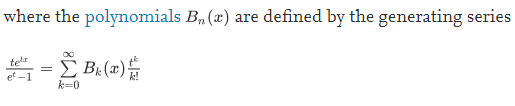
* Sperner's theorem: A family of sets in which none of the sets is a strict subset of another is called a Sperner family, or an antichain of sets, or a clutter. For example, the family of k-element subsets of an n-element set is a Sperner family. No set in this family can contain any of the others, because a containing set has to be strictly bigger than the set it contains, and in this family all sets have equal size. The value of k that makes this example have as many sets as possible is n/2 if n is even, or the nearest integer to n/2 if n is oddSperner's theorem states that these examples are the largest possible Sperner families over an n-element set.
* Number of positive integral solutions of equation 1/x+1/y=1/n

It can be rewritten as (x – n) \* (y – n) = n2 , so number of solution = number of divisors of n2

* 





i.e. Bn=Bernoulli polynomial of degree n. and Bn(k) = value of the polynomial on x = k

Or maybe you can use interpolation because it is an n + 1 degree polynomial.

Or use this code:

//x / (e^x - 1) = 1 / sum{x^i / (i + 1)!} = sum\_{i=0}^{\infty} B[i]/i! \* x^i

//so p[i] = B[i] / i!

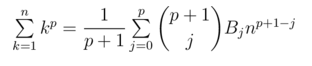
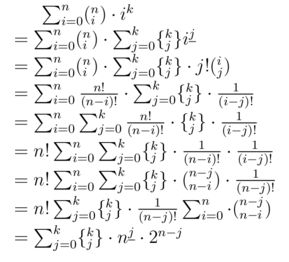
poly Bernoulli(int n) {

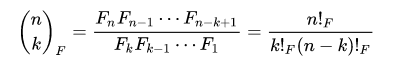
poly p(n); p.a[0] = 1;

for(int i = 1; i < n; i++) p.a[i] = p.a[i - 1] / (i + 1);

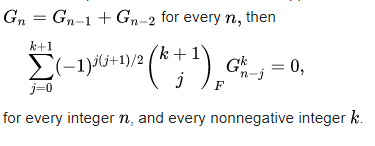
return p.inverse(n);

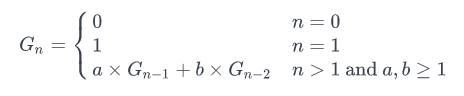
}

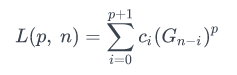
* 
*  then 
* Consider a list of all natural numbers. Now you remove all numbers whose binary representation has at least two consecutive 1 bits, from the list, hence generating a new list having elements 1,2 , 4, 5, 8 and so on. The n-th number in this list is the fibonacci base representation of n when turned into binary.
*  where n ^(j underscore) = P(n, j)
* Fibonomial coefficient

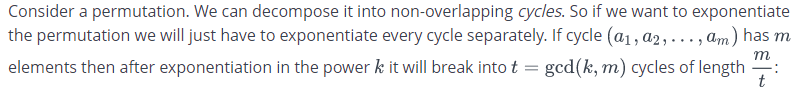


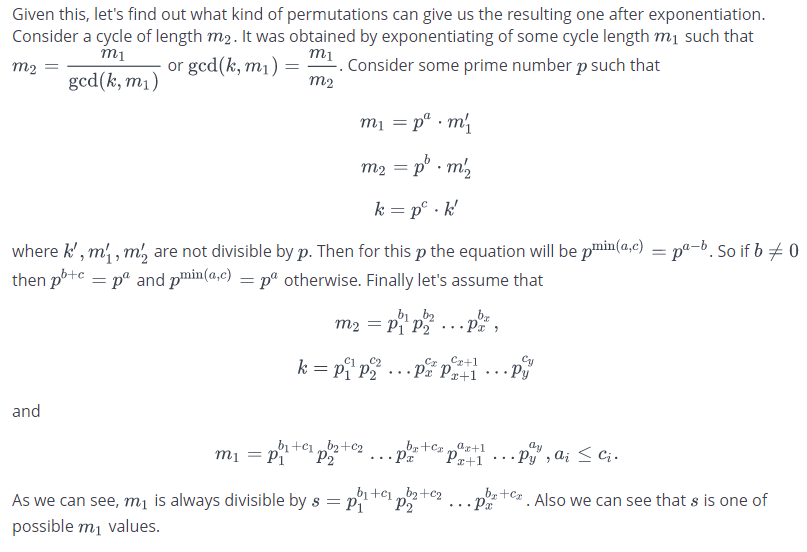
given any generalized Fibonacci sequence Gn that is, a sequence that satisfies

