L18 : Number Theory 4

1-Tut : Fibonacci Sum

Send Feedback

The fibonacci sequence is defined by the following relation:

F(0) = 0
F(1) = 1
F(N) = F(N - 1) + F(N - 2), N >= 2

Your task is very simple. Given two non-negative integers N and M (N <= M), you have to calculate and return the sum (F(N) + F(N + 1) + ... + F(M)) mod 1000000007.

Input Format :

First line of input will contain T(number of test cases), each test case follows as.
Two non-negative space-separated integers N and M. (N <= M)

Output Format :

A new line containing the answer for the each test case.

Constraints:

1 <= T <= 10^3
1 <= N <= M <= 10^18

Sample Input :

```
1
10 19
```

Sample Output :

```
10857
```

```
#include <bits/stdc++.h>
using namespace std;
#define MOD 1000000007
typedef unsigned long long ll;

void multiply(ll A[2][2], ll M[2][2])
{

   ll firstValue = A[0][0] * M[0][0] + A[0][1] * M[1][0];
         ll secondValue = A[0][0] * M[0][1] + A[0][1] * M[1][1];
         ll thirdValue = A[1][0] * M[0][0] + A[1][1] * M[1][0];
         ll fourthValue = A[1][0] * M[0][1] + A[1][1] * M[1][1];

   A[0][0] = firstValue % MOD;
   A[0][1] = secondValue % MOD;
   A[1][0] = thirdValue % MOD;
   A[1][1] = fourthValue % MOD;
}
void power(ll A[2][2], ll n)
{
    if (n == 1)
    {
        return;
    }
    power(A, n / 2);
    multiply(A, A);
    if (n % 2 != 0)
    {
        ll F[2][2] = {{1, 1}, {1, 0}};
        multiply(A, F);
    }
}
ll getFibonacci(ll n)
{
    if (n == 0 || n == 1)
    {
        return n;
    }
    ll A[2][2] = {{1, 1}, {1, 0}};
    power(A, n - 1);
    return A[0][0] % MOD;
}

int main()
{

    int t;
    cin >> t;

    while (t--)
    {
        ll m,n;
        cin >> m >> n;

        cout << (getFibonacci(n + 2)%MOD - getFibonacci(m + 1)%MOD+MOD)%MOD << endl;
    }
}
```

2-Tut : Boring Factorials

Send Feedback

Sameer and Arpit want to overcome their fear of Maths and so they have been recently practising Maths problems a lot. Aman, their friend has been helping them out. But as it goes, Sameer and Arpit have got bored with problems involving factorials. Reason being, the factorials are too easy to calculate in problems as they only require the residue modulo some prime and that is easy to calculate in linear time. So to make things interesting for them, Aman - The Mathemagician, gives them an interesting task. He gives them a prime number P and an integer N close to P, and asks them to find N! modulo P. He asks T such queries.

Input Format:

First line contains an integer T, the number of queries asked.
Next T lines contains T queries of the form “N P”. (quotes for clarity)

Output Format:

Output exactly T lines, containing N! modulo P.

Constraints:

1 <= T <= 1000
1 < P <= 2*10^9
1 <= N <= 2*10^9
Abs(N-P) <= 1000

Sample Input:

```
3
2 5
5 11
21 71
```

Sample Output:

```
2
10

6
```

```
#include<bits/stdc++.h>
#define ll long long int

using namespace std;

ll pow1(ll a, ll b, ll c)
{
    ll ans = 1LL;
    while (b > 0)
    {
        if (b & 1)
            ans = (ans * a) % c;
        a = (a * a) % c;
        b = b >> 1;
    }
    return ans;
}
int main()
{
    ll n, p, i, ans, fact;
    int t;
    cin>>t;
    while (t--)
    {
        fact = 1;
        cin>>n>>p;
        if (n >= p)
        {
            cout<<0<<endl;
            continue;
        }
        for (i = n + 1; i <= p - 1; i++)
        {
            fact = (fact * i) % p;
            if (fact == 0)
                break;
        }
        ans = pow1(fact, p - 2, p);
        cout<<p-ans<<endl;
    }
    return 0;
}
```

3-Tut : Income On Nth Day

Send Feedback

Daulat Ram is an affluent business man. After demonetization, IT raid was held at his accommodation in which all his money was seized. He is very eager to gain his money back, he started investing in certain ventures and earned out of them. On the first day, his income was Rs. X, followed by Rs. Y on the second day. Daulat Ram observed his growth as a function and wanted to calculate his income on the Nth day.

