Define *prime square* of an integer x as follows:

* if x is prime, its *prime square* is just x \* x;
* if x is not prime, its *prime square* equals the sum of composite numbers and squares of prime numbers in the range [0, x].

Given an integer n, return its *prime square*.

**Example**

* For n = 4, the output should be  
  primeSquare(n) = 18.

*Prime square* of 4 equals 1 + 2 \* 2 + 3 \* 3 + 4 = 1 + 4 + 9 + 4 = 18.

* For n = 5, the output should be  
  primeSquare(n) = 25.

5 is a prime number, so its *prime square* is just 5 \* 5 = 25.

**Input/Output**

* **[time limit] 3000ms (cs)**
* **[input] integer n**

*Constraints:*  
1 ≤ n ≤ 1000.

* **[output] integer**

The *prime square* of n.

<https://codefights.com/challenge/jHGYnJdmxD9SuJQYw/main>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication1

{

class Program

{

static bool esPrimo(int n)

{

if (n < 2) return false;

if (n == 2) return true;

if (n % 2 == 0) return false;

int sqr = (int)Math.Sqrt(n);

for (int i = 3; i <= sqr; i += 2)

{

if (n % i == 0) return false;

}

return true;

}

static int primeSquare(int n)

{

if (esPrimo(n))

{

return n \* n;

}

else

{

int sum = 0;

for (int i = 0; i <= n; i++)

{

if (esPrimo(i))

{

sum += (i \* i);

}

else

{

sum += i;

}

}

return sum;

}

}

static void Main(string[] args)

{

Console.ReadLine();

}

}

}