# Average of Averages

| | |
|---|---|
| Input file: | `standard input` |
| Output file: | `standard output` |
| Time limit: | 2 seconds |
| Memory limit: | 1024 megabytes |

*The average of averages is not always the average.*

VegiCat has just learned about segment trees and wants to practice using them.

For a positive integer $n$, he considers the sequence $a_i = i$ $(1 \leq i \leq n)$. He decides to build a segment tree to answer **range average** queries on this sequence.

In VegiCat's segment tree, each node corresponds to an interval $[l, r]$ and stores a value $v_{l,r}$, which VegiCat intends to represent the average over that interval. The root corresponds to the entire interval $[1, n]$.

However, VegiCat mistakenly assumes that a parent interval's average always equals the average of the values stored in its two child nodes. Therefore, he adds the values stored in the two child nodes and divides the sum by 2.

VegiCat first constructs the segment tree by calling Build$(1, n)$. After the segment tree has been built, he answers a query for an interval $[L, R]$ by calling Query$(1, n, L, R)$. The two recursive procedures are given in Algorithm 1.

**Algorithm 1** Build and Query

```
1:  procedure BUILD(l, r)
2:      if l = r then
3:          v_{l,l} ← a_l
4:          return
5:      end if
6:      m ← ⌊(l + r) / 2⌋
7:      BUILD(l, m)
8:      BUILD(m + 1, r)
9:      v_{l,r} ← (v_{l,m} + v_{m+1,r}) / 2
10: end procedure

11: procedure QUERY(l, r, L, R)
12:     if L ≤ l and r ≤ R then
13:         return v_{l,r}
14:     end if
15:     m ← ⌊(l + r) / 2⌋
16:     if R ≤ m then
17:         return QUERY(l, m, L, R)
18:     else if L > m then
19:         return QUERY(m + 1, r, L, R)
20:     end if
21:     x ← QUERY(l, m, L, R)
22:     y ← QUERY(m + 1, r, L, R)
23:     return (x + y) / 2
24: end procedure
```

Although VegiCat's algorithm is incorrect in general, it returns the correct average for some intervals. Count the number of such intervals.

For an interval $[L, R]$, its true average is

$$\frac{\sum_{i=L}^{R} a_i}{R - L + 1}.$$

More precisely, count the number of pairs $(L, R)$ satisfying $1 \le L \le R \le n$ such that

$$\text{Query}(1, n, L, R) = \frac{\sum_{i=L}^{R} a_i}{R - L + 1}.$$

All averages are computed as exact **rational numbers**, without rounding or truncation.

## Input

The first line contains an integer $T$ ($1 \le T \le 10^4$), the number of test cases. The description of the test cases follows.

Each of the next $T$ lines contains one integer $n$ ($1 \le n \le 10^6$), the length of the sequence.

It is guaranteed that the sum of $n$ over all test cases does not exceed $2 \cdot 10^6$.

## Output

For each test case, output an integer — the number of intervals $[L, R]$ for which VegiCat's query procedure returns exactly the true average.

## Example

| standard input | standard output |
|---|---|
| 3<br>1<br>5<br>14 | 1<br>10<br>35 |

## Note

For the second test case, $n = 5$, there are 10 valid intervals:

$$[1,1],\ [1,2],\ [2,2],\ [2,3],\ [2,5],$$
$$[3,3],\ [3,4],\ [4,4],\ [4,5],\ [5,5].$$

For example, querying $[2, 5]$ returns

$$\frac{\frac{2+3}{2} + \frac{4+5}{2}}{2} = \frac{7}{2},$$

which equals the true average of $2, 3, 4, 5$. Thus, $[2, 5]$ is counted.

In contrast, querying $[1, 5]$ returns

$$\frac{\frac{\frac{1+2}{2}+3}{2} + \frac{4+5}{2}}{2} = \frac{27}{8},$$

while its true average is 3. Thus, $[1, 5]$ is not counted.