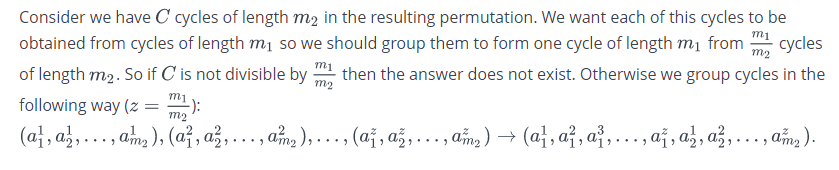


In general 

If L(p, n) is 0 for n > p then

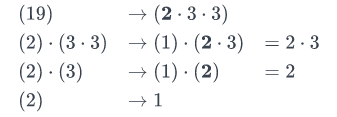
* 



* 
* Number of ways to perform n-1 swaps to form a cycle of length n is nn-2\*(n-1)!, just form a tree and take an order
* If pk = unit perm then k = lcm of all cycle lenth of p
* Number of steps it takes to reach n to 1 s.t n -> phi(n) -> phi(phi(n)) -> … -> 1

Let two[i] be the number of twos in total that all prime factors of i will generate

For example, the prime contributes 3 twos (highlighted in bold below):



Let n = p1^e1 \* p2^e2 \*…\*pk^ek, Number of steps = sum of two[pi]\*e[i]

Psedocode for two[i] ->

two[1] = 1;

// compute two[i]

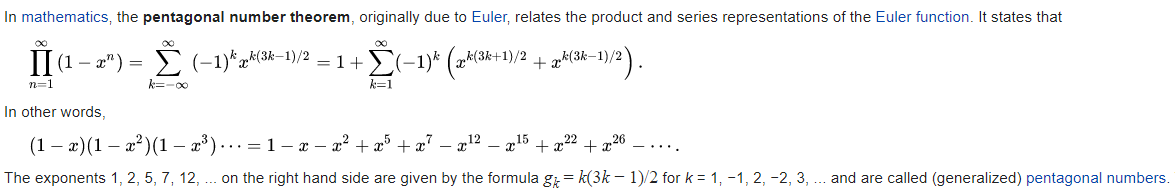
for (int j = i; j > 1;) {

int p = spf[j];

two[i] += two[p-1];

j /= p;

}

* 
* X0, x1, x2, x3, …., xn

X0 + x1, x1 + x2, x2+x3, … xn

….

If we continuosly do this n times then the polynomial of the first column of the n-th row will be

* If

Then

* If

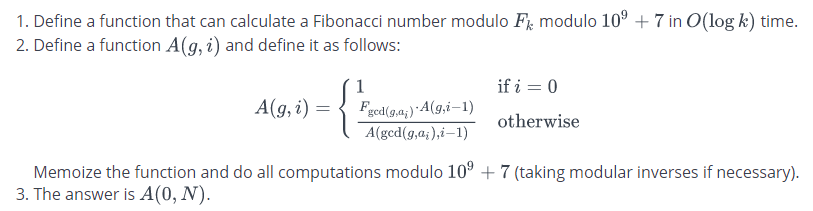
Then

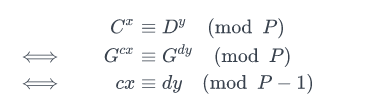
* IF
* Then,
* Number of permutations of n which form a binary heap. That means p\_1 is the root, p\_2, p\_3 are the childs of p\_1; p\_4, p\_5 are the childs of p\_2; p\_6, p\_7 are the childs of p\_3 and so on

= n! / (product of sizes of all the subtrees of the heap)

* Number of ways to choose n ids from 1 to b such that every id has distance at least k

=

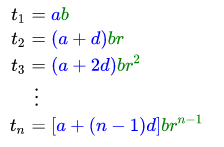
* 
* 
* Let G be th eprimitive root of a prime P

Then 

* Given two integers N and M, find how many permutations of 1, 2, ..., N (first N natural numbers) are there where the sum of every two adjacent numbers is at most M.

