January Clash '15

Challenge is over. If you participate now, you are not eligible for any job offer or prize.

Problems My Submissions Hall of Fame Analytics Judge

← Problems / A Coin Game

A Coin Game

Max. Marks: 100

The challenge is over and this problem has been moved to practice area. You can either submit your solution here or
Go to Practice Area. Also further submissions won't affect the leaderboard.

Problem Editorial My Submissions

This problem is a very interesting variation of the game of nim. We can solve this by considering several different examples and understanding, if there exists a pattern.

Let's establish some of the trivial facts and try to develop a solution on it. We use an ordered tuple (n1, n2) to represent a state in the game, where the first pile contains n1 stones and the second pile contains n2 stones.

So, as to build upon the final solution we need to abide by following conditions of game theory:

- A state X is losing only if all the reachable states from X are **winning**.
- A state X is **winning**, only if there exists some state Y that is **losing** and is reachable from X.

Trivially, (0, 0) is losing state. And since both (n1, 0) and (0, n2) can be reduced to (0, 0) in both cases, these are winning states.

(Note that whenever we don't explicitly mention, n1 and n2 are > 0. Also, reachable states refers to those states that can be reached in **exactly** one valid move)

Let us consider some of the non-trivial cases.

(1, 1) is winning.

(1, 1) -> (0, 0) {Pick 1 from each of piles and reduce to losing state}.

(1, 2) is losing.

We can reduce (1, 2) to (1, 1) {W}, (0, 1) {W} or (0, 2) {W}.
Since, all reachable states are winning, (1, 2) is losing.

(1, n2) is winning for all n2 != 2

We conclude that any (1, n2) state with n2 > 2 can be reduced to (1, 2) {L}.

(2, 3) is winning.

(2, 3) can be reduced to (1, 2) {L} in one move.

(2, 4) is NOT losing.

It might appear that (0, 0) is losing, (1, 2) is losing, we may have (2, 4) as losing but notice that (2, 4) reduces to (2, 1) which is same as (1, 2) {L}.

(3, 5) is losing

We can reduce it to: (3, 4) -> (1, 2){L}
(2, 5) {W}
(2, 4) {W}
.. and many other states which are winning.

On more analysis, we have

(4, 7) is losing

Thus we have a sequence of losing states: (0, 0), (1, 2), (2, 1), (3, 5), (4, 7), (5, 3), (6, 10), (7, 4)...

The pattern is to notice that the difference in values of n1 and n2 will increase by 1 everytime.
For each value of n1, we have a corresponding losing position at n1 + d, unless n1 is already assigned as a losing position. Refer to setter's solution for a clean implementation of this idea.

The tester came up with an interesting link showing that it is well studied academically as well: Wythoff's Game

Edit Editorial

Author Solution by ankit srivastava

```
1. /* Ankit Srivastava */
2. #include <bits/stdc++.h>
3. using namespace std;
4.
5. #define MOD                          1000000007
6. #define pb(x)                        push_back(x)
7. #define mp(x,y)                      make_pair(x,y)
```

```
#define FF                          first
#define SS                          second
#define s(n)                        scanf("%d",&n)
#define sl(n)                       scanf("%lld",&n)
#define sf(n)                       scanf("%lf",&n)
#define ss(n)                       scanf("%s",n)
#define sc(n)                       {char temp[4]; ss(temp); n=temp[
#define INF                         (int)1e9
#define LINF                        (long long)1e18
#define EPS                         1e-9
#define maX(a,b)                    ((a)>(b)?(a):(b))
#define miN(a,b)                    ((a)<(b)?(a):(b))
#define abS(x)                      ((x)<0?-(x):(x))

typedef long long ll;
typedef unsigned long long LL;
typedef pair<int,int> PII;
typedef pair<LL,LL> PLL;
typedef pair<int,PII> TRI;
typedef vector<int> VI;
typedef vector<LL> VL;
typedef vector<ll> vl;
typedef vector<PII> VII;
typedef vector<TRI> VT;

int n;
int TEST_NO;
int n1, n2;
const int MAXN = 1000006;
int loser[MAXN];

void precompute() {
        int d = 0;
        for (int i = 0; i < MAXN; ++i) {
                if(loser[i]) continue;
                loser[i] = i + d;
                  if(i + d < MAXN) loser[i + d] = i;
                d++;
        }
}
void read() {
        s(n1), s(n2);
}
void preprocess() {
        if(n1 > n2) swap(n1, n2);
}
void solve() {
        if(n2 == loser[n1]) puts("Don't Play");
        else puts("Play");
}
int main() {
        precompute();
        int t;
        s(t);
```

```
        for(TEST_NO = 1; TEST_NO <= t; TEST_NO ++) {
                read();
                preprocess();
                solve();
        }
        return 0;
}
```

