ВКОШП-2012

Разбор задач

Санкт-Петербург, 2012

Задача А
Рекламный Щит

- Автор задачи – Георгий Корнеев
- Условие – Андрей Комаров
- Подготовка тестов – Андрей Комаров
- Разбор – Андрей Комаров

Постановка задачи

- Дано несколько таблиц одинакового размера, состоящих из двух типов символов (включено, выключено).
- Требуется разбить их на группы так, чтобы любую из заданных таблиц можно было получить, включая или выключая целиком группы.
- Нужно минимизировать число групп.

Как решать?

- Две клетки *могут быть* в одной группе, если нет таблицы, в которой они различны.
- Для каждой клетки построим список таблиц, в которых она включена.

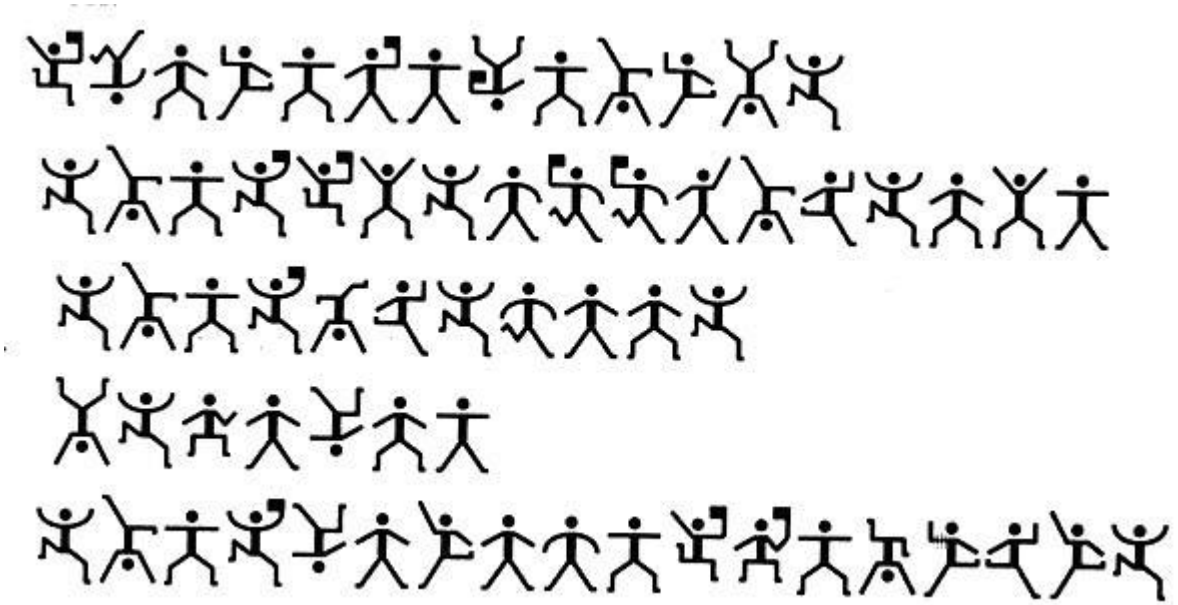
Как решать? (продолжение)

- Если у двух клеток одинаковые списки, то эти клетки могут быть в одной группе.
  - Если бы была таблица, в которой они различны, то в одном из списков она бы была, а в другом — нет.
- Если списки различны, то клетки точно в разных группах.
  - В той таблице, на которой достигается различие списков, одна из клеток включена, а вторая – выключена.

Как решать? (продолжение)

- Если списки одинаковые, то таблицы в одной и той же группе.
- Минимальное число групп — число различных списков.
- Посчитать количество различных списков можно, посчитав хеш от каждого.
- O(nmk).

Задача B
Хаотическая перестановка



- Автор задачи – Игорь Пышкин
- Условие – Павел Кунявский
- Подготовка тестов – Павел Кунявский
- Разбор – Павел Кунявский

Постановка задачи

- Дана перестановка.
- Разрешено сделать не более чем n обменов соседних элементов.
- Надо чтобы никакие три подряд идущих элемента не были упорядоченными.

Как решать? (Конструкция)

- Будем добавлять элементы слева на право по одному, поддерживая префикс хаотическим.
- Добавим очередной элемент.
- Если с последней тройкой все хорошо, делать ничего не надо.