The function he found out was FN = FN-1 + FN-2 + FN-1×FN-2

Given his income on day 0 and day 1, calculate his income on the Nth day (yeah Its that simple).

Input Format:

The first line of input consists of a single integer T denoting the number of test cases.
Each of the next T lines consists of three integers F0, F1 and N respectively.

Output Format:

For each test case, print a single integer FN, as the output can be large, calculate the answer modulo 10^9+7.

Constraints:

```
1 ≤ T ≤ 10^5
0 ≤ F0, F1, N ≤ 10^9
```

Sample Input :

```
2
0 1 2
1 2 4
```

Sample Output:

```
1
107
```

Explanation

In the second test case his income on day 0 is 1 and the income on day 1 is 2. We need to calculate his income on day 4.

F0=1
F1=2
F2=1 + 2 + 1×2 = 5
F3=2 + 5 + 2×5 = 17

F4=5 + 17 + 5×17 = 107

```cpp
#include<bits/stdc++.h>
using namespace std;
#define ll long long
#define mod 1000000007

ll fib(ll n)
{
    if (n == 0 || n == 1 || n == 5)
        return n;
    if (n == 2)
        return 1;
    n--;

    ll a[2][2] = {1, 1,
                  1, 0};
    ll ans[2][2] = {
        1, 0,
        0, 1};
    ll temp[2][2];
    ll m = mod - 1, i, j, k;
    while (n)
    {
        if (n & 1)
        {
            //ans=ans*a
            for (i = 0; i < 2; i++)
```

```
                for (j = 0; j < 2; j++)
                {
                    temp[i][j] = 0;
                    for (k = 0; k < 2; k++)
                    {
                        temp[i][j] += a[i][k] * ans[k][j];
                        temp[i][j] %= m;
                    }
                }
            for (i = 0; i < 2; i++)
                for (j = 0; j < 2; j++)
                    ans[i][j] = temp[i][j];
        }
        //a=a*a
        for (i = 0; i < 2; i++)
            for (j = 0; j < 2; j++)
            {
                temp[i][j] = 0;
                for (k = 0; k < 2; k++)
                {
                    temp[i][j] += a[i][k] * a[k][j];
                    temp[i][j] %= m;
                }
            }
        for (i = 0; i < 2; i++)
            for (j = 0; j < 2; j++)
                a[i][j] = temp[i][j];
        n >>= 1;
    }
    return ans[0][0];
}
ll mpow(ll a, ll b)
{
    ll ans = 1;
    while (b)
    {
        if (b & 1)
            ans = (ans * a) % mod;
        a = (a * a) % mod;
        b >>= 1;
    }
    return ans;
}
int main()
```

```
{
    int t;
    ll a, b, c, n, i, x, y;
    cin >> t;
    while (t--)
    {
        cin >> a >> b >> n;
        if (n == 0)
            cout << a << endl;
        else if (n == 1)
            cout << b << endl;
        else
        {
            x = fib(n - 1);
            y = fib(n);
            //cout<<x<<" "<<y<<endl;
            c = mpow(a + 1, x) * mpow(b + 1, y);
            c--;
            c = c % mod;
            if (c < 0)
                c += mod;
            cout << c << endl;
        }
    }
    return 0;
}
```

4-Ass : Cubic Square

Send Feedback

Varun is learning method of successive squaring so that he can calculate a^b mod m quickly. To give himself practice he wrote many tuples of a, b and m and went to school thinking that he will do it after school.

After school he found that tuples he wrote are modified by his little sister. His sister converted each b into base 3. Varun wrote everything in base 10.

Help Varun to do his exercise.

Input Format:

First line of input contains a number T(number of test case). Each test case contains an integer a(base 10) followed by a string b (base 3) followed by integer m (base 10). All are space-separated.

Output Format;

Output a number for each test case a^b mod m in base 10 in new line.

