# 题目

给定一个整数数组，判断是否存在重复元素。

如果任意一值在数组中出现至少两次，函数返回 true 。如果数组中每个元素都不相同，则返回 false 。

**示例 1:**

输入: [1,2,3,1]

输出: true

**示例 2:**

输入: [1,2,3,4]

输出: false

**示例 3:**

输入: [1,1,1,3,3,4,3,2,4,2]

输出: true

# 分析

设计出现次数的一般考虑：哈希表、位运算、排序、set（次数为1）。

## 方法一：哈希表

**思路：**判断key值是否大于1

**代码：**

class Solution {

public:

bool containsDuplicate(vector<int>& nums) {

unordered\_map <int,int>mp;

for(int i:nums)

{

mp[i]++; //i对应的value值++

if(mp[i]>1) //i对应的value值大于1，则说明存在重复元素

return true;

}

return false;

}

};

或：

class Solution {

public:

bool containsDuplicate(vector<int>& nums) {

unordered\_map<int,int> mp;

for(int i=0;i<nums.size();i++)

mp[nums[i]]++;

unordered\_map<int,int>::iterator it = mp.begin();

for( ; it!=mp.end(); it++)

{

if(it->second>=2)

return true;

}

return false;

}

};

## 方法二：集合set

**思路：**利用集合，与原数组比较大小。

**代码：**

class Solution {

public:

bool containsDuplicate(vector<int>& nums) {

unordered\_set <int> st (nums.begin(),nums.end());

return nums.size() > st.size();

//如果原数组的大小大于集合的大小，则说明存在重复元素

}

};

## 方法三：排序

**思路：**比较前后两个元素是否相等，来判断是否存在重复元素

**代码：**

class Solution {

public:

bool containsDuplicate(vector<int>& nums) {

sort(nums.begin(),nums.end());

for(int i=1;i<nums.size();i++)

{

if(nums[i]==nums[i-1])

return true;

}

return false;

}

};