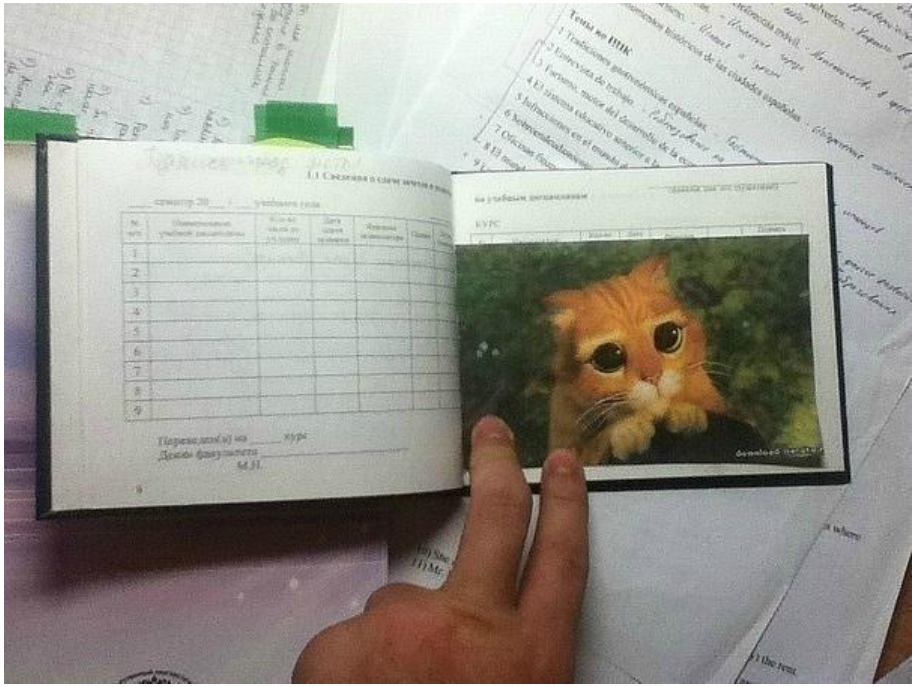
Как решать? (продолжение)

- Если последние три элемента упорядочены, то надо поменять местами последние два.
- При этом не будут упорядочены ни последняя тройка, ни предпоследяя.

Еще одно решение

- Можно решать задачу за не более, чем $\frac{n}{2}$ обменов.
- Будем добавлять по два элемента сразу.
- Разбором случаев, можно доказать, что достаточно поменять или последнюю или предпоследнюю пару.

Задача С
Экзамен



- Автор задачи – Глеб Евстропов
- Условие – Михаил Пядеркин
- Подготовка тестов – Михаил Пядеркин
- Разбор – Михаил Пядеркин, Павел Кунявский

Постановка задачи

- Дан массив из натуральных чисел. Мы можем вычесть один на префиксе, вычесть один на суффиксе и прибавить 1 к элементу.
- Необходимо за минимальное число действий третьего типа превратить массив в нулевой

Переформулировка

- Найдем условие, при котором массив можно только операциями только первого и второго типа можно превратить в нулевой
- Массив должен представляться в виде суммы неубывающего и невозрастающего массивов

Решение простой задачи

- Рассмотрим убывающую часть. Пусть $a_i - a_{i+1} > 0$. Тогда убывающая часть должна уменьшиться хотя бы на $a_i - a_{i+1}$.
- Значение первого элемента убывающей последовательности не больше, чем a_1, значит, сумма «убываний» не должна превышать a_1. Обозначим эту сумму «убываний» переменной m.
- Покажем, что данное условие достаточно.

Решение простой задачи

- Сделаем первым элементом убывающей последовательности a_1, а возрастающей - ноль.
- Будем строить последовательности добавляя по одному элементу.
- Если a_i меньше чем a_{i+1}, увеличим возрастающую, иначе уменьшим убывающую.
- Условие на разности гарантирует, что убывающая не станет отрицательной.

Решение общей задачи

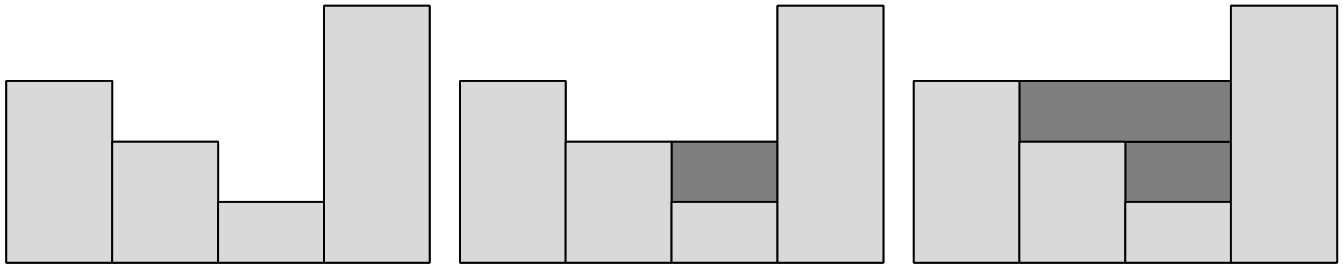
- Увеличениями одного элемента приведем последовательность к описанному выше виду.
- Чтобы уменьшить сумму «убываний» на один, надо увеличить на один блок подряд идущих убывающих элементов.
- Выгодно увеличивать на один самые короткие блоки.

