A **bit** is a binary digit, taking a logical value of either "1" or "0" (also referred to as "true" or "false" respectively). And every decimal number has a binary representation which is actually a series of bits. If a bit of a number is "1" and its next bit is also "1" then we can say that the number has a 1 adjacent bit. And you have to find out how many times this scenario occurs for all numbers up to N.

Examples:

Number	Binary	Adjacent Bits
12	1100	1
15	1111	3
27	11011	2

Input

For each test case, you are given an integer number $(0 \leq N \leq ((2^{63})-2))$, as described in the statement. The last test case is followed by a negative integer in a line by itself, denoting the end of input file.

Output

For every test case, print a line of the form '`Case` X`:` Y', where X is the serial of output (starting from 1) and Y is the cumulative summation of all adjacent bits from 0 to N.

Sample Input

```
0
6
15
20
21
22
-1
```

Sample Output

```
Case 1: 0
Case 2: 2
Case 3: 12
Case 4: 13
Case 5: 13
Case 6: 14
```