题目地址(Delete Sublist to Make Sum Divisible By K)

https://binarysearch.com/problems/Delete-Sublist-to-Make-Sum-Divisible-By-K

入选理由

1. 同余定理+前缀和的巧妙结合

题目描述

```
You are given a list of positive integers nums and a positive integer k. Return the length of the shortest sublist (can

Constraints

1 ≤ n ≤ 100,000 where n is the length of nums
Example 1
Input
nums = [1, 8, 6, 4, 5]
k = 7
Output
2
Explanation
We can remove the sublist [6, 4] to get [1, 8, 5] which sums to 14 and is divisible by 7.
```

标签

- 前缀和
- 数组
- Math
- 哈希表

难度

- 中等

前置知识

- 哈希表
- 同余定理及简单推导
- 前缀和

思路

题目的意思是让我们移除一段**最短连续子数组**，使得剩下的数字和为 k 的整数倍。

暴力的思路仍然是枚举所有的连续子数组，然后计算连续子数组的和 sum_range。如果数组的总和 total - sum_range 是 k 的整数倍，那么我们就得到了一个备胎，遍历完所有的子数组，取备胎中最短的即可。当然如果没有任何备胎，需要返回 -1。

当然，上述方法即使加上剪枝时间复杂度也相对较高，看到被 x 整除，求余数等问题都可以尝试考虑是否可以使用数学中的同余定理，看到连续子数组就可以考虑用前缀和进行优化。

本题可以使用前缀和 + 同余定理进行优化：

- 由前缀和我们知道子数组 A[i:j] 的和就是 pres[j] - pres[i-1]，其中 pres 为 A 的前缀和。
- 由同余定理我们知道两个模 k 余数相同的数字相减，得到的值定可以被 k 整除。

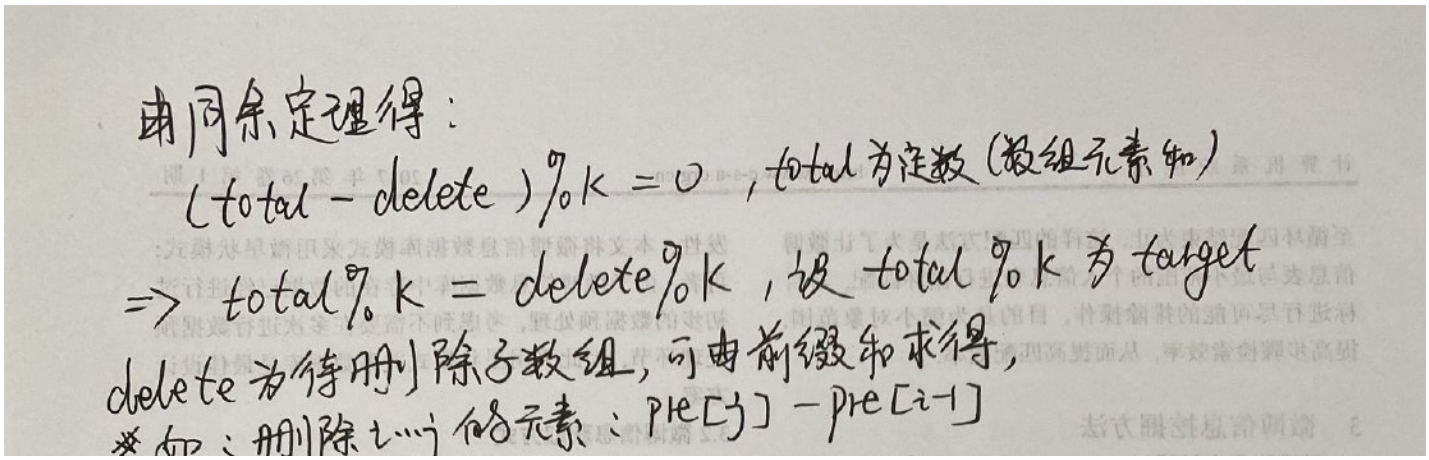
于是，我们可以将前缀和模 k 的余数 x 放到哈希表中，这个哈希表就充当了前缀和的角色，来记录最新的余数 x 对应的下标，记录最新的目的是为了找出符合要求的**最短**的连续子数组。