=  (n-k)! \* (n-k+1)^((k+1)/2) \* (n-k)^(k/2) if n < m < 2 \* n

* Arithmetico–geometric sequence

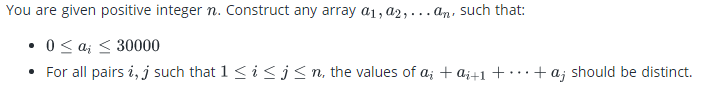


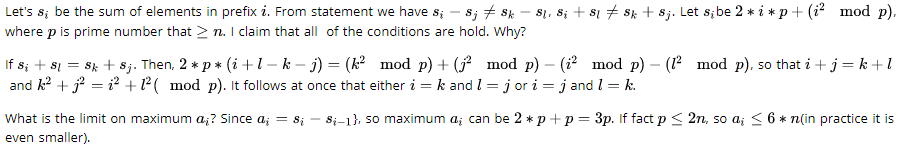
The sum of the first n terms of an arithmetico–geometric sequence has the form



* Pairs which satisfies a < (a ^ b) < b , same as a <= (a^b) <= b
* Msb of a should be on in b and msb of b should be greater than msb of a
* Pairs which satisfies a < (a ^ b) , same as a <= (a^b)
* msb of b should be greater than msb of a
* Pairs which satisfies (a ^ b) < b , same as (a^b) <= b
* Msb of a should be on in b
* Minimum of (subarray OR + subarray AND) is 2 \* min(A[i]) and maximum is 2 \* max(A[i])
* Choose exactly k disjoint subarrays s.t. their total sum is maximized.
* Select a segment (l, r) such its sum is maximum. Add it to answer and negate all indexes from l to r. Do this k times
* Involutions: permutations such that p^2 = identity perm.

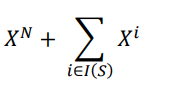


* 



* The problem is to find the expected time of the first occurrence of a given string in a random sequence(whose elements are from a given finite alphabet). In short, we have to find the position in an infinite random sequence at which the given string occurs for the first time, on average

Let X = alphabet size, N = string size,. Infix = prefix == suffix.

Then ans is 

* , compute dp[A][n], A <= 109, n <= 1000

 [.] means for all k <= n. How?

Suppose we want to calculate dp[2A][n]. Then, we consider for all possible a the sum of the values of all sequences where a of the elements are selected from 1, 2, ..., A and the remaining n - a are from i + 1, i + 2, ..., 2A.

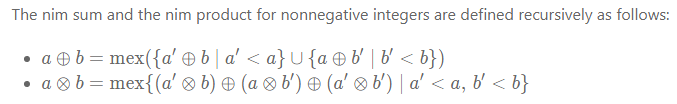


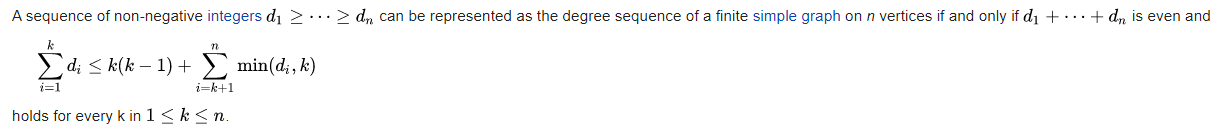


we can write A in binary and compute the answers step by step, using at most log A steps. Thus, the total time complexity is O(n2log A), which can pass.

