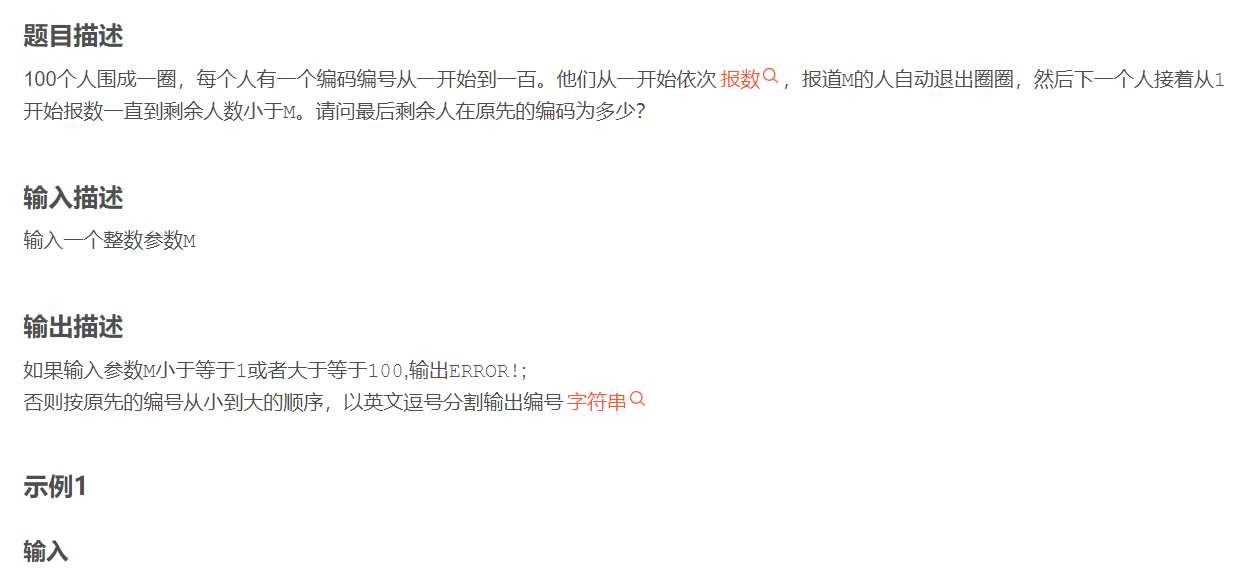
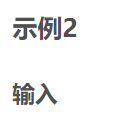
# **E卷-报数游戏[100分]（ Java | Python3 | C++ | C语言 | JsNode | Go ）**



3



58,91



4



34,45,97



import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner sc = new Scanner(

System.in); // 创建Scanner对象以读取用户输入

int k = sc.nextInt(); // 读取整数参数k

// 如果输入的参数k小于等于1或者大于等于100，输出"ERROR!"并退出程序

if (k <= 1 || k >= 100) {

System.out.println("ERROR!");

return;

}

Deque<Integer> q = new ArrayDeque<>(); // 创建一个双端队列

for (int i = 1; i <= 100; i++) {

q.add(i); // 初始化队列，包含1到100的编号

}

// 当队列中剩余的人数大于或等于k时，继续循环

while (q.size() >= k) {

// 依次从队列左侧取出前k-1个人并重新加入到队列右侧

for (int i = 0; i < k - 1; i++) {

int f = q.pollFirst(); // 从队列的左侧取出一个元素

q.addLast(f); // 将取出的元素加入到队列右侧

}

q.pollFirst(); // 删除队列左侧第k个人（即数到M的人）

}

List<Integer> result = new ArrayList<>(q); // 将剩余的元素转换为列表

Collections.sort(result); // 对剩余的编号按从小到大排序

System.out.println(result.toString().replaceAll("[\\[\\] ]",

"")); // 输出结果，以逗号分隔

}

}



from collections import deque # 导入collections模块中的deque数据结构（双端队列）

k = int(input()) # 从用户输入读取整数参数k

# 如果输入的参数k小于等于1或者大于等于100，输出"ERROR!"并退出程序

if k <= 1 or k >= 100:

print("ERROR!")

exit(0)

q = deque(range(1, 101)) # 初始化一个包含1到100的双端队列

# 当队列中剩余的人数大于或等于k时，继续循环

while len(q) >= k:

# 依次从队列左侧取出前k-1个人并重新加入到队列右侧

for i in range(k - 1):

f = q.popleft() # 从队列的左侧取出一个元素

q.append(f) # 将取出的元素加入到队列右侧

q.popleft() # 删除队列左侧第k个人（即数到M的人）

# 最后对剩余的编号按从小到大排序，并用逗号分隔后输出结果

print(','.join(map(str, sorted(q))))



#include <iostream>

#include <deque>

#include <vector>

#include <algorithm>

using namespace std;

int main() {

int k;

cin >> k; // 读取整数参数k

// 如果输入的参数k小于等于1或者大于等于100，输出"ERROR!"并退出程序

if (k <= 1 || k >= 100) {

cout << "ERROR!" << endl;

return 0;

