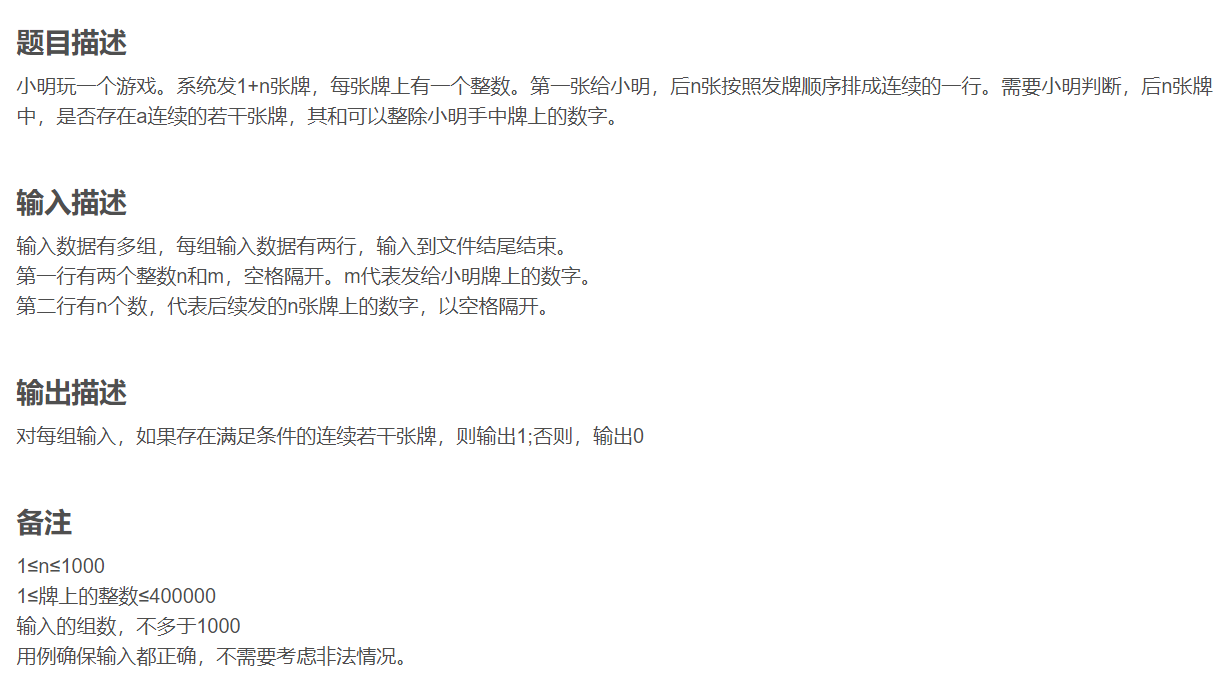
# **E卷-数字游戏[200分]（ Java | Python3 | C++ | C语言 | JsNode | Go ）**