Решение общей задачи

- Посчитаем количество блоков каждой длины, которые придется увеличивать.
- Надо увеличить $(m - a_1)$ блоков.

Реализация решения

- Будем хранить стек убывающих элементов.
- Добавляем очередной элемент. Пока он больше, чем последний элемент в стеке, будем вынимать последний элемент из стека, и пересчитывать количество отрезков каждой длины.



Задача D
Лесопосадки

- Автор задачи – Демид Кучеренко
- Условие – Павел Кротков
- Подготовка тестов – Павел Кротков
- Разбор – Павел Кротков

Постановка задачи

- Дан закон, по которому изменяется высота деревьев в лесу.
- Дерево за год вырастает на один метр тогда и только тогда, когда хотя бы у одного из его соседей высота больше ровно на один метр.
- По начальным высотам деревьев определить момент завершения процесса и высоту всех деревьев в этот момент.

Как решать?

- Начнем с первой идеи, которая приходит в голову.
- Будем каждый год проверять все деревья и увеличивать высоту тех, которые должны подрасти в этот год.

Оценка времени работы

- Время работы такой программы - $O(nmT)$, где T — количество лет, которое будет продолжаться процесс.
- Время работы зависит от оценки числа T.

Оценка числа T

- Каждое дерево подрастает не больше, чем k раз, где k — максимальная высота дерева в начальный момент.
- Всего деревьев $n \times m$.
- Итого: $T = O(nmk)$.
- Построим тест, на котором эта оценка достигается.

Оценка числа T

- Нужный тест — тот, в котором на каждой итерации подрастает одно дерево.
- Этого можно добиться, расставив деревья, которые будут расти, например, по спирали.

Оценка числа T

12	**10**	**8**	**6**	**4**	**2**	**1**
15	15	15	15	15	15	**1**
1	**1**	**1**	**3**	**5**	15	**1**
1	15	15	15	**7**	15	**1**
1	15	**13**	**11**	**9**	15	**1**
1	15	15	15	15	15	**1**
1	**1**	**1**	**1**	**1**	**1**	**1**

Оценка времени работы

- $O(nmk)$ лет (итераций).
- Каждая итерация за $O(nm)$.
- Итого $O(n^2m^2k)$.
- Time limit exceeded, test 9.

Оптимизация моделирования

- На каждой итерации проверяем только «интересные» деревья.
- Дерево считается интересным, если оно или хотя бы один его сосед подросли за предыдущий год.
- На первом шаге все деревья интересные.

Решение

- Проверяем все интересные деревья и запоминаем, какие из них подросли.
- Увеличиваем рост всех подросших деревьев, попутно добавляя в новое множество интересных каждое подросшее дерево и всех его соседей.
- Останавливаемся, когда очередное множество интересных деревьев пусто.

Полезный факт

- Поскольку координаты деревьев невелики $(x, y < 100)$, можно каждому дереву присвоить уникальный номер $xm + y$.
- Работать с множествами из чисел, а не из пар, существенно удобнее.

Задача E
Академия Джедаев

- Автор задачи – Федор Царев
- Условие – Антон Ковшаров
- Подготовка тестов – Антон Ковшаров
- Разбор – Антон Ковшаров

Постановка задачи

- В Академии обучают различным навыкам.
- Некоторые навыки можно изучать только после изучения каких-то других.
- Навыки изучают в двух корпусах.
- Изучение каждого навыка b минут.
- Переход из одного корпуса в другой a минут.
- Можно выбрать в какой корпус идти сначала.
- Необходимо вывести минимальное время на изучение всех навыков.

Как решать? (Жадность)

- Переберем с какого корпуса будем начинать.
- Если мы можем сейчас изучить навык – изучаем.
- Если сейчас нет навыков доступных для изучения в текущем корпусе, но мы еще не изучили все навыки, переходим в другой корпус.

Реализация

- Зависимости между навыками можно хранить в виде ориентированного графа, где ребро uv обозначает, что навык u необходим для навыка v.
- Навык доступен для изучения, если в него не входит никаких ребер.
- После изучения очередного навыка удаляем все ребра исходящие из него.

Реализация (продолжение)

- Храним две очереди для навыков доступных на данный момент в первом корпусе и во втором.
- Когда мы находимся в очередном корпусе и в соответствующей очереди есть элементы, достаем очередной и изучаем его, добавляя в очереди те навыки, входящая степень которых стала равна 0.

Задача F
Спасти котенка

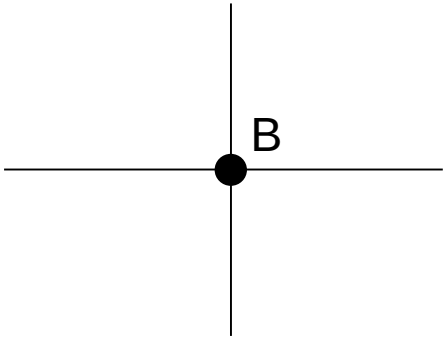
- Автор задачи – Демид Кучеренко
- Условие – Демид Кучеренко
- Подготовка тестов – Демид Кучеренко
- Разбор – Демид Кучеренко

Постановка задачи

- Дана таблица N×M.
- Нужно найти количество путей из клетки A в клетку C, проходящих через B.
- В одну клетку нельзя заходить дважды.

