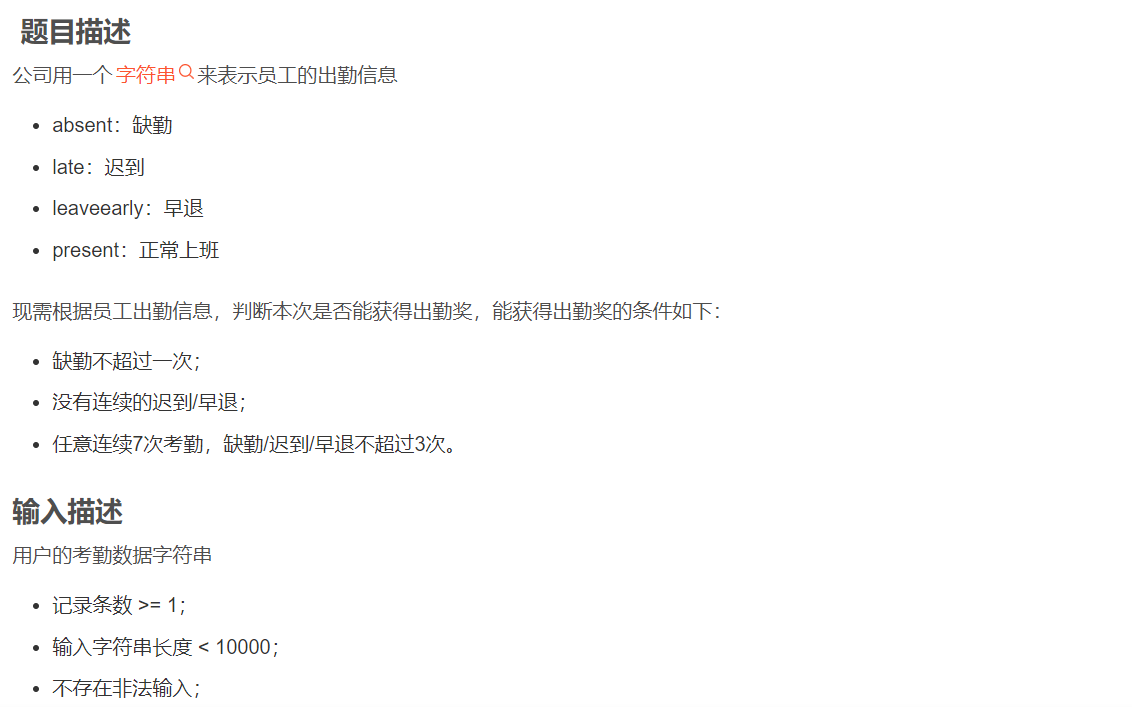
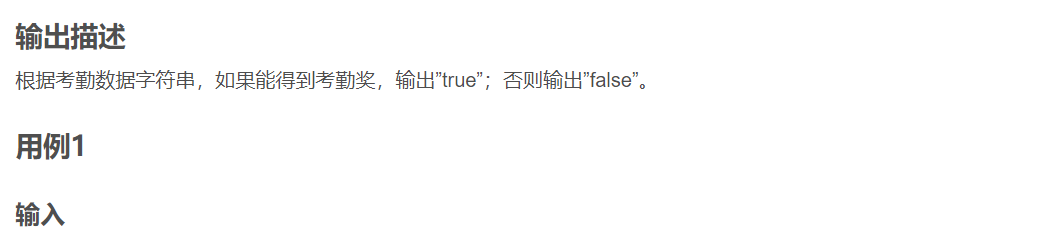
# **E卷-考勤信息[100分]（ Java | Python3 | C++ | C语言 | JsNode | Go）**





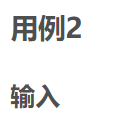
2

present

present present



true true



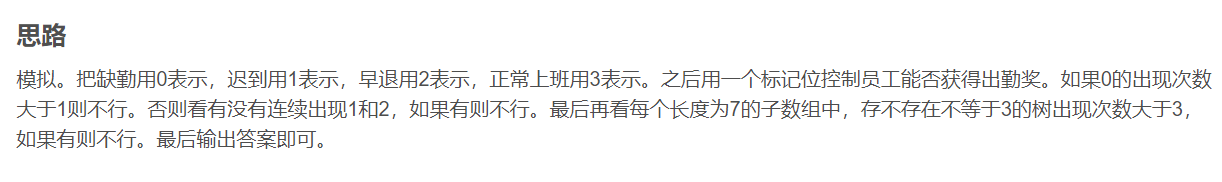
2

present

present absent present present leaveearly present absent



true false





import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

// 读取记录条数

int n = scanner.nextInt();

scanner.nextLine();

// 处理每条输入记录

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

// 读取单条记录

String s = scanner.nextLine();

// 输出判断结果

System.out.print(work(s) + " ");