Constraints:

1 <= T <= 1000

1 <= a, m <= 10^9

Number of digits in b will be less than 250.

Sample Input:

```
2
2 10 10
3 21101 19
```

Sample Output:

```
8

3
```

```
#include<bits/stdc++.h>
using namespace std;
int power(long long int a, string b, int m)
{
    int len = b.length();
    long long int final_ans = 1;
    for(int i=len-1;i>=0;i--)
    {
        if(b[i] == '0')
        {
            a %= m;
            a = (((a*a)%m)*a)%m;
        }
        if(b[i] == '1')
        {
            final_ans = final_ans * a;
            final_ans %= m;
            a = (((a*a)%m)*a)%m;
        }
        if(b[i] == '2')
        {
            final_ans = ((final_ans * a)%m) * a;
            final_ans %= m;
            a = (((a*a)%m)*a)%m;
        }
    }
    return final_ans;
}
int main()
{
    long long int t;
    cin >> t;
    //write code here
    while(t--)
    {
        long long int a; //in base 10
        string b; //in base 3
        int m; //in base 10
        cin >> a >> b >> m;

        cout << power(a,b,m) << endl;
    }

    return 0 ;

}
```

5-Ass : GCD Extreme

Send Feedback

Given the value of N, you will have to find the value of G. The meaning of G is given in the following code

```
G=0;
for(i = 1 ; i < N ; i++)
  for(j = i+1 ; j <= N ; j++)
     G+=gcd(i,j);
```

Here gcd() is a function that finds the greatest common divisor of the two input numbers.

Input Format:

The first line of input will contain T(number of the test case). Each test case contains an integer N.

Output Format:

For each test case print the answer in a new line.

Constraints:

1 <= T <= 10^5

1 <= N <= 10^5

Sample Input:

```
3
10
100
3
```

Sample Output:

```
67
13015

3
```

```

#include <bits/stdc++.h>

using namespace std;

#define endl '\n'
```

```
#define ll long long int
#define MAX 1000001

ll phi[MAX];
unsigned ll S[MAX], G[MAX];

int main()
{
            unsigned ll i, j, n;
    phi[1] = 1;
    for (i = 2; i < MAX; i += 2)
    {
        phi[i] = i / 2;
    }
    for (i = 3; i < MAX; i += 2)
    {
        if (!phi[i])
        {
            phi[i] = i - 1;
            for (j = i << 1; j < MAX; j += i)
            {
                if (!phi[j])
                {
                    phi[j] = j;
                }

                phi[j] = phi[j] / i * (i - 1);
            }
        }
    }
    for (i = 0; i < MAX; i++)
        S[i] = phi[i];
    for (i = 2; i < MAX; i++)
    {
        for (j = 2; j * i < MAX; j++)
        {
            S[i * j] += j * phi[i];
        }
    }
    G[1] = 0;
    for (i = 2; i < MAX; i++)
        G[i] = G[i - 1] + S[i];

    ll t;
    cin>>t;
    while (t--)
    {
        cin >> n;
        if (n == 0)
            break;
        cout << G[n] << endl;
    }

    return 0;
}
```

6-Ass : Sanchit And Nuclear Reactor

Send Feedback

We all know Sanchit Lee Cooper who is a Caltech theoretical physicist. He has eccentric and arrogant behavior. Due to his belief that he's intellectually superior, he's not ashamed to insult his own friends, like Howard, who is an engineer and not a real scientist. But nobody messes with an engineer. So Howard accepted an challenge from Sanchit. Sanchit was involved in numerous experiments as a wunderkind, such as his plan for building his own nuclear reactor - a plan stopped by government. So Sanchit presented Howard with a problem about his own nuclear reactor. It contains a large tank and at each second an atom is introduced in the tank which reacts with already existing atoms and produces some energy. Also he defined a special threshold number for his reactor called Cooper number m which is always a prime number. Energy output is defined as previous energy output of the tank multiplied by number of atoms present in it. But due to some special condition of the tank, all atoms attains stable state when number of atoms are multiple of Cooper number and no new reaction occurs. Energy output in this case is same as previous case. Also initial energy of the reactor is 1 and initially there is no atom in the tank. Now Sanchit ask Howard to tell the energy output after time T. But sadly Howard is not able to solve it and ask for your help.