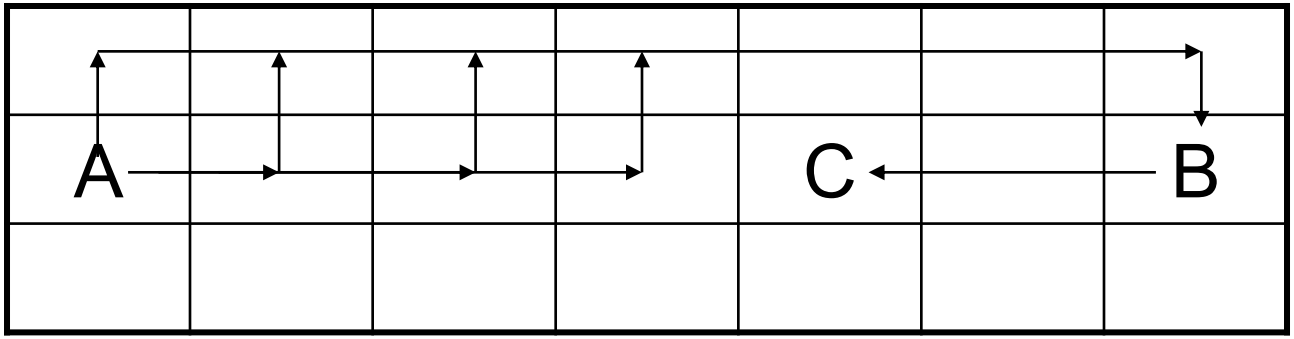
Как решать?

- Рассмотрим несколько вариантов расположения точек A и C.
- Для этого посмотрим в каких четвертях относительно точки B лежат точки A и C.



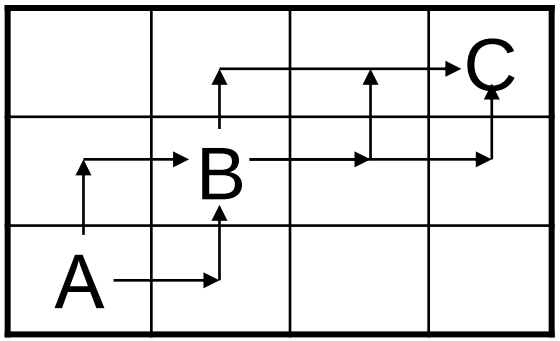
Как решать? (1 случай)

- Если все три точки лежат на одной прямой:
- Если точка B лежит между A и C, то $ans = 1$.
- В другом случае $ans = 2 \cdot dist(A, C)$, если мы находимся не в крайней строке/столбце. Иначе $ans = dist(A, C)$.



Как решать? (2 случай)

- Если точки A и C находятся в двух противоположных четвертях.
- Тогда ответом будет являться произведение количества простых путей от A до B и от B до C.

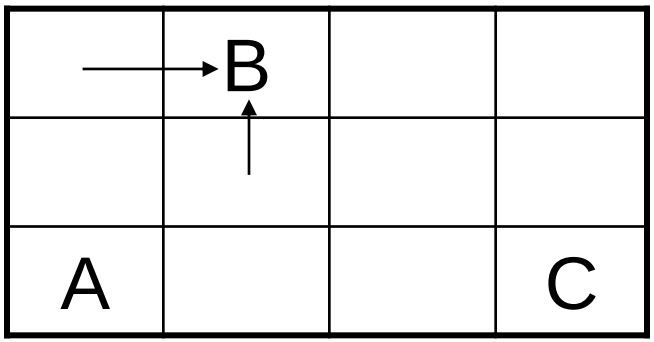


Количество простых путей

- Найдем количество простых путей из точки A в точку B.
- Пусть для того, чтобы дойти из точки A в точку B нам нужно сделать x шагов вправо и y шагов вверх. Тогда всего нам нужно сделать $x + y$ шагов.
- Среди этих шагов нам нужно выбрать x шагов вправо. Количество способов выбрать эти x шагов и будет искомым числом путей.
- Это будет число сочетаний из $x + y$ по x.

