/\*

No Grupo:

Nombre y Apellidos de los miembros:

\*/

#define DOM\_JUDGE

#include <iostream>

#ifndef DOM\_JUDGE

#include <ctime>

#include <stdlib.h>

#endif

using namespace std;

/\*

(1) En caso de utilizar una generalizacion,

determinaci�n de los par�metros de la generalizaci�n.

Determina los par�metros de la generalizaci�n. Para cada par�metro

indica: (i) su tipo; (ii) su nombre; (iii) si es un par�metro

de entrada, de salida, o de entrada / salida; (iv) su prop�sito,

descrito de manera clara y concisa.

(2) An�lisis de casos:

(2.1) Casos base

(2.2) Casos recursivos

(3) En caso de utilizar una generalizacion, definici�n por inmersi�n

del algoritmo. Describe de manera clara y concisa c�mo se lleva a

cabo el algoritmo, en qu� punto o puntos se invoca a la generalizaci�n,

con qu� par�metros reales, y c�mo se genera el resultado

a partir de los devueltos por la generalizaci�n.

\*/

int min\_poder(int enemigos[], int n, int max\_duelos) {

// Punto de entrada al algoritmo: a implementar

}

#ifdef DOM\_JUDGE

const int MAX\_ENEMIGOS = 1000;

bool ejecuta\_caso() {

int enemigos[MAX\_ENEMIGOS];

int num\_enemigos;

cin >> num\_enemigos;

if (num\_enemigos == -1) {

return false;

}

else {

int max\_torneos;

cin >> max\_torneos;

for (int i = 0; i < num\_enemigos; i++) {

cin >> enemigos[i];

}

cout << min\_poder(enemigos, num\_enemigos, max\_torneos) << endl;

return true;

}

}

#endif

#ifndef DOM\_JUDGE

int min\_poder\_naif(int enemigos[], int n, int max\_duelos) {

int poder = 0;

while (!factible(enemigos, n, poder, max\_duelos)) {

poder++;

}

return poder;

}

const int MAX\_ENEMIGOS\_TEST = 20;

const int MAX\_PODER = 20;

bool chequea() {

int enemigos[MAX\_ENEMIGOS\_TEST];

int num\_enemigos = (rand() % MAX\_ENEMIGOS\_TEST) + 1;

int max\_torneos = (rand() % num\_enemigos) + 1;

for (int i = 0; i < num\_enemigos; i++) {

enemigos[i] = (rand()+1)%MAX\_PODER;

}

int resul1 = min\_poder(enemigos, num\_enemigos, max\_torneos);

int resul2 = min\_poder\_naif(enemigos, num\_enemigos, max\_torneos);

if (resul1 != resul2) {

cout << "NO FUNCIONA PARA: NUM\_ENEMIGOS:" << num\_enemigos << " MAX\_TORNEOS:" << max\_torneos << endl;

cout << "PODERES ENEMIGOS:";

for (int p = 0; p < num\_enemigos; p++) {

cout << enemigos[p] << " ";

}

cout << endl;

cout << "SOLUCION ALGORITMO NAIF:" << resul2 << " SOLUCION ALGORITMO:" << resul1 << endl;

return false;

}

else {

return true;

}

}

#endif

int main() {

#ifndef DOM\_JUDGE

srand(time(NULL));

bool ok = true;

int i = 0;

while (ok && i < 1000) {

ok = chequea();

i++;

}

if (ok) cout << "OK" << endl;

system("pause");

#endif

#ifdef DOM\_JUDGE

while (ejecuta\_caso());

#endif

}