Can you reach?

There are **N** steps. In one second you can either climb 3 steps or 5 steps. You have to reach Nth step in minimum number of operations. If you reach Nth step, return minimum number of operations else return -1.

Input Format

- First line of input contains **T** number of Test cases.
- First line of each test case contains **N** number of steps.

Constraints

- **1<= T <=10^9**
- **0<= N <=10^9**

Output Format

- For each test case return minimum number of steps, if it is not possible return -1

Sample Input 0

```
3

4

6

7
```

Sample Output 0

```
-1

2

-1
```

Explanation 0

For N=4 In one second, we can reach 3 or 5 steps, 4 is less than 5. Therefore, we reach step 3. In next second, we left with only one step, so we cannot reach it

either in 3 or 5 steps return -1. **For N=6** In one second, we can reach 6 in 2 seconds. In one second 3rd step and in next second 6th step.

Sample Input 1

```
3

21

77

78
```

Sample Output 1

```
5

17

16
```

Soution in C:

```
#include <stdio.h>

#include <string.h>

#include <math.h>

#include <stdlib.h>

int steps(int n){

  if(n<0){

    return -1;

  }

  if(n%5==0){

    return n/5;

  }

  if(n%3==0){
```

```
        return 1+steps(n-3);
    }
    return (steps(n-5)<0)?-1:1+steps(n-5);
}

int main() {

    int testcases;
    scanf("%d",&testcases);
    while(testcases--){
        int n;
        scanf("%d",&n);
        printf("%d\n",steps(n));
    }
    return 0;
}
```

Solution in Java :

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

public class Solution {

    public static void main(String[] args) {
        Scanner scan= new Scanner(System.in);
```

```
        int testcases=scan.nextInt();
        while(testcases-- >0){
            int n=scan.nextInt();
            System.out.println(steps(n));
        }
    }
    public static int steps(int n){
        if(n<0){
            return -1;
        }
        if(n%5==0){
            return n/5;
        }
        if(n%3==0){
            return 1+steps(n-3);
        }
        return (steps(n-5)<0)?-1:1+steps(n-5);
    }
}
```