}

deque<int> q; // 创建一个双端队列

for (int i = 1; i <= 100; i++) {

q.push\_back(i); // 初始化队列，包含1到100的编号

}

// 当队列中剩余的人数大于或等于k时，继续循环

while (q.size() >= k) {

// 依次从队列左侧取出前k-1个人并重新加入到队列右侧

for (int i = 0; i < k - 1; i++) {

int f = q.front(); // 从队列的左侧取出一个元素

q.pop\_front(); // 删除取出的元素

q.push\_back(f); // 将取出的元素加入到队列右侧

}

q.pop\_front(); // 删除队列左侧第k个人（即数到M的人）

}

vector<int> result(q.begin(), q.end()); // 将剩余的元素转换为向量

sort(result.begin(), result.end()); // 对剩余的编号按从小到大排序

for (size\_t i = 0; i < result.size(); i++) {

if (i > 0) cout << ","; // 输出逗号分隔符

cout << result[i]; // 输出结果

}

cout << endl;

return 0;

}



#include <stdio.h>

#include <stdlib.h>

int main() {

int k;

scanf("%d", &k); // 读取整数参数k

// 如果输入的参数k小于等于1或者大于等于100，输出"ERROR!"并退出程序

if (k <= 1 || k >= 100) {

printf("ERROR!\n");

return 0;

}

int q[100]; // 创建一个数组表示队列

for (int i = 0; i < 100; i++) {

q[i] = i + 1; // 初始化队列，包含1到100的编号

}

int size = 100; // 队列的初始大小

// 当队列中剩余的人数大于或等于k时，继续循环

while (size >= k) {

// 依次从队列左侧取出前k-1个人并重新加入到队列右侧

for (int i = 0; i < k - 1; i++) {

int f = q[0]; // 从队列的左侧取出一个元素

for (int j = 0; j < size - 1; j++) {

q[j] = q[j + 1]; // 将队列中的元素向左移动

}

q[size - 1] = f; // 将取出的元素加入到队列右侧

}

for (int j = 0; j < size - 1; j++) {

q[j] = q[j + 1]; // 删除队列左侧第k个人（即数到M的人）

}

size--; // 队列大小减1

}

// 对剩余的编号按从小到大排序

for (int i = 0; i < size - 1; i++) {

for (int j = 0; j < size - 1 - i; j++) {

if (q[j] > q[j + 1]) {

int temp = q[j];

q[j] = q[j + 1];

q[j + 1] = temp;

}

}

}

// 输出结果，以逗号分隔

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

if (i > 0) printf(",");

printf("%d", q[i]);

}

printf("\n");

return 0;

}



const readline = require("readline");

// 创建接口以读取用户输入

const rl = readline.createInterface({

input: process.stdin,

output: process.stdout,

});

// 读取整数参数k

rl.question("", (k) => {

k = parseInt(k);

// 如果输入的参数k小于等于1或者大于等于100，输出"ERROR!"并退出程序

if (k <= 1 || k >= 100) {

console.log("ERROR!");

rl.close();

return;

}

let q = []; // 创建一个数组表示队列

for (let i = 1; i <= 100; i++) {

q.push(i); // 初始化队列，包含1到100的编号

}

// 当队列中剩余的人数大于或等于k时，继续循环

while (q.length >= k) {

// 依次从队列左侧取出前k-1个人并重新加入到队列右侧

for (let i = 0; i < k - 1; i++) {

let f = q.shift(); // 从队列的左侧取出一个元素

q.push(f); // 将取出的元素加入到队列右侧

}

q.shift(); // 删除队列左侧第k个人（即数到M的人）

}

q.sort((a, b) => a - b); // 对剩余的编号按从小到大排序

console.log(q.join(",")); // 输出结果，以逗号分隔

rl.close();

});



package main

import (

"fmt"

"sort"

)

func main() {

var k int

fmt.Scan(&k)

// 如果输入的参数k小于等于1或者大于等于100，输出"ERROR!"并退出程序

if k <= 1 || k >= 100 {

fmt.Println("ERROR!")

return

}

q := make([]int, 100) // 创建一个数组表示队列

for i := 0; i < 100; i++ {

q[i] = i + 1 // 初始化队列，包含1到100的编号

}

size := 100 // 队列的初始大小

// 当队列中剩余的人数大于或等于k时，继续循环

for size >= k {

// 依次从队列左侧取出前k-1个人并重新加入到队列右侧

for i := 0; i < k-1; i++ {

f := q[0] // 从队列的左侧取出一个元素

q = q[1:] // 删除取出的元素

q = append(q, f) // 将取出的元素加入到队列右侧

}

q = q[1:] // 删除队列左侧第k个人（即数到M的人）

size-- // 队列大小减1

}

sort.Ints(q) // 对剩余的编号按从小到大排序

for i, v := range q {

if i > 0 {

fmt.Print(",") // 输出逗号分隔符

}

fmt.Print(v) // 输出结果

}

fmt.Println()

}