**Find Prime numbers in a range**

Submissions: [8545](https://practice.geeksforgeeks.org/problem_submissions.php?pid=374)  Accuracy:

50.58%

   Difficulty: [Medium](https://practice.geeksforgeeks.org/Medium/0/0/)   Marks: 4

Show Topic Tags   

[GE](https://practice.geeksforgeeks.org/company/GE/)[Qualcomm](https://practice.geeksforgeeks.org/company/Qualcomm/)

Generate all prime numbers between two given numbers.

**Input:**

The first line of the input contains an integer T denoting the number of test cases. Then T test cases follow. Each test case consists of a single line containing two space separated integers m and n.  
**Output:**

For every test case print all prime numbers p such that m <= p <= n, space separated. Separate the answers for each test case by a new line.  
  
**Constraints:**  
1<=T<=60  
1 <= m <= n <= 100000,  
n - m <= 100000  
  
**Example:**

**Input:**

2

1 10

3 5

**Output:**

2 3 5 7

3 5

\*\* For More Input/Output Examples Use ['Expected Output'](https://practice.geeksforgeeks.org/problems/find-prime-numbers-in-a-range/0#ExpectOP) option \*\*

[Author: shubhamdhingra77](https://auth.geeksforgeeks.org/user/shubhamdhingra77/practice/)

<https://practice.geeksforgeeks.org/problems/find-prime-numbers-in-a-range/0>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

class Program

{

// function generate all prime number less then N in O(n)

static List<int> manipulated\_sieve(int N)

{

// 0 and 1 are not prime

// isprime[0] = isprime[1] = false;

List<bool> isprime = new List<bool>(N + 1);

List<int> prime = new List<int>(N + 1);

List<int> SPF = new List<int>(N + 1);

isprime.Add(false);

isprime.Add(false);

SPF.Add(0);

SPF.Add(0);

for (int i = 2; i < N + 1; i++)

{

isprime.Add(true);

SPF.Add(0);

}

// Fill rest of the entries

for (int i = 2; i < N; i++)

{

// If isPrime[i] == True then i is

// prime number

if (isprime[i])

{

// put i into prime[] vector

prime.Add(i);

// A prime number is its own smallest

// prime factor

SPF[i] = i;

}

// Remove all multiples of i\*prime[j] which are

// not prime by making isPrime[i\*prime[j]] = false

// and put smallest prime factor of i\*Prime[j] as prime[j]

// [ for exp :let i = 5 , j = 0 , prime[j] = 2 [ i\*prime[j] = 10 ]

// so smallest prime factor of '10' is '2' that is prime[j] ]

// this loop run only one time for number which are not prime

for (int j = 0;

j < (int)prime.Count &&

i \* prime[j] < N && prime[j] <= SPF[i];

j++)

{

isprime[i \* prime[j]] = false;

// put smallest prime factor of i\*prime[j]

SPF[i \* prime[j]] = prime[j];

}

}

return prime;

}

static void Main()

{

int t = int.Parse(Console.ReadLine().Trim());

List<int> sieve = manipulated\_sieve(100000);

while (t-- > 0)

{

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

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

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

for (int i = 0; i < sieve.Count; i++)

{

if (sieve[i] >= n && sieve[i] <= m)

{

Console.Write(sieve[i] + " ");

}

}

Console.WriteLine();

}

Console.ReadLine();

}

}