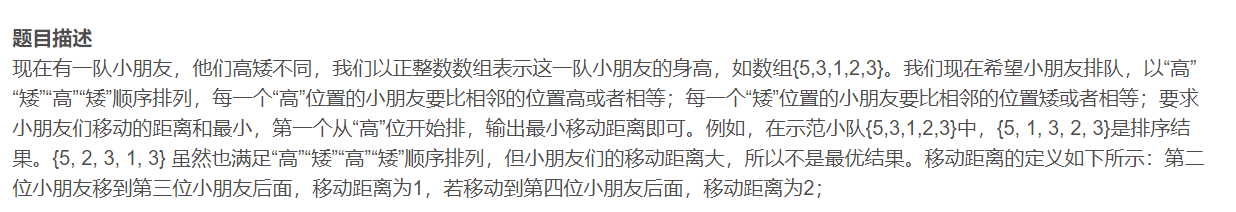
# **E卷-高矮个子排队[100分]（ Java | Python3 | C++ | C语言 | JsNode | Go）**











import java.util.\*;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

String s = scanner.nextLine();

// 标志位用于检查是否有非法字符

boolean flag = false;

// 检查字符串是否包含非法字符

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

char c = s.charAt(i);

if (c != ' ' && (c < '0' || c > '9')) {

flag = true;

break;

}

}

int now = 0;

List<Integer> v = new ArrayList<>();

// 解析字符串，将其转换为整数列表

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

char c = s.charAt(i);

if (c == ' ') {

v.add(now); // 遇到空格，将当前数字添加到列表

now = 0; // 重置累积数字

} else {

now = now \* 10 + (c - '0'); // 构建数字

}

}

v.add(now); // 添加最后一个数字

int n = v.size();

// 判断是否有非法字符或者数字个数是否超过 100

if (flag || n >= 100) {

System.out.println("[]");

return;

}

// 根据题目要求调整数字顺序

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

if (i <= n - 2) {

if (i % 2 == 0 && v.get(i) < v.get(i + 1)) {

Collections.swap(v, i, i + 1);

}

if (i % 2 == 1 && v.get(i) > v.get(i + 1)) {

Collections.swap(v, i, i + 1);

}

}

}

// 输出结果

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

if (i != 0) {

System.out.print(" ");

}

System.out.print(v.get(i));

}

}

}



# Python3 实现并带有详细中文注释

# 获取整行输入

s = input()

# 定义标志位用于检查非法字符

flag = 0

# 检查字符串是否包含非法字符

for char in s:

if char != " " and not char.isdigit():

flag = 1

# 当前数字累积器

now = 0

# 用于存放读取到的数字的列表

v = []

# 遍历字符串，将其解析为数字列表

for char in s:

if char == " ":

v.append(now) # 遇到空格时，将当前数字保存到列表

now = 0 # 重置累积器

else:

now = now \* 10 + int(char) # 累积构建数字

# 添加最后一个数字

v.append(now)

# 检查数字数量并输出合法的数字列表

n = len(v)

if flag or n >= 100:

print("[]")

else:

# 根据要求调整数字顺序

for i in range(n):

if i <= n - 2:

if i % 2 == 0 and v[i] < v[i + 1]: # 偶数索引：如果当前数字小于下一个数字，则交换

v[i], v[i + 1] = v[i + 1], v[i]

if i % 2 == 1 and v[i] > v[i + 1]: # 奇数索引：如果当前数字大于下一个数字，则交换

v[i], v[i + 1] = v[i + 1], v[i]

# 输出结果

print(" ".join(map(str, v)))



#include <stdio.h>

#include <string.h>

#include <ctype.h>

int main() {

// 定义一个字符数组来存储输入字符串

char s[1000];

// 从标准输入读取一行字符并存入数组s中

fgets(s, 1000, stdin);

// 标志变量，用于判断输入字符串是否只包含数字和空格

int flag = 0;

// 获取输入字符串的长度并存储在len变量中

size\_t len = strlen(s);

// 如果字符串结尾有换行符，则去掉它

if (len > 0 && s[len - 1] == '\n') {

s[len - 1] = '\0';

}

// 遍历字符串，确保所有字符都是数字或空格

for (int i = 0; i < strlen(s); i++) {

if (s[i] != ' ' && !isdigit(s[i])) {

flag = 1; // 如果发现非数字和非空格字符，将标志置为1

break;

}

}

// 用于临时存储当前数字

int now = 0;

// 定义一个数组来存储分割后的整数

int v[100];

// 数组索引

int idx = 0;

// 将输入字符串分割成整数并存储到数组v中

for (int i = 0; i < strlen(s); i++) {

if (s[i] == ' ') {

if (idx < 100) {

v[idx++] = now; // 将当前数字存入数组

}

now = 0; // 重置当前数字

} else {

// 更新当前数字

now = now \* 10 + (s[i] - '0');

}

}

// 将最后一个数字存入数组

if (idx < 100) {

v[idx++] = now;

}

// 获取数组中的数字个数

int n = idx;

// 如果发现非法字符或数字个数超过100，输出空数组并退出程序

if (flag || n >= 100) {

printf("[]\n");

return 0;

}

// 对 vector 中的数字按照题目要求进行排序操作

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

if (i <= n - 2) {// 检查当前位置和下一个位置的数字是否需要交换

if (i % 2 == 0 && v[i] < v[i + 1]) {// 当前索引是偶数且当前数字小于下一个数字时，交换

int tmp = v[i];

v[i] = v[i + 1];

v[i + 1] = tmp;

