Domains Contests Rank Leaderboard Jobs

1

All Contests > Game Theory > Day 2: Tower Breakers, Revisited!

Day 2: Tower Breakers, Revisited!

locked

by forthright48

Problem Submissions Leaderboard Discussions Editorial

Editorial by forthright48

Imagine each tower as a Nim pile which has a Nimvalue equal to the number of prime factors of h_i. Reducing a tower to its divisor is the same as taking away a non-zero prime factor from it.

Thus, this game is the same as a Nim game and our answer is the XOR of all Nim piles. If the Nim sum is 0, then player 2 wins; otherwise, player 1 wins.

Statistics

Difficulty: Medium
Time Complexity: O(N) + Precalculation
Required Knowledge: Nim Game
Publish Date: Mar 11 2016

Set by forthright48

Problem Setter's code :

C++

```
/***********Template Starts Here***********/
#include <bits/stdc++.h>

#define pb push_back
#define nl puts ("")
#define sp printf ( " " )
#define phl printf ( "hello\n" )
#define ff first
#define ss second
#define POPCOUNT __builtin_popcountll
#define RIGHTMOST __builtin_ctzll
#define LEFTMOST(x) (63-__builtin_clzll((x)))
#define MP make_pair
#define FOR(i,x,y) for(vlong i = (x) ; i <= (y) ; ++i)
#define ROF(i,x,y) for(vlong i = (y) ; i >= (x) ; --i)
#define CLR(x,y) memset(x,y,sizeof(x))

using namespace std;

typedef long long vlong;
typedef unsigned long long uvlong;

const vlong inf = 2147383647;
const double pi = 2 * acos ( 0.0 );
const double eps = 1e-9;

#define SIZE 1000010
vector<int> prime;
char stat[SIZE];

void sieve( int n ) {
    prime.pb ( 2 );

    stat[0] = stat[1] = 1;
    for ( int i = 4; i <= n; i += 2 ) {
        stat[i] = 1;
    }

    int sqrtn = sqrt ( n );
    for ( int i = 3; i <= sqrtn; i += 2 ) {
        if ( stat[i] == 0 ) {
            for ( int j = i * i; j <= n; j += 2 * i ) stat[j] = 1;
        }
    }

    for ( int i = 3; i <= n; i += 2 ) if ( stat[i] == 0 ) prime.pb ( i );

}

int nim[SIZE];

int factorize ( int n ) {
    int res = 0;
    int sqrtn = sqrt ( n );

    for ( int i = 0; i < prime.size() && prime[i] <= sqrtn; i++ ) {
        if ( stat[n] == 0 ) break;

        if ( n % prime[i] == 0 ) {
            int temp = 0;
            while ( n % prime[i] == 0 ) {
                temp++;
                n /= prime[i];
            }
            res += temp;
            sqrtn = sqrt ( n );
        }
    }

    if ( n > 1 ) {
        res += 1;
    }

    return res;
}

void precal() {
    FOR(i,1,SIZE-1){
        nim[i] = factorize ( i );
    }

}

void solution() {
    int kase;
    scanf ( "%d", &kase );

    while ( kase-- ) {

        int res = 0;
        int n;
        scanf ( "%d", &n );

        while ( n-- ) {
            int t;
            scanf ( "%d", &t );

            t = nim[t];

            res ^= t;
        }

        if ( res ) printf ( "1\n" );
        else {
            printf ( "2\n" );
        }
    }
}

int main () {
    sieve( SIZE - 1 );
    precal();

    solution();

    return 0;
}
```

Tested by alllleksssa

Problem Tester's code :

C++

```
#include<stdio.h>
#include<algorithm>

using namespace std;

const int maxi=1e6+5;

int cnt[maxi],b[maxi];
int p,q,ans,n,t,tmp,xs,m;

void solve()
{
  scanf("%d",&n);

   xs=0;
  for (int i=0;i<n;i++)
   {
       scanf("%d",&p);
       xs=xs^cnt[p];
    }

    if (xs==0) printf("2\n"); else printf("1\n");
}

void sito()
{
    cnt[1]=0;

    for (int i=2;i<maxi;i++)
    {
        if (b[i]==0)
        {
          for (int j=i;j<maxi;j+=i)
          {
              m=j;
              b[j]=1;
              while (m%i==0)
              {
                cnt[j]++;
                m=m/i;
              }

          }
        }
    }
}

int main()
{
    scanf("%d",&t);

    sito();

    while (t--)
       solve();

  return 0;
}
```