Tester Solution by Shobhit Saxena

```
#include <string>
#include <vector>
#include <algorithm>
#include <numeric>
#include <set>
#include <map>
#include <queue>
#include <iostream>
#include <sstream>
#include <cstdio>
#include <cmath>
#include <ctime>
#include <cstring>
#include <cctype>
#include <cassert>
#include <limits>
#include <functional>
#include <complex>
#define rep(i,n) for(int (i)=0;(i)<(int)(n);++(i))
#define rer(i,l,u) for(int (i)=(int)(l);(i)<=(int)(u);++(i))
#define reu(i,l,u) for(int (i)=(int)(l);(i)<(int)(u);++(i))
#if defined(_MSC_VER) || __cplusplus > 199711L
#define aut(r,v) auto r = (v)
#else
#define aut(r,v) __typeof(v) r = (v)
#endif
#define each(it,o) for(aut(it, (o).begin()); it != (o).end(); ++ i
#define all(o) (o).begin(), (o).end()
#define pb(x) push_back(x)
#define mp(x,y) make_pair((x),(y))
#define mset(m,v) memset(m,v,sizeof(m))
#define INF 0x3f3f3f3f
#define INFL 0x3f3f3f3f3f3f3f3fLL
using namespace std;
typedef vector<int> vi; typedef pair<int,int> pii; typedef vector<
template<typename T, typename U> inline void amin(T &x, U y) { if(
template<typename T, typename U> inline void amax(T &x, U y) { if(

bool solve(int N1, int N2) {
        const int N = 1000000;
        if(N1 > N || N2 > N) exit(1);
        static vector<int> a(N+1);
```

```
        if(!a[1]) {
                vector<bool> used(N * 2 + 2, false);
                int lastnum = 0;
                for(int n = 0; lastnum < N; ++ n) {
                        while(used[lastnum])
                                ++ lastnum;
                        int p = lastnum, q = p + n;
                        if(p <= N) a[p] = q;
                        if(q <= N) a[q] = p;
                        used[p] = used[q] = true;
                }
        }
        return a[N1] != N2;
}

int main() {
        int T;
        scanf("%d", &T);
        assert(1 <= T && T <= 100000);
        rep(ii, T) {
                int N1, N2;
                scanf("%d%d", &N1, &N2);
                assert(0 <= N1 && N1 <= 1000000 && 0 <= N2 && N2 <
                bool ans = solve(N1, N2);
                puts(ans ? "Play" : "Don't Play");
        }
        return 0;
}
```

RECENT SUBMISSIONS

User	Result	Time	Lang
Vamshi M...		9.4799	Java
Vamshi M...		8.9713	Java
Sumit		7.7132	C++
ROHIT KU...		1.843	C++
ROHIT KU...		1.871	C++
Akash Ku...		1.2073	C
Akash Ku...		1.2068	C
Ankit Ro...		2.0524	C++

View All

Annual Report

IIIrd Floor, Salarpuria Business Center,
4th B Cross Road, 5th A Block,
Koramangala Industrial Layout,
Bangalore, Karnataka 560095, India.

contact@hackerearth.com
+91-80-4155-4695
+1-650-461-4192