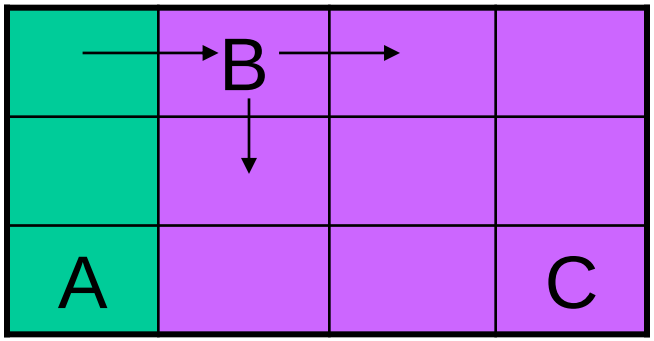
Как решать? (3 случай)

- Если точки A и C находятся в двух соседних четвертях.
- Рассмотри два способа как мы могли попасть в клетку B (снизу и слева для данного рисунка).
- Ответом будет сумма ответов этих случаев.



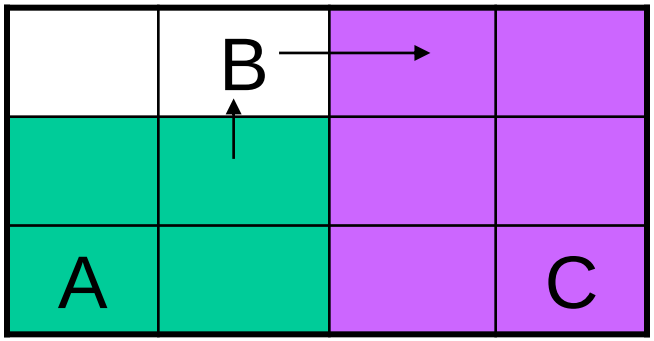
Как решать? (3 случай)

- Пусть мы пришли слева.
- Тогда ответ будет количество способов прийти из A в левого соседа B, умноженное на количество способов дойти от B до C.



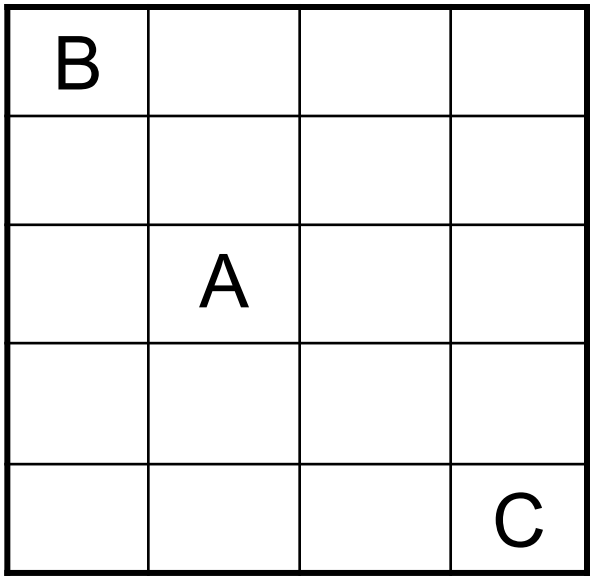
Как решать? (3 случай)

- Пусть мы пришли снизу.
- Тогда ответ будет количество способов прийти из A в нижнего соседа B, умноженное на количество способов дойти от правого соседа B (только туда мы могли пойти из B) до C



Как решать? (4 случай)

- Если точки A и C находятся в одной четверти
- В одну клетку по-прежнему нельзя заходить два раза
- Посчитаем количество пар непересекающихся путей из B до A и C одновременно
- Приведем координаты к такому виду, что точка B будет в точке $(1,1)$, а координаты остальных точек будут положительны



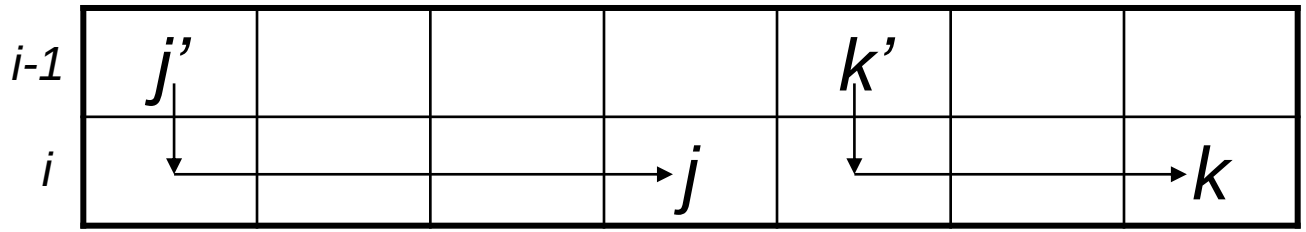
Как решать? (4 случай)

- Воспользуемся методом динамического программирования.
- Будем считать количество способов оказаться в клетках $(i\ ,j)$ и $(i\ ,k)$ одновременно $(j < k)$.
- Рассмотрим откуда мы могли прийти в позицию (i, j, k).
- Переход будет осуществляться из какой-то позиции (i, j', k'), причем $j' \leq j$, а $k' \leq k$.

i-1	*j'*				*k'*		
i				*j*			*k*

Как решать? (4 случай)

- Что значит, что мы переходим из позиции (i, j', k')?
- Это значит, что именно в позициях j' и k' мы делаем шаг вниз и переходим на строку с номером i, а дальше просто идем направо до позиций j и k.
- Заметим, что позиция k' должна быть правее, чем j. Иначе пути пересекутся.