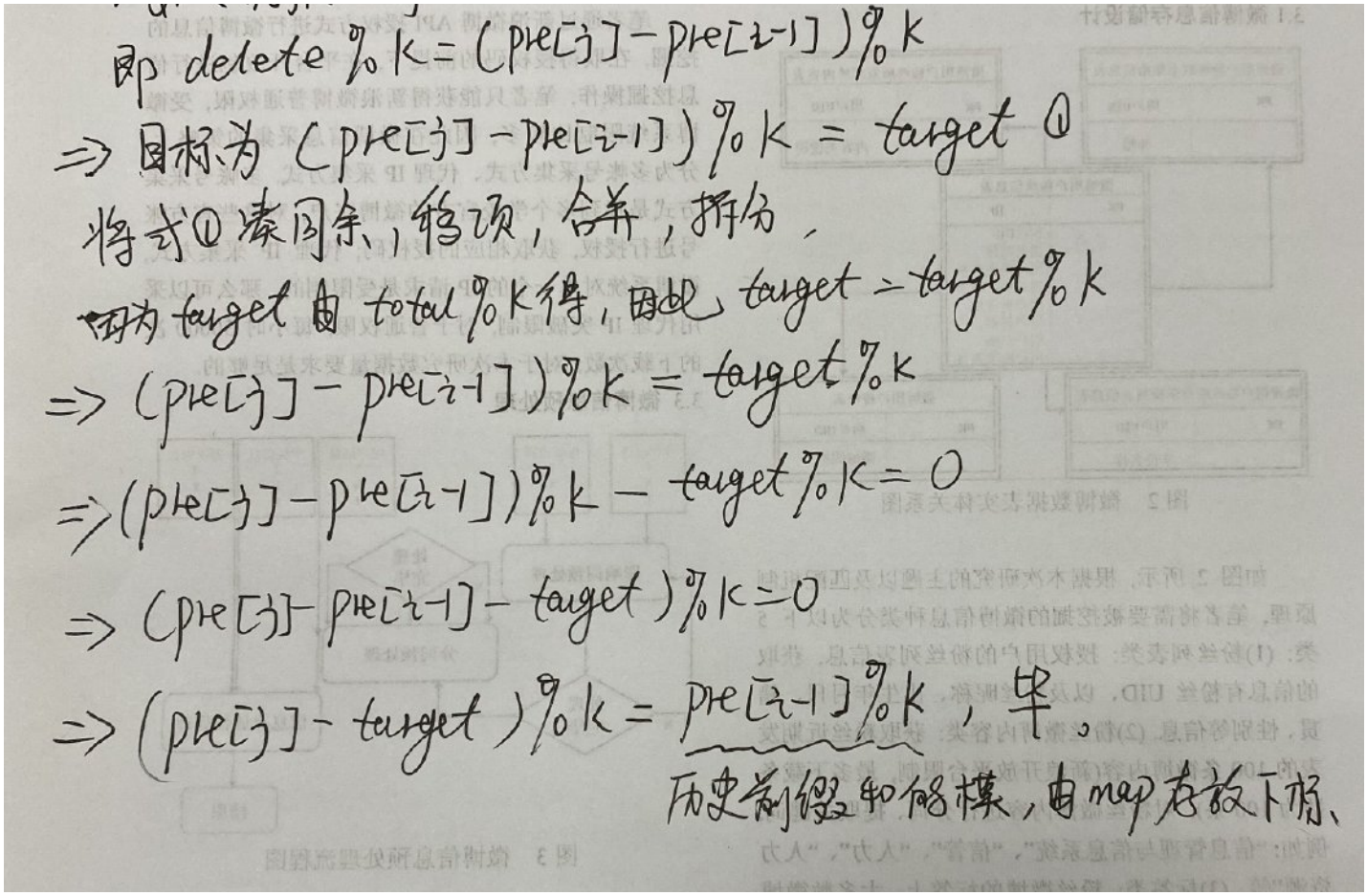
算法上，我们可以先计算出总体的数组和 total 模 k 的余数，记为 target，那么我们的目标就是**找到一段模 k 等于 target 的子数组**。注意，我们需要提前在哈希表中放一个 ：

```
{
    0: -1
}
```

