Max Distance Between Same Elements

solved by	Senan
Platform	GeeksForGeeks
difficulty	Easy
tags	Array Hash Map Vector
language	C++
solved on	@10/10/2024
link	https://www.geeksforgeeks.org/problems/max-distance-between-same-elements/1
Completion	☑

Intuition

The problem requires us to find the maximum distance between two occurrences of the same element in the array. To achieve this, we can store the index of each element when it first appears in the array, and whenever the element is encountered again, calculate the distance between the current index and the stored index. The maximum of these distances is the answer.

Approach

1. Use an unordered map `mpp` to store the first occurrence of each element in the array `arr`.
2. Initialize a variable `maxi` to store the maximum distance, starting with `INT_MIN`.
3. Loop through the array `arr`. For each element:
    - If the element is already in the map (it has appeared before), calculate the distance between the current index and its first occurrence, and update `maxi` if the distance is larger.
    - If the element is not in the map, store its index.
4. After the loop, `maxi` will hold the maximum distance between two occurrences of any element.
5. Return `maxi`.

Complexity

Time Complexity:

The time complexity is **O(n)**, where `n` is the number of elements in the array. This is because we loop through the array once, and the operations on the unordered map (insertion and lookup) are average **O(1)**.

Space Complexity:

The space complexity is **O(n)** because, in the worst case, we might store every element of the array in the unordered map if there are no duplicates.

Code

```
class Solution {
  public:
    int maxDistance(vector<int> &arr) {
        unordered_map<int,int> mpp;
        int maxi = INT_MIN;

        for(int i = 0; i < arr.size(); i++){
            if(mpp.count(arr[i]))
                maxi = max(maxi, i - mpp[arr[i]]);
            else
                mpp[arr[i]] = i;
        }
        return maxi;
    }
};
```