Input Format

The first line of input contains T(number of the test case), each test case follows as.
contian two space-separated integers N and M where M is a prime number.

Output Format

You have to determine the energy output after time T. As the number can be quite large so output it modulo Cooper number m.

Constraints:

1 <= T <= 100
1 <= N <= 10^18
1 <= M <= 10^4

Sample Input

```
2
1 5
```

2 5

Sample Output

1
2

Explanation

After 1 seconds, there is only 1 atom in the tank. Hence energy output is 1. After 2 seconds, there are 2 atoms which reacts to give energy output of 2.

```
#include<bits/stdc++.h>
using namespace std;

#define endl '\n'
#define ll long long int

long long fact(ll a, ll m){
        long long res=1;
        for(ll i=2;i<=a;i++)
                res=(res%m * i%m)%m;
        return res;
}
int main(){

        ll n;
        cin>>n;
        while(n--){
                ll t;
                ll m;
                cin>>t>>m;
                if(t<m){
                        if(m-t==1){
                                cout<<1<<endl;
                                continue;
                        }
                        else{
                                cout<<fact(t, m)%m<<endl;
                                continue;
                        }
                }
                else{
                        ll last = t%m;
                        ll facto=fact(last, m)%m;

                        if((t/m)%2==0)
                                cout<<facto<<endl;
```

```
            else
                cout<<((m-1)%m*facto%m)%m<<endl;
        }
    }
    return 0;
}
```

7-Ass : Innocent Swaps and His Emotions

Send Feedback

There are only three phases in Swaps life: Sleep, Play and Study. Also, there are two types of emotions Swaps experiences: Happy and Sad. Each phase of his life brings either kind of emotions.

The sleep and the play phase makes Swaps happy whereas the study phase makes him sad. Quite obvious, isn't it? But we know that life isn't that great, one cannot be happy all the time.

Swaps, being a very sensitive guy, doesn't like to mix his emotions on a particular day. So each day, he is in exactly one of the three phases.

Given N which denotes the number of days and K which denotes the exact number of days Swaps needs to be happy out of these N days, can you tell him in how many ways can he achieve this? Since the output value can be very large, take modulo with 1000000007(10^9+7)

Input Format:

The first line of the input contains T, denoting the number of test cases.
The next T lines contain two space-separated integers N and K.

Constraints:

1 <=T <= 10^5
1<= K <= N <= 10^6

Output Format:

For each test-case, output a single integer, the number of ways modulo 1000000007(10^9+7).

Sample Input 1:

```
3
1 1
2 1
3 2
```

Sample Output 2:

```
2
4
12
```

Explanation

In the first test case, he needs to feel joyful on Day 1. Hence, answer is 2 (He can either play video games or sleep).

In the second test case, he can be joyful either on Day 1 or Day 2. So number of ways = 4.

```
#include <bits/stdc++.h>
using namespace std;
```

```
#define ll long long
#define MAX 1000001
#define mod 1000000007
ll fact[MAX];
ll modexpo(ll a, ll b)
{
    ll ans = 1;
    while (b)
    {
        if (b & 1)
            ans = (ans * a) % mod;
        a = (a * a) % mod;
        b >>= 1;
    }
    return ans;
}
int main()
{
        ll n,
        i, t, k, ans;
    fact[0] = 1;
    for (i = 1; i < MAX; i++)
    {
        fact[i] = i * fact[i - 1];
        if (fact[i] >= mod)
            fact[i] %= mod;
    }
    cin >> t;
    while (t--)
    {
        cin >> n >> k;
        if (k > n)
        {
            cout << endl;
            continue;
        }
        ans = modexpo(2, k);
        ans = (ans * fact[n]) % mod;
        ans = (ans * modexpo(fact[k], mod - 2)) % mod;
        ans = (ans * modexpo(fact[n - k], mod - 2)) % mod;
        cout << ans << endl;
    }
    return 0;
}
```

8-Ass : Sehwag and ETF

Send Feedback

Sehwag has been solving a lot of mathematical problems recently. He was learning ETF (Euler Totient Function) and found the topic quite interesting. So, he tried solving a question on ETF. He will be given two numbers L and R. He has to find the probability that the ETF of a number in the range [L, R] is divisible by a number K.