}

}

public static int get(String s) {

// 根据字符串第二个字符判断状态

if (s.charAt(1) == 'a') return 1; // 缺勤

if (s.charAt(1) == 'e') return 2; // 早退

if (s.charAt(1) == 'r') return 3; // 正常上班

return 0; // 迟到

}

public static String work(String record) {

// 将考勤记录分割为单个考勤状态

String[] s = record.split(" ");

int[] a = new int[s.length +

10]; // 初始化标识数组，增加额外长度防止越界

boolean f = true; // 标识是否符合出勤奖条件

int tot = 0; // 缺勤计数器

// 转换考勤状态并计数

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

int x = get(s[i]);

if (x == 0) tot++;

a[i] = x;

}

// 条件1：检查缺勤次数是否超过一次

if (tot > 1) f = false;

// 条件2：检查是否有连续的迟到或早退

for (int i = 1; i < s.length; i++) {

if (a[i] <= 2 && a[i - 1] <= 2 && a[i] >= 1 && a[i - 1] >= 1) f = false;

}

// 条件3：检查任意连续7次考勤，特殊事件不超过3次

for (int i = 0; i <= s.length - 7; i++) {

int c = 0;

for (int j = i; j < i + 7; j++) {

if (a[j] != 3) c++;

}

if (c > 3) f = false;

}

// 返回结果, 使用小写

return f ? "true" : "false";

}

}



# 读取输入的记录条数

n = int(input())

def get(s):

"""

根据给定的考勤字符串，返回对应的标识数字:

"a" -> absent -> 缺勤 -> 返回 1

"e" -> leaveearly -> 早退 -> 返回 2

"r" -> present -> 正常上班 -> 返回 0

其他 -> (这里假设为 "late" -> 迟到) -> 返回 3

"""

res = 0

if s[1] == "a":

res = 1

elif s[1] == "e":

res = 2

elif s[1] == "r":

res = 3

return res

def work():

# 读取当前记录并分割成单个考勤状态的列表

s = input().split()

# 初始化状态数组

a = []

# 标识是否符合出勤奖的条件

f = True

# 缺勤计数器

tot = 0

# 转换每个考勤状态到对应的标识数字并处理逻辑

for i in s:

x = get(i)

if x == 0:

tot += 1

a.append(x)

# 为了防止越界，在数组后添加十个正常上班的标识

a += [3] \* 10

# 条件1：检查缺勤次数是否超过一次

if tot > 1:

f = False

# 条件2：检查是否有连续的迟到或早退

for i in range(1, len(a)):

if 1 <= a[i] <= 2 and 1 <= a[i - 1] <= 2:

f = False

# 条件3：检查任意连续7次考勤，缺勤/迟到/早退不超过3次

for i in range(len(a)):

c = 0

if i + 7 > len(a):

break

for j in range(i, i + 7):

if a[j] != 3:

c += 1

if c > 3:

f = False

# 打印结果，转换为小写字符串

print(str(f).lower(), end=" ")

# 处理每一条考勤记录

for \_ in range(n):

work()



#include <iostream>

#include <vector>

#include <string>

#include <sstream>

using namespace std;

int get(const string &s) {

// 根据字符串第二个字符判断状态

if (s[1] == 'a') return 1; // 缺勤

if (s[1] == 'e') return 2; // 早退

if (s[1] == 'r') return 3; // 正常上班

return 0; // 迟到

}

string work(const string &record) {

// 使用字符串流分割字符串为单个考勤状态

istringstream iss(record);

vector<int> a;

string s;

bool f = true; // 标识是否符合出勤奖条件

int tot = 0; // 缺勤计数器

// 转换考勤状态并计数

while (iss >> s) {

int x = get(s);

if (x == 0) tot++;

a.push\_back(x);

}

for (int i = 0; i < 10; i++) a.push\_back(3); // 添加额外元素，防止越界

// 条件1：检查缺勤次数是否超过一次

if (tot > 1) f = false;

// 条件2：检查是否有连续的迟到或早退

for (int i = 1; i < a.size(); i++) {

if (a[i] <= 2 && a[i - 1] <= 2 && a[i] >= 1 && a[i - 1] >= 1) f = false;

}

// 条件3：检查任意连续7次考勤，特殊事件不超过3次

for (int i = 0; i <= a.size() - 7; i++) {

int c = 0;

for (int j = i; j < i + 7; j++) {

if (a[j] != 3) c++;

}

if (c > 3) f = false;

}

// 返回结果

return f ? "true" : "false";

}

int main() {

int n;

cin >> n;

cin.ignore(); // 忽略换行符

// 处理每条输入记录

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

string record;

getline(cin, record);

// 输出判断结果

cout << work(record) << " ";

}

return 0;

}



#include <stdio.h>

#include <string.h>

#include <stdlib.h>

int get(const char\* s) {

// 根据字符串第二个字符判断状态

if (s[1] == 'a') return 1; // 缺勤

if (s[1] == 'e') return 2; // 早退

if (s[1] == 'r') return 3; // 正常上班

return 0; // 迟到

}

char\* work(const char\* record) {

static char result[6]; // 存储结果的静态数组

strcpy(result, "true"); // 结果初始为"true"

int a[10010] = {0}; // 初始化标识数组

char s[10000];

int f = 1; // 标识是否符合出勤奖条件

int tot = 0; // 缺勤计数器

int idx = 0;