6 7

2 12 6 3 5 5

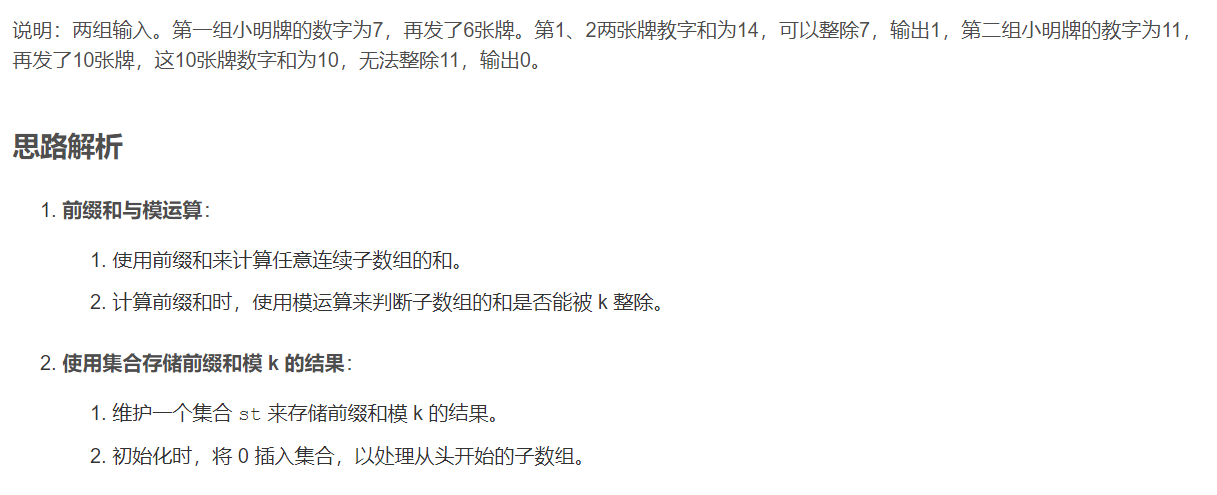
10 11

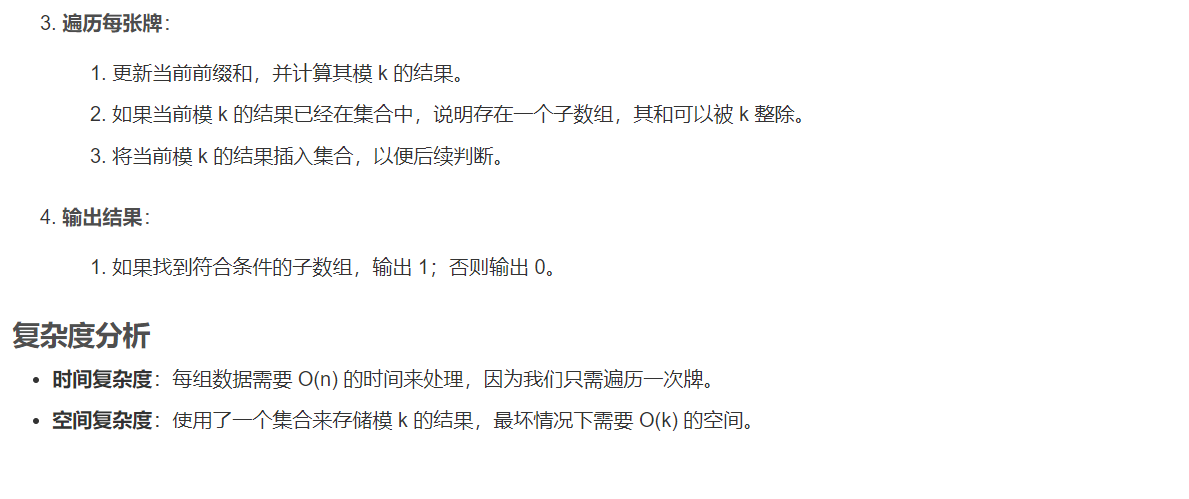
1 1 1 1 1 1 1 1 1 1



1

0







import java.util.HashSet;

import java.util.Scanner;

import java.util.Set;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// 读取每组输入数据，直到文件结束

while (scanner.hasNextInt()) {

int n = scanner.nextInt(); // 读取牌的数量n

int k = scanner.nextInt(); // 读取小明手中牌上的数字k

Set<Integer> st = new HashSet<>(); // 用于存储前缀和模k的结果

st.add(0); // 初始化，插入0以处理从头开始的子数组

int s = 0; // 当前前缀和

int ok = 0; // 标记是否存在满足条件的子数组

// 遍历n张牌

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

int x = scanner.nextInt(); // 读取当前牌上的数字

s = (s + x) % k; // 更新前缀和并取模

// 检查前缀和模k的结果是否已经出现过

if (st.contains(s)) {

ok = 1; // 如果出现过，表示存在满足条件的子数组

}

st.add(s); // 将当前前缀和模k的结果插入集合

}

System.out.println(ok); // 输出结果，1表示存在，0表示不存在

}

scanner.close();

}

}



import sys

# 读取每组输入数据，直到文件结束

for line in sys.stdin:

n, k = map(int, line.split()) # 读取牌的数量n和小明手中牌上的数字k

cards = list(map(int, sys.stdin.readline().split())) # 读取n张牌上的数字

st = set() # 用于存储前缀和模k的结果

st.add(0) # 初始化，插入0以处理从头开始的子数组

s = 0 # 当前前缀和

ok = 0 # 标记是否存在满足条件的子数组

# 遍历n张牌

for x in cards:

s = (s + x) % k # 更新前缀和并取模

# 检查前缀和模k的结果是否已经出现过

if s in st:

ok = 1 # 如果出现过，表示存在满足条件的子数组

st.add(s) # 将当前前缀和模k的结果插入集合

print(ok) # 输出结果，1表示存在，0表示不存在



#include <bits/stdc++.h>

using namespace std;

using ll = long long;

int main()

