Practice Compete Jobs Rank Leaderboard gadhiya

Dashboard > Algorithms > Dynamic Programming > Red John is Back

Badge Progress ★★★★

Points: 4727.88 Rank: 491

Red John is Back

by amititkgp

Problem Submissions Leaderboard Discussions Editorial

Red John has committed another murder. But this time, he doesn't leave a red smiley behind. What he leaves behind is a puzzle for Patrick Jane to solve. He also texts Teresa Lisbon that if Patrick is successful, he will turn himself in. The puzzle begins as follows.

There is a wall of size *4xN* in the victim's house. The victim also has an infinite supply of bricks of size *4x1* and *1x4* in her house. There is a hidden safe which can only be opened by a particular configuration of bricks in the wall. In every configuration, the wall has to be completely covered using the bricks. There is a phone number written on a note in the safe which is of utmost importance in the murder case. Gale Bertram wants to know the total number of ways in which the bricks can be arranged on the wall so that a new configuration arises every time. He calls it *M*. Since Red John is back after a long time, he has also gained a masters degree in Mathematics from a reputed university. So, he wants Patrick to calculate the number of prime numbers (say *P*) up to *M* (i.e. <= *M*). If Patrick calculates *P*, Teresa should call Red John on the phone number from the safe and he will surrender if Patrick tells him the correct answer. Otherwise, Teresa will get another murder call after a week.

You are required to help Patrick correctly solve the puzzle.

Input Format

The first line of input will contain an integer *T* followed by *T* lines each containing an integer *N*.

Output Format

Print exactly one line of output for each test case. The output should contain the number *P*.

Constraints

$1 <= T <= 20$
$1 <= N <= 40$

Sample Input

```
2
1
7
```

Sample Output

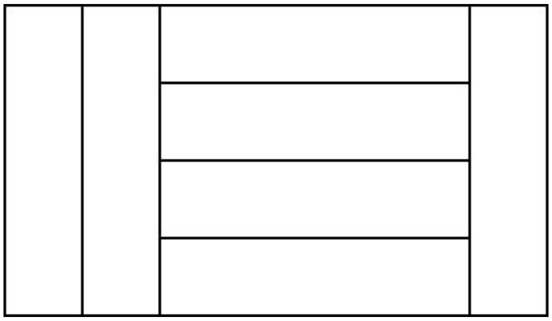
```
0
3
```

Explanation

For *N* = 1, the brick can be laid in 1 format only

The number of primes <= 1 is 0 and hence the answer.

For *N* = 7, one of the ways in which we can lay the bricks is



There are 5 ways of arranging the bricks for N = 7 and there are 3 primes <= 5 and hence the answer 3.

Submissions: 12030
Max Score: 65
Difficulty: Medium

Rate This Challenge:

☆☆☆☆☆

More

Current Buffer (saved locally, editable)

Java 8

```java
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {

        Scanner scan = new Scanner(System.in);
        int tst = scan.nextInt();

        for(int i = 0 ; i < tst ; i++){

            int N = scan.nextInt();

            if(N == 1 || N == 2 || N == 3){
                System.out.println(0);
            }
            else if(N == 4){
                System.out.println(1);
            }
            else{

                int ways = 0 ;
                int verticalBricks = N - 4;
                int horizontalBrickSet = 1;

                int totalNumOfSquares  = N / 4;

                for(int sqr = 0 ; sqr <= totalNumOfSquares ; sqr++){

                    int remainingBricks = N - sqr * 4;

                    if(sqr == 0 ){
                        ways = ways + 1;
                        continue;
                    }

                    ways += computeNcR(remainingBricks + sqr, sqr);

                }
                System.out.println(sieveOfEratosthenes(ways));
            }
        }
    }

    public static int computeNcR(int n, int r){

        int output = 1;
        int divisor = 1;

        for(int i = 0 ; i < r ; i++){

            output *= (n - i);
            divisor *= (i + 1);

        }
        return (output / divisor);
    }

     public static int sieveOfEratosthenes(int n)
    {
         int out = 0;

        boolean prime[] = new boolean[n+1];
        for(int i=0;i<=n;i++){
            prime[i] = true;
        }

        for(int p = 2; p <=Math.sqrt(n); p++)
        {
                for(int i = p*2; i <= n; i += p){
                    prime[i] = false;
                }
        }

        for(int i = 2; i <= n; i++)
        {
            if(prime[i])
                out++;
        }
         return out;
    }
}
```

Line: 83 Col: 1

Upload Code as File

□ Test against custom input

Run Code

Submit Code

Join us on IRC at #hackerrank on freenode for hugs or bugs.

Contest Calendar | Interview Prep | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy | Request a Feature