* 
* Find a sequence of n positive integers such that product of all – sum of all = d

(n – 2) times 1, 2, n + d

* Nimber
* = = 
* Erdős–Gallai theorem

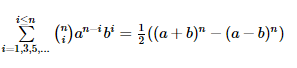
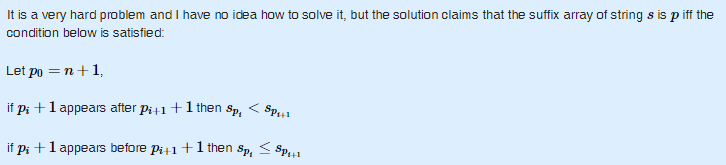


* Havel–Hakimi algorithm

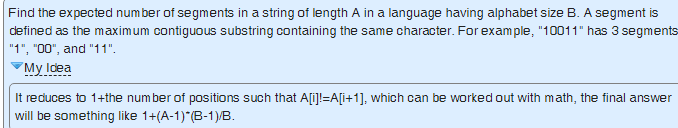
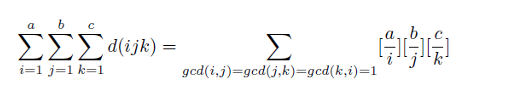
Sort the degrees, select any node and add edges with this to other nodes. In each step generally it takes the largest degree node. If you select the smallest degree node it will create a connected graph if it is possible to create a connected graph



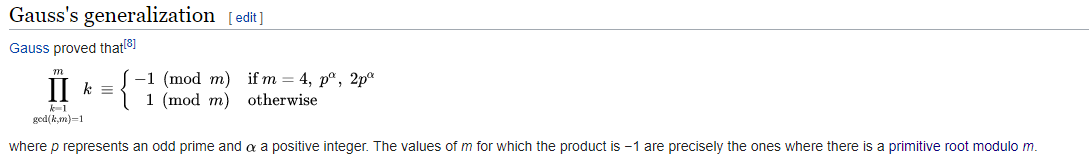


* (a|b)-b == a&(~ b)
* 
* 
* 

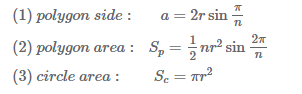
 And the solution to our problem is 

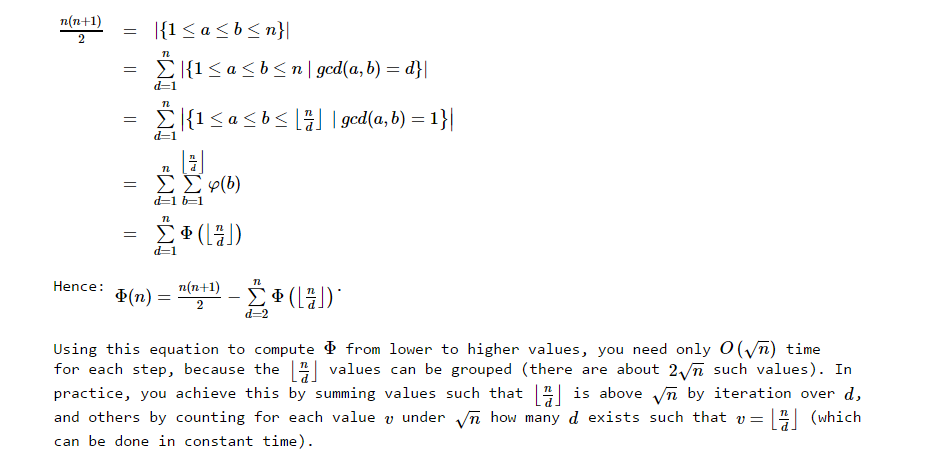
* 
* 

Here, d(x) = number of divisors of x and [a/b] = floor(a / b)

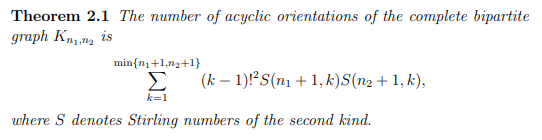
* 
* 
* Calculates the side length and area of the regular polygon inscribed to a circle.

N = number of sides, r = radius



* 

Here the function is the prefix sum of phi

* 
* Balanced Parentheses count with prefix

Find the count of balanced parentheses sequences consisting of n+k pairs of parentheses where the first k symbols are open brackets.





This is basically a convolution on Catalan. The number of variables is k+1