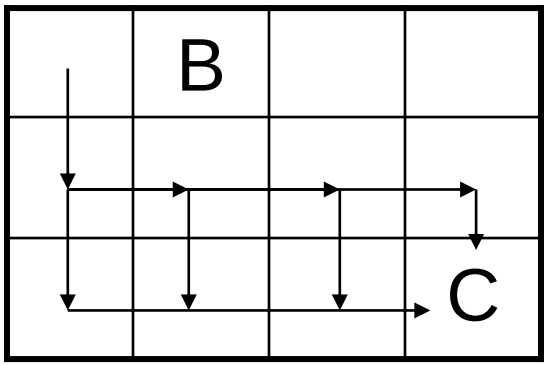


Как решать? (4 случай)

- Таким образом $dp[i][j][k] = \sum_{j'=k}^{j} \sum_{k'=j+1}^{k} dp[i-1][j'][k']$.
- Если перебирать i, j, k, j', k', то сложность получится $O(n^5)$.
- Заметим, что все слагаемые у нас принадлежат предыдущему, $(i-1)$-ому слою, а сумму мы берем на отрезках (например по параметру *k'* на отрезке от $j+1$ до k).
- С помощью предподсчета частичных сумм будем находить сумму на отрезке за $O(1)$.
- Такое решение будет иметь сложность $O(n^4)$.

Как решать? (4 случай)

- Теперь нужно восстановить ответ.
- Пусть мы посчитали динамику для i-ой строки, где $i = \min(A.i, C.i)$. Предположим, что в этой строке расположена клетка A. Тогда через все клетки, отличные от A, проходят пути до C.
- Для каждой пары $(j, A.j)$ посчитаем количество простых путей из $(i+1, j)$ до C. Просуммируем и получим ответ.



Еще одно решение

- Обработаем так же как и в том решении случай, когда все три точки на одной прямой.
- Иначе пути от начального положения до котенка и от котенка до лифта оба кратчайшие.
- Пусть они имеют общую точку. Тогда она находится на одинаковом расстоянии от котенка, в обоих путях.

Еще одно решение

- Развернем первую половину пути. Будем строить два пути – от котенка до начального положения и от котенка до лифта.
- Будем считать, что пути имеют одинаковую длину. Если это не так, то добавим конец пути в путь нужное количество раз.
- Посчитаем динамику $d[x1][y1][x2][y2]$. Количество пар хороших путей из котенка в $(x1, y1)$ и $(x2, y2)$, соответственно.
- Пути являются хорошими, если они не пересекаются и могут быть началами кратчайших путей в нужные точки.

Еще одно решение

- Пересчитывать эту динамику удобно вперед. Из каждого состояния можно продолжить каждый из путей двумя способами.
- Можно продолжить путь, приблизившись по x координате и по y координате к концу.
- Некоторые из переходов могут вести в плохие состояния. Например пути могут пересечься, или нельзя приблизиться по одной из координат, потому что по ней путь уже дошел.
- Время работы такого решения $O(n^4)$.

Еще одно решение

- Ускорим работу данного решения. Заметим, что далеко не все состояния достижимы. Будем переберать только достижимые состояния.
- Это удобно делать обходом в ширину. Подсчитаем весь следующий слой состояний, для этого запишем в массив все соседние состояния в сравнении с нашем слоем, отсортируем его и соединим одинаковые состояния в одно.

Еще одно решение

- Оценим количество достижимых состояний. Пока один из путей не закончился две вершины находятся на одинаковом расстоянии. Пар таких вершин $O(n^2)$. Значит суммарно их $O(n^3)$. Когда один из путей закончился - состояний $O(n^2)$.
- Суммарно расстояний порядка n^3. Тогда время работы решения $O(n^3 \log(n))$ или $O(n^3)$, если заменить сортировку на хеш-таблицу.

Задача G
Топот котов

- Автор задачи – Максим Буздалов
- Условие – Борис Минаев
- Подготовка тестов – Борис Минаев
- Разбор – Борис Минаев

Постановка задачи

- В доме N этажей, на каждом этаже M квартир.
- Квартира i находится над квартирой i – m, если i > m.
- Известно, сколько котов живет в каждой квартире.
- Жителям квартиры придется заплатить штраф *c* рублей, если у них котов больше, чем в два раза больше, чем у соседей снизу.
- Сколько штрафа заплатят жильцы дома?

Как решать?

