# 题目

给定一个整数数组和一个整数 k，判断数组中是否存在两个不同的索引 i 和 j，使得 nums [i] = nums [j]，并且i和j 的差的绝对值至多为k。

**示例 1:**

输入: nums = [1,2,3,1], k = 3

输出: true

**示例 2:**

输入: nums = [1,0,1,1], k = 1

输出: true

**示例 3:**

输入: nums = [1,2,3,1,2,3], k = 2

输出: false

# 分析

## 方法一：哈希表

**思路：**利用hashmap, key为vector中的数, value存放对应的下标。

遍历vector, 若当前值已经存在于hashmap中，判断当前的下标与hashmap中记录的下标差是否不大于k，若是，返回true；否则，更新hashmap中该值的下标，继续下一个数。

**代码：**

class Solution {

public:

bool containsNearbyDuplicate(vector<int>& nums, int k) {

int n = nums.size(), idx = 0;

unordered\_map<int, int> nmap; // key:nums[i], value:index

for (int i = 0; i < n; ++i) {

auto iter = nmap.find(nums[i]);

if (iter != nmap.end()) {

if (i - iter->second <= k) return true;

else iter->second = i;

}

else nmap[nums[i]] = i;

}

return false;

}

};

## 方法二：集合set

**思路：**给定数组nums和常数k，存在不同的i、j使得nums[i] = nums[j]，且abs(i-j) <= k。

**代码：**

bool containsNearbyDuplicate(vector<int>& nums, int k) {

unordered\_set<int> set;

//搜索、插入和移除平均常数时间复杂度，不会超时

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

{

if(set.find(nums[i]) != set.end())

return true;

set.insert(nums[i]); //已经自动去重

if(set.size() > k )

set.erase(nums[i-k]); //滑动窗口长度最大为k

}

return false;

}

## 方法三：排序

**思路：**

**代码：**