**Maximum Gcd and Sum**

**locked**

**by [dipjal](https://www.hackerrank.com/dipjal)**

* [**Problem**](https://www.hackerrank.com/contests/w34/challenges/maximum-gcd-and-sum)
* [**Submissions**](https://www.hackerrank.com/contests/w34/challenges/maximum-gcd-and-sum/submissions)
* [**Leaderboard**](https://www.hackerrank.com/contests/w34/challenges/maximum-gcd-and-sum/leaderboard)
* [**Discussions**](https://www.hackerrank.com/contests/w34/challenges/maximum-gcd-and-sum/forum)
* [**Editorial**](https://www.hackerrank.com/contests/w34/challenges/maximum-gcd-and-sum/editorial)

** Editorial by [dipjal](https://www.hackerrank.com/dipjal" \t "_blank)**

For each number from  to , store its largest multiple that is present in array A.

N = (int) 1e6;

for(int i = 1; i <= n; ++i) {

++cntA[arr[i]]; // cnt array stores the count of each number in array A

}

for(int i = 1; i <= N; ++i) {

for(int j = i; j <= N; j += i) {

if(cntA[j]) {

lmulA[i] = j; // lmulA[i] stores the largest multiple of i, present in A.

}

}

}

Also, for each number from  to , store its largest multiple that is present in array B.

for(int i = 1; i <= n; ++i) {

++cntB[arr[i]]; // cntB array stores the count of each number in array B

}

for(int i = 1; i <= N; ++i) {

for(int j = i; j <= N; j += i) {

if(cntB[j]) {

lmulB[i] = j; // lmulB[i] stores the largest multiple of i, present in B.

}

}

}

Now, iterate for each number from  to , and find the largest number whose multiple is present in both the arrays. Lets say the number is mx.

int mx = 0;

for(int i = 1; i <= N; ++i) {

if(lmulA[i] > 0 && lmulB[i] > 0) {

mx = i;

}

}

The answer is largest multiple of mx in array A + largest multiple of mx in array B.

int answer = lmulA[mx] + lmulB[mx];

** Set by [dipjal](https://www.hackerrank.com/dipjal" \t "_blank)**

Problem Setter's code :

#include "bits/stdc++.h"

using namespace std;

const int N = 1e6 + 6;

int cnt[N];

int lmulA[N];

int lmulB[N];

int n;

int arr[N];

int brr[N];

int main() {

scanf("%d" , &n);

for(int i = 1; i <= n; ++i) {

scanf("%d" , arr + i);

}

for(int i = 1; i <= n; ++i) {

scanf("%d" , brr + i);

}

for(int i = 1; i <= n; ++i) {

++cnt[arr[i]];

}

for(int i = 1; i < N; ++i) {

for(int j = i; j < N; j += i) {

if(cnt[j]) {

lmulA[i] = max(lmulA[i] , j);

}

}

}

for(int i = 1; i <= n; ++i) {

--cnt[arr[i]];

}

for(int i = 1; i <= n; ++i) {

++cnt[brr[i]];

}

for(int i = 1; i < N; ++i) {

for(int j = i; j < N; j += i) {

if(cnt[j]) {

lmulB[i] = max(lmulB[i] , j);

}

}

}

int mx = 0;

for(int i = 1; i < N; ++i) {

if(lmulA[i] && lmulB[i]) {

mx = i;

}

}

printf("%d\n" , lmulA[mx] + lmulB[mx]);

return 0;

}

**https://hrcdn.net/s3_pub/hr-avatars/7662b7d7-bc20-4deb-b81b-98f7960a5ced/150x150.png Tested by [Birjik](https://www.hackerrank.com/Birjik" \t "_blank)**

Problem Tester's code :

#include <iostream>

#include <iomanip>

#include <cstdlib>

#include <algorithm>

#include <fstream>

#include <cstdio>

#include <cmath>

#include <cstring>

#include <string>

#include <ctime>

#include <queue>

#include <stack>

#include <vector>

#include <map>

#include <set>

#include <deque>

#include <cassert>

#include <unordered\_map>

#include <bitset>

#include <unordered\_set>

using namespace std;

#define pb push\_back

#define pp pop\_back

#define f first

#define s second

#define mp make\_pair

#define sz(a) (int)((a).size())

#ifdef \_WIN32

# define I64 "%I64d"

#else

# define I64 "%lld"

#endif

#define fname "."

typedef long long ll;

typedef unsigned long long ull;

typedef long double ld;

typedef pair < int, int > pi;

typedef pair < int, ull > pu;

const int inf = (int)1e9 + 123;

const ll infl = (ll)1e18 + 123;

const double eps = 1e-9;

const int MAX\_N = (int)1e6 + 5;

const int mod = (int)1e9 + 7;

int n;

int a[MAX\_N], b[MAX\_N];

int maxiA[MAX\_N], maxiB[MAX\_N];

void check(int x, int l, int r) {

assert(x >= l && x <= r);

}

int main() {

#ifdef DEBUG

freopen("input.txt", "r", stdin);

#endif

scanf("%d", &n);

check(n, 1, (int)5e5);

for (int i = 1, x; i <= n; i++) {

scanf("%d", &x);

check(x, 1, (int)1e6);

a[x]++;

}

for (int i = 1, x; i <= n; i++) {

scanf("%d", &x);

check(x, 1, (int)1e6);

b[x]++;

}

for (int i = (int)1e6; i > 0; i--) {

maxiA[i] = maxiB[i] = -1;

for (int j = i; j <= (int)1e6; j += i) {

if (a[j] > 0)

maxiA[i] = j;

if (b[j] > 0)

maxiB[i] = j;

}

if (maxiA[i] != -1 && maxiB[i] != -1) {

printf("%d\n", maxiA[i] + maxiB[i]);

return 0;

}

}

return 0;

}