```java
package euler;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.util.concurrent.ConcurrentLinkedQueue;

/**
 * Finds the product of the Pythagorean triplet that sums to the input sum
 * Project Euler Problem Number 9
 * @author sulliadfd
 *
 */

public class PythagoreanTriplet {
    private static ConcurrentLinkedQueue<int[]> queue = new ConcurrentLinkedQueue<int[]>();
    private static int sum;
    private static int[] head;
    /**
     * @param args
     */
    public static void main(String[] args) {
        sum=0;
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        System.out.println("I will find the product of the Pythagorean triplet that sums to your
input (a+b+c=n where a^2+b^2=c^2):");
        while(true) {
            try {
                sum=Integer.valueOf(br.readLine());
                break;
            } catch (IOException ioe) {
                System.out.println("Incorrect Input, try again:");
            }
        }
        fillAndCheck();
        int ans[] = checkSum();
        if (ans[0]==0) {
            System.out.println("No solution");
        } else {
            System.out.println("The solution is "+ans[0]+" and the numbers were "+ans[1]+",
"+ans[2]+", "+ans[3]);
        }
    }

    /**
     *  @param null
     *  @return void
     */

    public static void fillAndCheck() {
        for (int i=1;i<sum;i++) {
            for (int j=i;j<sum;j++) {
                for (int k=j;k<sum;k++) {
                    queue.add(new int [] {i,j,k});
                }
            }
            checkTriplets();
        }
    }

    public static void checkTriplets() {
        head=queue.remove();
        while(true) {
            if ((Math.pow(head[0],2)+Math.pow(head[1], 2)==Math.pow(head[2],
2))||queue.isEmpty()) {
                break;
            } else {
                head=queue.remove();
            }
        }
        int[] current=head;
        while(!queue.isEmpty() && queue.peek()!=head) {
            if (Math.pow(current[0],2)+Math.pow(current[1], 2)==Math.pow(current[2], 2)) {
                queue.add(current);
            }
            current=queue.remove();
        }
    }

    public static int[] checkSum() {
        head=queue.remove();
        int[] current=head;
        int product[]={0,0,0,0};
        while(!queue.isEmpty()) {
            if (current[0]+current[1]+current[2]==sum) {
                product[0]=current[0]*current[1]*current[2];
                product[1]=current[0];
                product[2]=current[1];
                product[3]=current[2];
                break;
            }
            current=queue.remove();
        }
        return product;
    }

}
```