**Mind Palaces**

Attempted by: **4037**

/

Accuracy: **60%**

/

Maximum Score: **20**

/

22 Votes

Tag(s):

Ad-Hoc, Algorithms, Easy

**PROBLEM**

**EDITORIAL**

**MY SUBMISSIONS**

**ANALYTICS**

Sherlock Holmes loves mind palaces! We all know that.

A mind palace, according to Mr. Holmes is something that lets him retrieve a given memory in the least time posible. For this, he structures his mind palace in a very special way. Let a NxM Matrix denote the mind palace of Mr. Holmes. For fast retrieval he keeps each row and each column sorted. Now given a memory X, you have to tell the position of the memory in Sherlock's mind palace.

**Input**  
Input begins with a line containing space separated N and M.  
The next N lines each contain M numbers, each referring to a memory Y.  
The next line contains Q, the number of queries.  
The next Q lines contain a single element X, the memory you have to search in Sherlock's mind palace.

**Output**  
If Y is present in Mr. Holmes memory, output its position (0-based indexing).  
Else output "-1 -1" (quotes for clarity only).

**Constraints**  
2 ≤ N,M ≤ 1000  
2 ≤ Q ≤ 1000  
-10^9 ≤ X,Y ≤ 10^9

**Note :** Large Input Files. Use faster I/O methods.

**SAMPLE INPUT**

5 5

-10 -5 -3 4 9

-6 -2 0 5 10

-4 -1 1 6 12

2 3 7 8 13

100 120 130 140 150

3

0

-2

170

**SAMPLE OUTPUT**

1 2

1 1

-1 -1

**Explanation**

The sample is self-explanatory.

**Time Limit:**1.0 sec(s) for each input file.

**Memory Limit:**256 MB

**Source Limit:**1024 KB

**Marking Scheme:**Marks are awarded when all the testcases pass.

**Allowed Languages:**Bash, C, C++, C++14, Clojure, C#, D, Erlang, F#, Go, Groovy, Haskell, Java, Java 8, JavaScript(Rhino), JavaScript(Node.js), TypeScript, Julia, Kotlin, Lisp, Lisp (SBCL), Lua, Objective-C, OCaml, Octave, Pascal, Perl, PHP, Python, Python 3, R(RScript), Racket, Ruby, Rust, Scala, Swift, Swift-4.1, Visual Basic

<https://www.hackerearth.com/practice/data-structures/hash-tables/basics-of-hash-tables/practice-problems/algorithm/mind-palaces-3/>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.IO;

namespace ConsoleApp1

{

class Program

{

static void Main(string[] args)

{

//long[][] mat =

//{

// new long[]{ 1,170, 48, 77},

// new long[]{67, 33, 22, 11},

// new long[]{55, 78, 4, 2}

//};

//long[] pos = ObtenerPosicion(mat, mat.LongLength, mat[0].LongLength, 78);

//Console.WriteLine(pos[0] + " " + pos[1]);

string[] input = Console.ReadLine().Trim().Split(' ');

long n = long.Parse(input[0]);

long m = long.Parse(input[1]);

//long[][] matriz = new long[n][];

int i = 0;

long cont\_fila = n;

Dictionary<long, long[]> dic =

new Dictionary<long, long[]>();

while (cont\_fila-- > 0)

{

long[] fila = Array.ConvertAll(Console.ReadLine().Trim().Split(' '), e => long.Parse(e));

for(int j =0; j< m; j++)

{

dic[fila[j]] = new long[] { i, j };

}

i++;

}

//Console.WriteLine("muestro la matriz");

//for (int k = 0; k < n; k++)

//{

// for (int l = 0; l < m; l++)

// {

// Console.Write(matriz[k][l] + " ");

// }

// Console.WriteLine();

//}

int queryes = int.Parse(Console.ReadLine());

while (queryes-- > 0)

{

long q = long.Parse(Console.ReadLine());

//long[] pos = ObtenerPosicion(matriz, n, m, q);

if (dic.ContainsKey(q))

{

Console.WriteLine(dic[q][0] + " " + dic[q][1]);

}

else

{

Console.WriteLine("-1 -1");

}

//Console.WriteLine(tupla.Item1 + " " + tupla.Item2);

}

Console.ReadLine();

}

}

}