Define *GCPD* (Greatest Common Prime Divisor) as the largest prime number that divides both given positive integers. Your task is to find *GCPD* of the given integers a and b.

Example

* For a = 12 and b = 18, the output should be  
  greatestCommonPrimeDivisor(a, b) = 3;
* For a = 12 and b = 13, the output should be  
  greatestCommonPrimeDivisor(a, b) = -1.

Input/Output

* **[execution time limit] 3 seconds (cs)**
* **[input] integer a**

*Guaranteed constraints:*  
2 ≤ a ≤ 150.

* **[input] integer b**

*Guaranteed constraints:*  
2 ≤ b ≤ 150.

* **[output] integer**
  + *GCPD* of a and b or -1 if it doesn't exist.

<https://app.codesignal.com/challenge/2S8ZobHRrpmFn4TiC>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp1

{

class Program

{

public static SortedSet<int> primeFactors(int n)

{

// Print the number of 2s that divide n

SortedSet<int> pf = new SortedSet<int>();

while (n % 2 == 0)

{

//Console.Write(2 + " ");

pf.Add(2);

n /= 2;

}

// n must be odd at this point. So we can

// skip one element (Note i = i +2)

for (int i = 3; i \* i <= n; i += 2)

{

// While i divides n, print i and divide n

while (n % i == 0)

{

//Console.Write(i + " ");

pf.Add(i);

n /= i;

}

}

// This condition is to handle the case whien

// n is a prime number greater than 2

if (n > 2)

{

//Console.Write(n);

pf.Add(n);

}

return pf;

}

static int greatestCommonPrimeDivisor(int a, int b)

{

SortedSet<int> pfa = primeFactors(a);

SortedSet<int> pfb = primeFactors(b);

foreach(int item in pfa.Reverse())

{

if(pfb.Contains(item))

{

return item;

}

}

return -1;

}

static void Main(string[] args)

{

int a = 12;

int b = 18;

Console.WriteLine(greatestCommonPrimeDivisor(a, b));

Console.ReadLine();

}

}

}