// 使用strtok拆分字符串处理考勤状态

strcpy(s, record);

char\* token = strtok(s, " ");

while (token != NULL) {

int x = get(token);

if (x == 0) tot++;

a[idx++] = x;

token = strtok(NULL, " ");

}

// 条件1：检查缺勤次数是否超过一次

if (tot > 1) f = 0;

// 条件2：检查是否有连续的迟到或早退

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

if (a[i] <= 2 && a[i - 1] <= 2 && a[i] >= 1 && a[i - 1] >= 1) f = 0;

}

// 条件3：检查任意连续7次考勤，特殊事件不超过3次

for (int i = 0; i <= idx - 7; i++) {

int c = 0;

for (int j = i; j < i + 7; j++) {

if (a[j] != 3) c++;

}

if (c > 3) f = 0;

}

// 返回结果

return f ? "true" : "false";

}

int main() {

int n;

scanf("%d", &n);

getchar(); // 忽略换行符

// 处理每条输入记录

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

char record[10000];

fgets(record, 10000, stdin);

// 去掉字符串末尾的换行符

record[strcspn(record, "\n")] = '\0';

// 输出判断结果

printf("%s ", work(record));

}

return 0;

}



const readline = require("readline");

const rl = readline.createInterface({

input: process.stdin,

output: process.stdout,

});

let inputLines = [];

let n;

let currentLine = 0;

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

inputLines.push(input);

});

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

n = parseInt(inputLines[0]);

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

process.stdout.write(work(inputLines[i]).toLowerCase() + " ");

}

});

function get(s) {

// 根据字符串第二个字符判断状态

if (s[1] === "a") return 1; // 缺勤

if (s[1] === "e") return 2; // 早退

if (s[1] === "r") return 3; // 正常上班

return 0; // 迟到

}

function work(record) {

let s = record.split(" ");

let a = new Array(s.length + 10).fill(3); // 初始化标识数组

let f = true; // 标识是否符合出勤奖条件

let tot = 0; // 缺勤计数器

// 转换考勤状态并计数

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

let x = get(s[i]);

if (x === 0) tot++;

a[i] = x;

}

// 条件1：检查缺勤次数是否超过一次

if (tot > 1) f = false;

// 条件2：检查是否有连续的迟到或早退

for (let i = 1; i < s.length; i++) {

if (a[i] <= 2 && a[i - 1] <= 2 && a[i] >= 1 && a[i - 1] >= 1) f = false;

}

// 条件3：检查任意连续7次考勤，特殊事件不超过3次

for (let i = 0; i <= s.length - 7; i++) {

let c = 0;

for (let j = i; j < i + 7; j++) {

if (a[j] !== 3) c++;

}

if (c > 3) f = false;

}

// 返回结果

return f ? "true" : "false";

}



package main

import (

"bufio"

"fmt"

"os"

"strconv"

"strings"

)

// get 根据字符串的第二个字符判断状态

func get(s string) int {

switch s[1] {

case 'a':

return 1 // 缺勤

case 'e':

return 2 // 早退

case 'r':

return 3 // 正常上班

default:

return 0 // 迟到

}

}

// work 处理单条记录并输出是否符合条件

func work(record string) string {

s := strings.Fields(record) // 分割考勤记录

a := make([]int, len(s)) // 初始化标识数组，增加额外长度防止越界

f := true // 标识是否符合出勤奖条件

tot := 0 // 缺勤计数器

// 转换考勤状态并计数

for i, str := range s {

x := get(str)

if x == 0 {

tot++

}

a[i] = x

}

for i := 0;i < 10;i++{

a = append(a,3)

}

// 条件1：检查缺勤次数是否超过一次

if tot > 1 {

f = false

}

// 条件2：检查是否有连续的迟到或早退

for i := 1; i < len(a); i++ {

if a[i] <= 2 && a[i-1] <= 2 && a[i] >= 1 && a[i - 1] >= 1 {

f = false

break

}

}

// 条件3：检查任意连续7次考勤，特殊事件不超过3次

for i := 0; i <= len(a)-7; i++ {

c := 0

for j := i; j < i+7; j++ {

if a[j] != 3 {

c++

}

}

if c > 3 {

f = false

break

}

}

// 返回结果

if f {

return "true"

}

return "false"

}

func main() {

scanner := bufio.NewScanner(os.Stdin)

scanner.Scan()

n, \_ := strconv.Atoi(scanner.Text()) // 读取记录条数

results := make([]string, n)

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

scanner.Scan()

record := scanner.Text() // 读取单条记录

results[i] = work(record) // 处理记录并取得结果

}

// 输出所有结果

fmt.Println(strings.Join(results, " "))

}