

Problem 565: Array Nesting

Problem Information

Difficulty: Medium

Acceptance Rate: 56.28%

Paid Only: No

Tags: Array, Depth-First Search

Problem Description

You are given an integer array `nums` of length `n` where `nums` is a permutation of the numbers in the range `[0, n - 1]`.

You should build a set `s[k] = {nums[k], nums[nums[k]], nums[nums[nums[k]]], ...}` subjected to the following rule:

* The first element in `s[k]` starts with the selection of the element `nums[k]` of `index = k`.*
The next element in `s[k]` should be `nums[nums[k]]`, and then `nums[nums[nums[k]]]`, and so on. * We stop adding right before a duplicate element occurs in `s[k]`.

Return _the longest length of a set_ `s[k]`.

Example 1:

Input: nums = [5,4,0,3,1,6,2] **Output:** 4 **Explanation:** nums[0] = 5, nums[1] = 4, nums[2] = 0, nums[3] = 3, nums[4] = 1, nums[5] = 6, nums[6] = 2. One of the longest sets s[k]: s[0] = {nums[0], nums[5], nums[6], nums[2]} = {5, 6, 2, 0}

Example 2:

Input: nums = [0,1,2] **Output:** 1

Constraints:

* `1 <= nums.length <= 105` * `0 <= nums[i] < nums.length` * All the values of `nums` are **unique**.

Code Snippets

C++:

```
class Solution {
public:
    int arrayNesting(vector<int>& nums) {
        }
    };
}
```

Java:

```
class Solution {
    public int arrayNesting(int[] nums) {
        }
    }
}
```

Python3:

```
class Solution:
    def arrayNesting(self, nums: List[int]) -> int:
```