}

if (i % 2 == 1 && v[i] > v[i + 1]) {// 当前索引是奇数且当前数字大于下一个数字时，交换

int tmp = v[i];

v[i] = v[i + 1];

v[i + 1] = tmp;

}

}

}

// 输出排序后的结果

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

if (i != 0) {

printf(" ");

}

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

}

return 0;

}



#include <stdio.h>

#include <string.h>

#include <ctype.h>

int main() {

// 定义一个字符数组来存储输入字符串

char s[1000];

// 从标准输入读取一行字符并存入数组s中

fgets(s, 1000, stdin);

// 标志变量，用于判断输入字符串是否只包含数字和空格

int flag = 0;

// 获取输入字符串的长度并存储在len变量中

size\_t len = strlen(s);

// 如果字符串结尾有换行符，则去掉它

if (len > 0 && s[len - 1] == '\n') {

s[len - 1] = '\0';

}

// 遍历字符串，确保所有字符都是数字或空格

for (int i = 0; i < strlen(s); i++) {

if (s[i] != ' ' && !isdigit(s[i])) {

flag = 1; // 如果发现非数字和非空格字符，将标志置为1

break;

}

}

// 用于临时存储当前数字

int now = 0;

// 定义一个数组来存储分割后的整数

int v[100];

// 数组索引

int idx = 0;

// 将输入字符串分割成整数并存储到数组v中

for (int i = 0; i < strlen(s); i++) {

if (s[i] == ' ') {

if (idx < 100) {

v[idx++] = now; // 将当前数字存入数组

}

now = 0; // 重置当前数字

} else {

// 更新当前数字

now = now \* 10 + (s[i] - '0');

}

}

// 将最后一个数字存入数组

if (idx < 100) {

v[idx++] = now;

}

// 获取数组中的数字个数

int n = idx;

// 如果发现非法字符或数字个数超过100，输出空数组并退出程序

if (flag || n >= 100) {

printf("[]\n");

return 0;

}

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

if (i <= n - 2) {

if (i % 2 == 0 && v[i] < v[i + 1]) {

int tmp = v[i];

v[i] = v[i + 1];

v[i + 1] = tmp;

}

if (i % 2 == 1 && v[i] > v[i + 1]) {

int tmp = v[i];

v[i] = v[i + 1];

v[i + 1] = tmp;

}

}

}

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

if (i != 0) {

printf(" ");

}

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

}

return 0;

}



const readline = require("readline");

// 创建 readline 接口以便从标准输入读取数据

const rl = readline.createInterface({

input: process.stdin,

output: process.stdout,

});

// 读取输入并处理

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

let flag = false;

// 检查字符串是否包含非法字符

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

if (s[i] !== " " && (s[i] < "0" || s[i] > "9")) {

flag = true;

break;

}

}

let now = 0;

const v = [];

// 将输入字符串解析为整数列表

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

if (s[i] === " ") {

v.push(now); // 遇到空格时，将当前数字存入数组

now = 0; // 重置当前数字累积器

} else {

now = now \* 10 + (s[i] - "0"); // 构建当前数字

}

}

v.push(now); // 添加最后一个数字

const n = v.length;

if (flag || n >= 100) {

console.log("[]"); // 非法字符或数字太多时打印空数组

} else {

// 按规定顺序调整数字

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

if (i <= n - 2) {

if (i % 2 === 0 && v[i] < v[i + 1]) {

[v[i], v[i + 1]] = [v[i + 1], v[i]]; // 交换偶数索引和下一个数字

}

if (i % 2 === 1 && v[i] > v[i + 1]) {

[v[i], v[i + 1]] = [v[i + 1], v[i]]; // 交换奇数索引和下一个数字

}

}

}

// 输出结果

console.log(v.join(" "));

}

rl.close(); // 关闭 readline 接口

});



package main

import (

"bufio"

"fmt"

"os"

"strings"

)

func main() {

reader := bufio.NewReader(os.Stdin)

s, \_ := reader.ReadString('\n') // 读取一行输入

s = strings.TrimSpace(s) // 去掉首尾空格

flag := false

for i := 0; i < len(s); i++ {

// 检查字符串中是否包含非法字符

if s[i] != ' ' && (s[i] < '0' || s[i] > '9') {

flag = true

break

}

}

var v []int

now := 0

for i := 0; i < len(s); i++ {

if s[i] == ' ' {

v = append(v, now) // 将当前数字添加到数组

now = 0 // 重置当前数字累积器

} else {

now = now\*10 + int(s[i]-'0') // 构建当前数字

}

}

v = append(v, now) // 添加最后一个数字

n := len(v)

if flag || n >= 100 {

fmt.Println("[]") // 非法字符或数字太多时打印空数组

return

}

// 按规定顺序调整数字

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

if i <= n-2 {

if i%2 == 0 && v[i] < v[i+1] {

v[i], v[i+1] = v[i+1], v[i] // 交换偶数索引和下一个数字

}

if i%2 == 1 && v[i] > v[i+1] {

v[i], v[i+1] = v[i+1], v[i] // 交换奇数索引和下一个数字

}

}

}

// 打印结果

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

if i != 0 {

fmt.Print(" ")

}

fmt.Print(v[i])

}

}