- Считаем данные в массив a.
- Для всех квартир с номером больше m проверим, должны ли жильцы платить штраф.
- for (int i = m; i <= n*m; i++)

  if (a[i] > 2 * a[i – m]) ans += c;
- Выведем ответ.

Задача H
Странный город

- Автор задачи – Дамир Гарифуллин
- Условие – Нияз Нигматуллин
- Подготовка тестов – Нияз Нигматуллин
- Разбор – Нияз Нигматуллин

Постановка задачи

- Задан неориентированный граф.
- Требуется удалить из него ребра, чтобы каждая вершина имела нечетную степень

Как решать? (идея)

- Решаем для каждой компоненты связности отдельно.
- Если в компоненте нечетное количество вершин, то ответа нет, так как сумма всех степеней должна быть четна
- Научимся решать задачу для дерева, тогда можно будет построить любой остов, а все остальные ребра выкинуть

Как решать? (идея)

- Рассмотрим какое-нибудь поддерево.
- Если оно содержит четное количество вершин, решим задачу для него отдельно, а ребро к родителю из корня поддерева добавлять не будем
- Если же оно содержит нечетное количество вершин, решим задачу для поддерева с тем условием, что корень должен остаться нечетным, тогда добавим ребро в родителя

Как решать? (реализация)

- Итого решение за O(V + E): например, подвесить дерево с помощью поиска в глубину и посчитать четность количества вершин в поддереве.
- Заметим, что и то, и другое всегда можно сделать.
- Это можно просто доказать по индукции.

Как решать? (доказательство)

- Если одна или две вершины, то очевидно.
- Если есть поддерево из **k** вершин:
  - **k** – четно, тогда у корня всегда нечетное количество нечетных поддеревьев, потому что в сумме у поддеревьев (**k** – 1) вершина, для всех поддеревьев задача решается по индукции, поэтому из корня будет исходить нечетное количество ребер.
  - **k** – нечетное, тогда у корня четное количество нечетных поддеревьев, поэтому из корня будет исходить четное количество ребер, а из остальных нечетное, по аналогичной причине в четном случае.

Задача I
Длинный питон

- Автор задачи – Денис Кириенко
- Условие – Анна Малова
- Подготовка тестов – Александр Тимин
- Разбор – Демид Кучеренко

Постановка задачи

- В питоне помещается n попугаев или m мартышек.
- Сколько минимально и максимально попугаев помещается в мартышке?

Как решать?

- Рассмотрим случай для минимума.
- Будем брать максимально длинных попугаев и максимально коротких мартышек, чтобы минимизировать отношение длины мартышки к длине попугая.
- Максимально возможная длина попугая равна $\frac{l}{n}$, где l - длина питона.
- Минимально возможная длина мартышки равна $\frac{l}{m+1} + \varepsilon$.
- Потому что $m + 1$ мартышка в питона уже не влезет.

Как решать?

- Тогда в одну мартышку помещается $\frac{\frac{l}{m+1}+\varepsilon}{\frac{l}{n}} = \frac{\frac{1}{m+1}+\varepsilon}{\frac{1}{n}} = \frac{n}{m+1}+\varepsilon$ попугаев.
- Таким образом ответом будет являться число $\frac{n}{m+1}$, округленное вниз.

Как решать?

- Аналогично рассмотрим случай для максимума.
- Длина одного попугая равна $\frac{l}{n+1} + \varepsilon$.
- Длина мартышки - $\frac{l}{m}$.
- Тогда в одну мартышку входит $\frac{\frac{l}{m}}{\frac{l}{n+1}+\varepsilon} = \frac{\frac{1}{m}}{\frac{1}{n+1}+\varepsilon} = \frac{n+1}{m\,(1+\varepsilon)}$ попугаев.

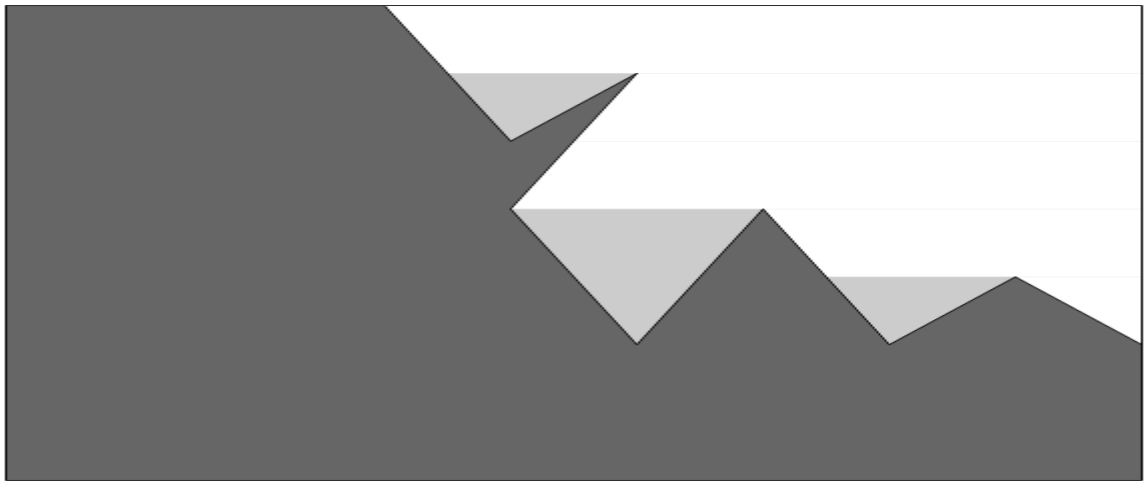
Как решать?

