Problem 1806: Minimum Number of Operations to Reinitialize a Permutation

Problem Information

Difficulty: Medium
Acceptance Rate: 72.52%
Paid Only: No
Tags: Array, Math, Simulation

Problem Description

You are given an **even** integer `n`■■■■■■. You initially have a permutation `perm` of size `n`■■ where `perm[i] == i`■ **(0-indexed)** ■■■■.

In one operation, you will create a new array `arr`, and for each `i`:

* If `i % 2 == 0`, then `arr[i] = perm[i / 2]`. * If `i % 2 == 1`, then `arr[i] = perm[n / 2 + (i - 1) / 2]`.

You will then assign `arr`■■■■ to `perm`.

Return _the minimum**non-zero** number of operations you need to perform on _`perm` _to return the permutation to its initial value._

Example 1:

Input: n = 2 **Output:** 1 **Explanation:** perm = [0,1] initially. After the 1st operation, perm = [0,1] So it takes only 1 operation.

Example 2:

Input: n = 4 **Output:** 2 **Explanation:** perm = [0,1,2,3] initially. After the 1st operation, perm = [0,2,1,3] After the 2nd operation, perm = [0,1,2,3] So it takes only 2 operations.

Example 3:

Input: n = 6 **Output:** 4

Constraints:

* `2 <= n <= 1000` * `n`■■■■■■ is even.

Code Snippets

C++:

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

}
};
```

Java:

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

}
}
```

Python3:

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