{

int n, k;

// 读取每组数据，直到文件结束

while (cin >> n >> k)

{

set<int> st; // 用于存储前缀和模k的结果

st.insert(0); // 初始化，插入0以处理从头开始的子数组

int s = 0; // 当前前缀和

int ok = 0; // 标记是否存在满足条件的子数组

// 遍历n张牌

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

{

int x;

cin >> x; // 读取当前牌上的数字

s = (s + x) % k; // 更新前缀和并取模

// 检查前缀和模k的结果是否已经出现过

if (st.count(s))

{

ok = 1; // 如果出现过，表示存在满足条件的子数组

}

st.insert(s); // 将当前前缀和模k的结果插入集合

}

cout << ok << "\n"; // 输出结果，1表示存在，0表示不存在

}

}



#include <stdio.h>

#include <string.h>

#define MAXN 1000

int main() {

int n, k;

// 读取每组输入数据，直到文件结束

while (scanf("%d %d", &n, &k) != EOF) {

int st[MAXN + 1] = {0}; // 用于存储前缀和模k的结果

st[0] = 1; // 初始化，插入0以处理从头开始的子数组

int s = 0; // 当前前缀和

int ok = 0; // 标记是否存在满足条件的子数组

// 遍历n张牌

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

int x;

scanf("%d", &x); // 读取当前牌上的数字

s = (s + x) % k; // 更新前缀和并取模

if (s < 0) s += k; // 处理负数情况

// 检查前缀和模k的结果是否已经出现过

if (st[s]) {

ok = 1; // 如果出现过，表示存在满足条件的子数组

}

st[s] = 1; // 将当前前缀和模k的结果插入集合

}

printf("%d\n", ok); // 输出结果，1表示存在，0表示不存在

}

return 0;

}



const readline = require("readline");

// 创建接口以读取标准输入

const rl = readline.createInterface({

input: process.stdin,

output: process.stdout,

});

let inputLines = [];

// 读取所有输入行

rl.on("line", (line) => {

inputLines.push(line);

});

rl.on("close", () => {

// 处理每组输入数据

for (let i = 0; i < inputLines.length; i += 2) {

const [n, k] = inputLines[i].split(" ").map(Number); // 读取牌的数量n和小明手中牌上的数字k

const cards = inputLines[i + 1].split(" ").map(Number); // 读取n张牌上的数字

const st = new Set(); // 用于存储前缀和模k的结果

st.add(0); // 初始化，插入0以处理从头开始的子数组

let s = 0; // 当前前缀和

let ok = 0; // 标记是否存在满足条件的子数组

// 遍历n张牌

for (const x of cards) {

s = (s + x) % k; // 更新前缀和并取模

if (s < 0) s += k; // 处理负数情况

// 检查前缀和模k的结果是否已经出现过

if (st.has(s)) {

ok = 1; // 如果出现过，表示存在满足条件的子数组

}

st.add(s); // 将当前前缀和模k的结果插入集合

}

console.log(ok); // 输出结果，1表示存在，0表示不存在

}

});



package main

import (

"bufio"

"fmt"

"os"

"strconv"

"strings"

)

func main() {

scanner := bufio.NewScanner(os.Stdin)

// 读取每组输入数据，直到文件结束

for scanner.Scan() {

// 读取牌的数量n和小明手中牌上的数字k

line := scanner.Text()

parts := strings.Fields(line)

n, \_ := strconv.Atoi(parts[0])

k, \_ := strconv.Atoi(parts[1])

// 读取n张牌上的数字

scanner.Scan()

line = scanner.Text()

cardParts := strings.Fields(line)

cards := make([]int, n)

for i, part := range cardParts {

cards[i], \_ = strconv.Atoi(part)

}

st := make(map[int]bool) // 用于存储前缀和模k的结果

st[0] = true // 初始化，插入0以处理从头开始的子数组

s := 0 // 当前前缀和

ok := 0 // 标记是否存在满足条件的子数组

// 遍历n张牌

for \_, x := range cards {

s = (s + x) % k // 更新前缀和并取模

if s < 0 {

s += k // 处理负数情况

}

// 检查前缀和模k的结果是否已经出现过

if st[s] {

ok = 1 // 如果出现过，表示存在满足条件的子数组

}

st[s] = true // 将当前前缀和模k的结果插入集合

}

fmt.Println(ok) // 输出结果，1表示存在，0表示不存在

}

}