Разбор задач простого тура.

Задача B. Соревнование картингистов

В этой задаче требовалось просто *аккуратно* реализовать то, что написано в условии: найти сумму чисел, найти минимум среди этих сумм и имя, ему соответствующее.

Одна из проблем, которая могла возникнуть – считывание данных. Рассмотрим следующий код:

Program.pas	Input.txt

var
a, b: longint; s: string;
begin
  assign(input, 'input.txt'); assign(output, 'output.txt');
  reset(input); rewrite(output);
 	
  read(a, b); // читает 1, 2 и текующая позиция в файле
  // указывает на символ перевода строки
 	
  readln(s); // читает до конца строки, так как мы уже к конце
  // строки, то в s после считывания находится пустая
  // строка.
	1 2
	abracadabra

Чтобы всё работало, нужно вместо read(a, b); написать readln(a, b); который после чтения чисел сразу переходит на следующую строку.

Задача C. Размещение груза

Нужно было перебрать все возможные положения груза (их всего *(R-1)*(C-1)*). Для каждого положения проверяем, не оказались ли под грузом стены и считаем сколько под ним коробок. Увеличиваем ответ для соответствующего числа коробок.

Задача F. Тимбилдинг

Пусть M-количество девушек, N – количество парней. Тогда если мы никого не отправляем в командировку, то мы сможем организовать не более T = Min(M div 2, N) команд. Очевидно, что чтобы максимизировать количество команд надо отправлять в командировку людей, не вошедших в эти T команд и, возможно, ещё кого-то. Количество таких людей X = (N+M – 3*T). Если X >= K, то ответ, очевидно T.

Если же X < K, то мы сможем выкинуть K людей таким образом, чтобы из оставшихся можно было сформировтаь полноценные *(N+M-K) div 3* команды.

Если обобщить эти два случая, то искомое число команд равно *min(min(m div 2, n), (n + m - k) div 3)*

Задача A. Хоббит или Туда и обратно

Очевидное решение:

в двумерном массиве нарисовать оба маршрута и найти точки пересечений. Очевидно, что такое решение занимает O(n^2) памяти, что в худшем случае порядка 10 ГигаБайт. Это слишком много.

Рассмотрим оптимальное решение:
Нам дана строка, половина которой - путь туда, а половина - путь обратно. Поставим на карту сразу двух хоббитов. Оба хоббита будут идти от левой верхней клетке к правой нижней. При этом первый будет идти по маршруту, по которому шёл хоббит в прямом направлении в задаче, а второй будет идти в обратную сторону по обратному маршруту хоббита из задачи. Циклом переберём строку, при этом на каждой итерации будем рассматривать i-ый символ с начала входной строки для первого хоббита и i-ый символ с конца пути для второго хоббита. С учётом этих символов изменяем координаты обоих хоббитов.
Если в какой-то момент времени их координаты совпадают, значит нашлась точка маршрутов.

Осталось только вывести это число, не забыв про то, что есть точки, которые учитывать по условию не надо.

P.S. Это работает, потому что до каждой возможной точки пересечения мы добираемся за конкретное число шагов, т.е если до какой-то точки мы можем дойти за 5 шагов, то мы не сможем добраться до неё другим путём за 4, 6 или другое число шагов, только ровно за 5.

Задача D. Разбалловка

Рассмотрим всех участников.
Очевидно, что в паре двух участника таких, что у первого есть тест, который не прошёл у второго, второй не может быть точно победителем олимпиады, так как первый мог набрать больше баллов на этом тесте.

Так для каждого участника проверим всех его соперников. Если для каждого соперника не нашлось такого, что выполнится условие, описанное выше (то есть не нашлось участника, который мог набрать больше баллов, чем данный), то данный участник точно может быть победителем.

Задача E. Звёздные имена.

Будем решать задачу при помощи динамического программирования.

Состояние:
dp[первая буква][длина][последняя буква] = количество способов построить имя такой длины начинающееся с первой буквы и заканчивающееся на последнюю букву по модулю 10^9+7

Переход:

Перебираем букву C, тогда

if (b!=c) then dp[a][i + 1][c] += dp[a][i][b]

Начальное состояние:

dp[c][1][c] = 1; для всех C от 1 до N, Где N – количество букв.

Ответ:

Сумма dp[c][L-1][c] для всех C от 1 до N.

При вычислениях не забывать про модуль 10^9+7.