# Check If N And Its Double Exist

| | |
|---|---|
| solved by | Senan |
| Platform | LeetCode |
| difficulty | Easy |
| # Serial | 1346 |
| tags | Hash Map |
| language | C++ |
| solved on | @01/12/2024 |
| link | https://leetcode.com/problems/check-if-n-and-its-double-exist/ |
| Completion | ☑ |

## Intuition

The problem is to determine if there exist two indices `i` and `j` such that `arr[i] == 2 * arr[j]`. We need to identify this relationship efficiently. Using a hash set, we can store seen elements and check in constant time if the required condition holds for the current element.

## Approach

1. Traverse the array using a loop.
2. For each element `arr[i]`, check if either `2 * arr[i]` or `arr[i] / 2` (if even) exists in the set.
3. If found, return `true`.
4. Otherwise, insert the current element into the set and continue.
5. If no such pair exists, return `false`.

## Complexity

### Time Complexity:

- The loop runs in `O(n)` where `n` is the size of the array.
- Each `insert` and `count` operation on the hash set is `O(1)` on average.
- So, the overall time complexity is **`O(n)`**.

### Space Complexity:

- The space complexity is **`O(n)`** for storing the elements in the set.

## Code

```
class Solution {
public:
    bool checkIfExist(vector<int>& arr) {
        unordered_set<int> st;
        for (int i = 0; i < arr.size(); i++) {
            if (st.count(2 * arr[i]) ||
                (arr[i] % 2 == 0 && st.count(arr[i] / 2)))
```

```
                return true;
            st.insert(arr[i]);
        }
        return false;
    }
};
```