```java
package euler;

import java.io.*;
import java.util.concurrent.LinkedBlockingQueue;

/**
 * Finds the first triangle number to have n divisors
 * Project Euler Problem Number 12
 * @author sulliadfd
 */

public class TriangleNumberDivisor {
    private static int N, triangle=3, i=0, k=2, d=1;
    private static LinkedBlockingQueue<Integer> queue = new LinkedBlockingQueue<Integer>();

    /**
     * @param args
     */
    public static void main(String[] args) {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        System.out.println("I will find the first triangle number to have over n
divisors.\nn=:");
        while(true) {
            try {
                N=Integer.valueOf(br.readLine())+1;
                break;
            } catch (IOException ioe) {
                System.out.println("Incorrect Input, try again:");
            }
        }
        start();
    }

    /**
     * Starts the program.  Fills queue with actual divisors, checking size.
     * @param void
     * @return void
     */
    public static void start() {
        while(true) {
            queue.add(1);
            queue.add(triangle);
            while (d!=Math.ceil(Math.sqrt(triangle))) {
                checkTriangle(d);
                d++;
            }
            queue = new LinkedBlockingQueue<Integer>();
            d=1;
            i=0;
            k++;
            triangle+=k;
        }
    }

    /**
     * check to see if a single number should be added in the queue
     * @return void
     * @param int div
     */
    public static void checkTriangle(int div) {
        if (triangle%div==0 && !queue.contains(div)) {
            addNum(div);
        }
    }

    /**
     * adds a single number to the queue and checks its size
     * @return void
     * @param int n
     */
    public static void addNum(int n) {
        if (queue.size()!=N && queue.size()!=N+1) {
            queue.add(n);
            if (!queue.contains(triangle/n)) {
                queue.add(triangle/n);
                i++;
                if (queue.size()==N) end();
            }
            i++;
        } else end();
    }

    /**
     * Prints solution and quits.
     */
    public static void end() {
        System.out.println("Solution = "+triangle);
        System.exit(0);
    }

}
```