1. Non-Repeating Element

Find the first non-repeating element in a given array **arr** of **N** integers.
Note: Array consists of only positive and negative integers and **not zero.**

Example 1:

Input : arr[] = {-1, 2, -1, 3, 2}
Output : 3
Explanation:
-1 and 2 are repeating whereas 3 is
the only number occuring once.
Hence, the output is 3.

Example 2:

Input : arr[] = {1, 1, 1}
Output : 0

Your Task:
This is a function problem. The input is already taken care of by the driver code. You only need to complete the function **firstNonRepeating()** that takes an array **(arr)**, sizeOfArray **(n)**, and **returns** the first non-repeating element. The driver code takes care of the printing.

Expected Time Complexity: O(N).
Expected Auxiliary Space: O(N).

Constraints:
$1 <= N <= 10^7$
$-10^{16} <= A^i <= 10^{16}$
$\{A_i != 0 \}$

Solution:

```python
def firstNonRepeating(arr, n):
    # Complete the function
    hash_map = {}
    for i in arr:
        if i in hash_map:
            hash_map[i] += 1
        else:
            hash_map[i] = 1
    for i in arr:
        if hash_map[i] == 1:
            return i

    return 0


n = int(input())
arr = list(map(int, input().strip().split()))
print(firstNonRepeating(arr, n))
```

Explanation:

1. The first for loop creates a hash map of the elements in the input array arr, where the key is the element and the value is the frequency of that element in the array.

2. The second for loop iterates through the input array arr again and checks the frequency of each element in the hash map.

3. If the frequency of an element is 1, the function returns that element as it is the first non-repeating element in the array.

4. If no such element is found, the function returns 0.