AE00 - Rectangles

[#ad-hoc-1](http://www.spoj.com/problems/tag/ad-hoc-1)

Byteman has a collection of N squares with side 1. How many different rectangles can he form using these squares?

Two rectangles are considered different if none of them can be rotated and moved to obtain the second one. During rectangle construction, Byteman can neither deform the squares nor put any squares upon any other ones.

Input

The first and only line of the standard input contains one integer N (1 <= N <= 10000).

Output

The first and only line of the standard output should contain a single integer equal to the number of different rectangles that Byteman can form using his squares.

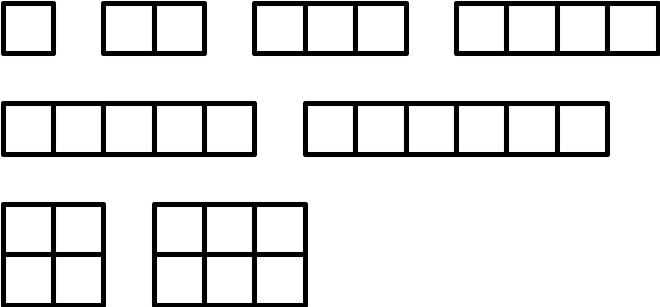
Example

For the input data:

6

the correct result is:

8



*Task author: Jakub Radoszewski.*

<http://www.spoj.com/problems/AE00/>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication1

{

class Program

{

static int cant\_n(int n)

{

if (n == 1) return 1;

HashSet<int> factores = new HashSet<int>();

int ans = 0;

for (int i = 1; i <= n / 2; i++) //pruebo hasta n/2, porque por ejemplo si n=10

// el maximo divisor es 5 = n/2

{

if (!factores.Contains(i) && !factores.Contains(n / i))

{

if (n % i == 0)

{

ans++;

/\*

agrego en el hash el factor i, para saber que ya lo probé

a cada factor se le asocia UN solo factor,

porque por ejemplo, si n=15, si el factor es 3, se le asocia 5

si el factor es 5, se le asocia 3, pero como el rctangulo de 3 \*5

es lo mismo que el de 5\*3 (uno SE VE parado y el otro acostado

pero en ESTE problema se cuentan como el mismo) hice este hash

para anotar los que ya probé

\*/

factores.Add(i);

/\*

agrego en el hash n/i, porque si probó con por ejemplo con 3\*5,

que no pruebe 5\*3 porque ES LO MISMO,

porque estaría probando con el rectangulo acostado (si lo probe en vertical)

y con el rectangulo en vertical (si lo probe acostado)

\*/

factores.Add(n / i);

}

}

}

return ans;

}

static void Main(string[] args)

{

//int a = 1, b = 1;

//int n = 6;

//int ans = 0;

//Console.WriteLine(cant\_n(12));

// Console.WriteLine(ans);

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

int ans = 0;

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

{

ans += cant\_n(i);

}

Console.WriteLine(ans);

Console.ReadLine();

}

}

}