1. **Magic Pairs**

Houdini calls a pair of numbers **(A, B)** as a magic pair, if **A + B** is an ***odd number***.

Given a number ***N***, find out how many magic pairs can be formed using the numbers between ***1*** and ***N*** (Both inclusive).

Input Format:
Only line of input contains a single integer ***N***.

Output Format:
Print the total number of Magic Pairs.

Constraints:
1 ≤ ***N*** ≤ 10000

Sample I/O:

Input 1:
2
Output 1:
2

Input 2:
3
Output 2:
4

Input 3:
199
Output 3:
19800

Input 4:
1
Output 4:
0

Explanation:
Input 1:
Magic pairs that can be formed using the numbers between **1** and **2** (Both inclusive) are **(1, 2) and (2, 1)**. So the answer is 2.

Input 2:
We can form the following magic pairs using the numbers from **1** to **3** (Both inclusive) are **(1, 2), (2, 1), (2, 3) and (3, 2)**. So the answer is 4.

Input 4:
You cannot form a pair with a single number. So the answer is 0.

2. **Twin Primes in a Range**

Two primes p1 and p2 are called Twin Primes, if abs(p1 - p2) = 2.

Informally, if the difference between two primes p1 and p2 is exactly 2, then they are called Twin Primes.

Examples:
(3, 5)
(5, 7)
(11, 13)
(17, 19)

Given two numbers **A** and **B**, print all Twin Primes (in pairs) between **A** and **B**.

Note:
1. Always print a twin prime pair in a way such that first element of pair will be less than second element in the pair.
2. You should only print a twin prime pair, if and only if both numbers in the pair are less than or equal to the upperbound
3. For example if input is 1 and 12, you cannot print the prime pair 11 and 13, since 13 exceeds the upper bound of input.

See the Sample I/O for more clarity.

Input Format:
The only line of input contains two numbers **A** and **B**.

Output Format:
All twin prime pairs present between A and B (inclusive). Print each pair in a seperate line.

Constraints:
1 <= A, B <= 10000

Sample I/O:

Input 1:
1 20
Output 1:
3 5
5 7
11 13
17 19

Input 2:
100 200
Output 2:
101 103
107 109
137 139
149 151
179 181
191 193
197 199

Input 3:
1000 1488
Output 3:
1019 1021
1031 1033
1049 1051
1061 1063
1091 1093
1151 1153
1229 1231

1277 1279
1289 1291
1301 1303
1319 1321
1427 1429
1451 1453
1481 1483