Contains Duplicate

```
class Solution {
public:
    bool containsDuplicate(vector<int>& nums) {
      int i;
      int j;
      int n = nums.size();

      for (i = 0;i<n;i++){
        for (j = i+1;j<n;j++){
          if (nums[i] == nums[j])return true;
        }
      }
      return false;
    }
};
```

- initial solution
  - time limit exceeded
  - each index of array check every other array using nested for loops
- Big O
  - O(n^2)

```
class Solution {
public:
    bool containsDuplicate(vector<int>& nums) {
      int i;
      int n = nums.size();

      unordered_map<int,int> map;

      for (i = 0;i<n;i++){
        map[nums[i]] += 1;
        if (map[nums[i]] > 1) return true;
```

```
      }

      return false;
    }
};
```

- Working solution
  - last solution was too slow due to nested for loops
  - can use hash map to store frequency of each number
  - indexing every number into the hash table as a key and incrementing the value by 1
  - if the value is greater then 1 for any number true is returned
- Big O
  - O(n)