Problem 1884: Egg Drop With 2 Eggs and N Floors

Problem Information

Difficulty: Medium
Acceptance Rate: 74.32%
Paid Only: No
Tags: Math, Dynamic Programming

Problem Description

You are given **two identical** eggs and you have access to a building with `n` floors labeled from `1` to `n`.

You know that there exists a floor `f` where `0 <= f <= n` such that any egg dropped at a floor **higher** than `f` will **break** , and any egg dropped **at or below** floor `f` will **not break**.

In each move, you may take an **unbroken** egg and drop it from any floor `x` (where `1 <= x <= n`). If the egg breaks, you can no longer use it. However, if the egg does not break, you may **reuse** it in future moves.

Return _the**minimum number of moves** that you need to determine **with certainty** what the value of _`f` is.

Example 1:

Input: n = 2 **Output:** 2 **Explanation:** We can drop the first egg from floor 1 and the second egg from floor 2. If the first egg breaks, we know that f = 0. If the second egg breaks but the first egg didn't, we know that f = 1. Otherwise, if both eggs survive, we know that f = 2.

Example 2:

Input: n = 100 **Output:** 14 **Explanation:** One optimal strategy is: - Drop the 1st egg at floor 9. If it breaks, we know f is between 0 and 8. Drop the 2nd egg starting from floor 1 and going up one at a time to find f within 8 more drops. Total drops is 1 + 8 = 9. - If the 1st

egg does not break, drop the 1st egg again at floor 22. If it breaks, we know f is between 9 and 21. Drop the 2nd egg starting from floor 10 and going up one at a time to find f within 12 more drops. Total drops is 2 + 12 = 14. - If the 1st egg does not break again, follow a similar process dropping the 1st egg from floors 34, 45, 55, 64, 72, 79, 85, 90, 94, 97, 99, and 100. Regardless of the outcome, it takes at most 14 drops to determine f.

Constraints:

* `1 <= n <= 1000`

Code Snippets

C++:

```
class Solution {
public:
int twoEggDrop(int n) {

}
};
```

Java:

```
class Solution {
public int twoEggDrop(int n) {

}
}
```

Python3:

```
class Solution:
def twoEggDrop(self, n: int) -> int:
```