以应对从数组索引 0 处开始取子数组的情况。

推导过程（理解的可跳过此部分）





代码

代码支持： Java， CPP,Python, JS

Java Code:

注：-1 % 4 为-1，而我们期望为 3，为了解决正负数求余统一，采用 Math.floorMod，等同于先+4 再模 4

即：

```
int floorMod(const int& a, const int& b)
{
    return (a + b) % b;
}
```

Java Code：

```
import java.util.*;

class Solution {

    public int solve(int[] nums, int k) {
```

```
        int tar = 0;

        for (int n : nums)
            tar += n;

        tar = Math.floorMod(tar, k);

        Map<Integer, Integer> map = new HashMap<>();
        map.put(0, -1);

        int prefix = 0, res = nums.length;

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

            prefix += nums[i];
            int mod = Math.floorMod(prefix, k);
            map.put(mod, i);

            if (map.containsKey(Math.floorMod(prefix - tar, k)))
                res = Math.min(res, i - map.get(Math.floorMod(prefix - tar, k)));
        }

        return res == nums.length ? -1 : res;
    }
}
```

C++ Code:

```
int floorMod(const int& a, const int& b)
{
    return (a % b + b) % b;
}
int solve(vector<int>& nums, int k) {
    int allSum = 0;
    for (int& num : nums)
        allSum += num;

    allSum = floorMod(allSum, k);
    unordered_map<int, int> dict;
    dict[0] = -1;

    int preSum = 0;
    int minLen = nums.size();
    for (int i = 0; i < nums.size(); i++) {
        preSum += nums[i];
        int mod = floorMod(preSum, k);
        dict[mod] = i;

        if (dict.count(floorMod(preSum - allSum, k)))
            minLen = min(minLen, i - dict[floorMod(preSum - allSum, k)]);
```

```
    }
    return minLen == nums.size() ? -1 : minLen;
}
```

Python:

```
class Solution:
    def solve(self, nums, k):
        total = sum(nums)
        mod = total % k

        ans = len(nums)
        total = 0
        dic = {0: -1}
        for j in range(len(nums)):
            total += nums[j]
            cur = total % k
            target = (cur - mod + k) % k
            if target in dic:
                ans = min(ans, j - dic[target])
            dic[cur] = j

        if ans == len(nums):
            return -1
        return ans
```

JS:

```
var floorMod = function (a, b) {
  return ((a % b) + b) % b;
};
class Solution {
  solve(nums, k) {
    var map = new Map();
    map.set(0, -1);
    var res = nums.length;
    var target = 0;
    var currSum = 0;
    for (let i = 0; i < nums.length; i++) {
      target += nums[i];
    }
    target = target % k;
    for (let i = 0; i < nums.length; i++) {
      currSum = (nums[i] + currSum) % k;
      map.set(currSum, i);
      var prevSum = floorMod(currSum - target, k);
      if (map.has(prevSum)) {
        res = Math.min(res, i - map.get(prevSum));
      }
```

```
    }
    return res === nums.length ? -1 : res;
  }
}
```

复杂度分析

令 n 为数组长度。

- 时间复杂度：$O(n)$
- 空间复杂度：$O(\min(n, k))$

相关题目（换皮题）

- **K-Divisible Sublist**

- **974. 和可被 K 整除的子数组**

- **523. 连续的子数组和**

上一页 下一页