Input Format:

The first line contains an integer T, representing the number of test cases.
The next T lines will contain three integers L, R and K.

Constraints:

1 <= T <= 10
1 <= L <= R <= 10^12
0 <= R - L <= 10^5
1 <= K <= 10^6

Output Format:

Print the answer in a new line after rounding off the first 6 digits after the decimal place.

Sample Input 1:

```
3
1 4 2
2 5 2
3 10 4
```

Sample Output 1:

```
0.500000
0.750000

0.375000
```

```
#include<bits/stdc++.h>
using namespace std;

#define ll long long
#define MAX 1100001
#define ss(n) scanf("%lld", &n)

bool primes[MAX];
ll phi[MAX];
ll p[300001];
ll num[MAX];
int main()
{
    ll t, a, b, k, i, j, x, base;
    cin >> t;
    for (i = 3; i < MAX; i += 2)
```

```
        primes[i] = 1;
    primes[2] = 1;
    for (i = 3; i * i < MAX; i += 2)
    {
        if (primes[i])
        {
            for (j = i * i; j < MAX; j += 2 * i)
                primes[j] = 0;
        }
    }
    p[0] = 2;
    ll c = 1;
    for (i = 3; i < MAX; i += 2)
    {
        if (primes[i])
            p[c++] = i;
    }
    while (t--)
    {
        cin >> a >> b >> k;
        if (k == 1)
        {
            std::cout << std::fixed;
            std::cout << std::setprecision(6) << 1 << endl;
            continue;
        }
        for (i = a; i <= b; i++)
        {
            phi[i - a] = i;
            num[i - a] = i;
        }
        for (i = 0; p[i] * p[i] <= b; i++)
        {
            base = a / p[i] * p[i];
            while (base < a)
                base += p[i];
            while (base < p[i])
                base += p[i];
            if (base == p[i])
            {
                // cout<<phi[base-a]<<endl;
                base += p[i];
            }
            for (j = base; j <= b; j += p[i])
            {
                while (num[j - a] % p[i] == 0)
                    num[j - a] /= p[i];
                phi[j - a] -= phi[j - a] / p[i];
                //cout<<j<<" "<<phi[j-a]<<endl;
            }
        }
        for (i = a; i <= b; i++)
        {
            if (num[i - a] > 1)
                phi[i - a] -= phi[i - a] / num[i - a];
            num[i - a] = 1;
        }
        ll c = 0;
        for (i = a; i <= b; i++)
        {
            //cout<<i<<" "<<phi[i-a]<<endl;
            if (phi[i - a] % k == 0)
                c++;
        }
        double ans = c;
        ans /= (b - a + 1);
        std::cout << std::fixed;
        std::cout << std::setprecision(6) << ans << endl;
    }
    return 0;
}
```

Understanding Purpose : Nth fib number optimized

```
#include<bits/stdc++.h>
using namespace std;


void multiply(int A[2][2],int M[2][2]){

        int firstValue = A[0][0] * M[0][0] + A[0][1] * M[1][0];
        int secondValue = A[0][0] * M[0][1] + A[0][1] * M[1][1];
        int thirdValue = A[1][0] * M[0][0] + A[1][1] * M[1][0];
        int fourthValue = A[1][0] * M[0][1] + A[1][1] * M[1][1];

        A[0][0] =firstValue;
        A[0][1] = secondValue;
        A[1][0] = thirdValue;
        A[1][1] = fourthValue;

}
void power(int A[2][2],int n){
        if(n==1){
                return;
        }
        power(A,n/2);
        multiply(A,A);
        if(n%2 !=0){
                int F[2][2] = {{1,1},{1,0}};
                multiply(A,F);
        }
}
int getFibonacci(int n){
        if(n==0 || n==1){
                return n;
        }
        int A[2][2] = {{1,1},{1,0}};
        power(A,n-1);
        return A[0][0];
}
int main(){
        int n;
        cin >> n;
        cout << getFibonacci(n)<<endl;
        return 0;
}
```