with N as 1 . Now the only thing that is a problem is how to compute pf[x] , right? Well that's easy , We can obtain that from sieve algorithm

(If you have not yet read sieve algo then please read it first) . Here is pseudocode for that.

```
bool prime[100010];                 //  prime[i] is true if  i   is a prime number and otherwise false
int pf[100010];                          // pf[i] indicates any one prime number of i

  void sieve(){
     fill(prime,prime+100010,true);
     prime[0]=prime[1]=false;

     for(lli i=2;i<100010;i++){
            if(prime[i]){
                 for(lli j=i*i;j<100010;j+=i)
                   {prime[j]=false;  pf[j]=i ; }          // i is the prime number that divides j
            }
      }
  }
```

In this way we can precompute the pf array . Here is pseudocode for prime factorisation .

We have a number N whose prime factors we want to find and a vector v in which all prime factors of N will be stored .

```
While(N>1){
        v.push_back(pf[N]);
        int y=N;
        while(y%pf[N]==0)y/=pf[N];
        N=y;
}
```

We understood how to factorise a number by this method but how it's complexity came to be O(logN) . Well it's a homework exercise for you . (hint: count how many max different prime factors can a number less than 10^6 have)

There is a limitation of this method that we have to precompute the array pf[] and there are issued with memory limit . But still this will run efficiently for 10^6 . For 10^7 the sieve itself takes NlogN time which may or may not pass the time limit (depends on how many seconds are given).

You can also find all factors of N in O(sqrtN) . It's quite an easy algorithm that's why we are not explaining it here , (Read it from here: http://www.geeksforgeeks.org/print-all-prime-factors-of-a-given-number/) .