- Таким образом если $n+1$ кратно m, то ответом будет $\left\lfloor \frac{n+1}{m} \right\rfloor - 1$.
- Иначе, ответом будет число $\left\lfloor \frac{n+1}{m} \right\rfloor$.

Задача J
Швабра

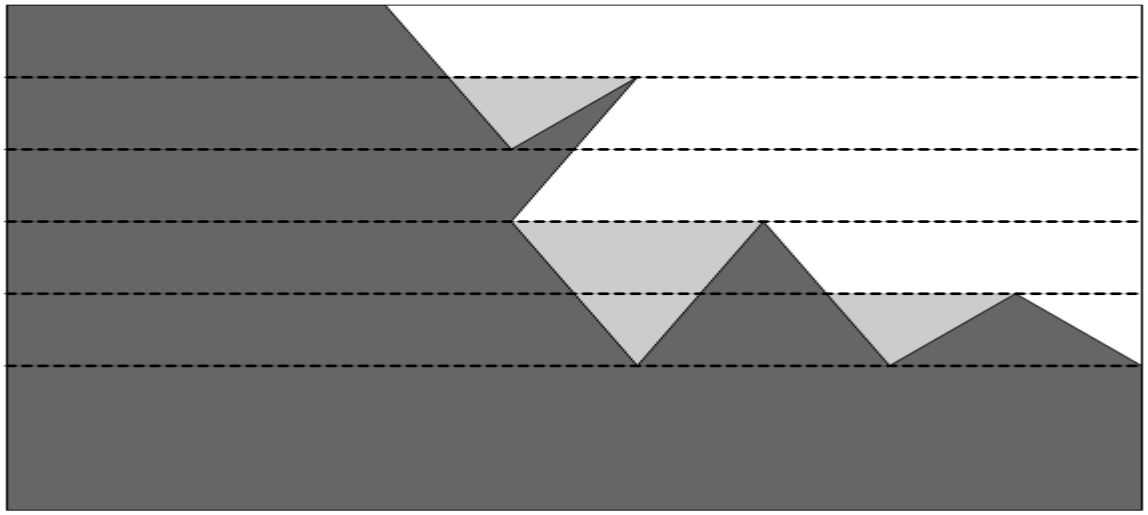
- Автор задачи – Елена Андреева
- Условие – Анна Малова
- Подготовка тестов – Виталий Демьянюк
- Разбор – Виталий Демьянюк

Постановка задачи



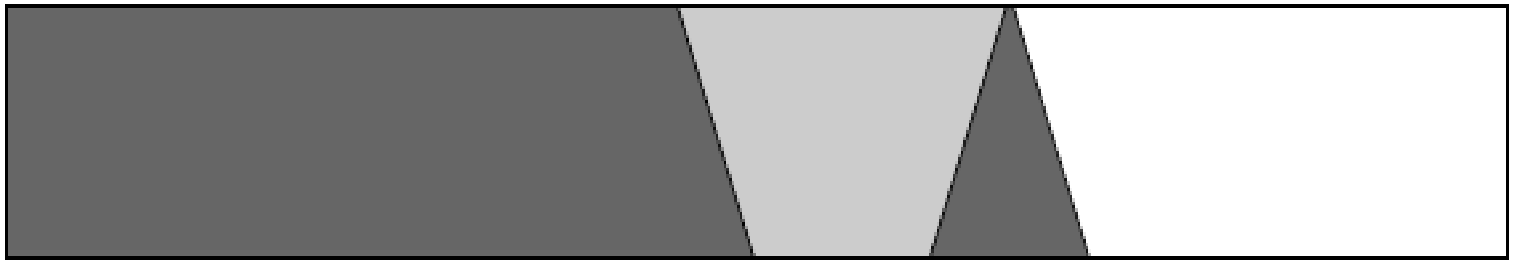
- Есть прямоугольная швабра, у которой вместо правого верхнего угла ломаная.
- Шваброй провели от одной стороны комнаты до другой, не отрывая ее от стены.
- Определить площадь грязной части пола в углу комнаты.

Решение



- Разобьем прямоугольник горизонтальными линиями, проходящими через вершины ломаной.
- Для каждого подпрямоугольника решим задачу отдельно.

Решение для подпрямоугольника

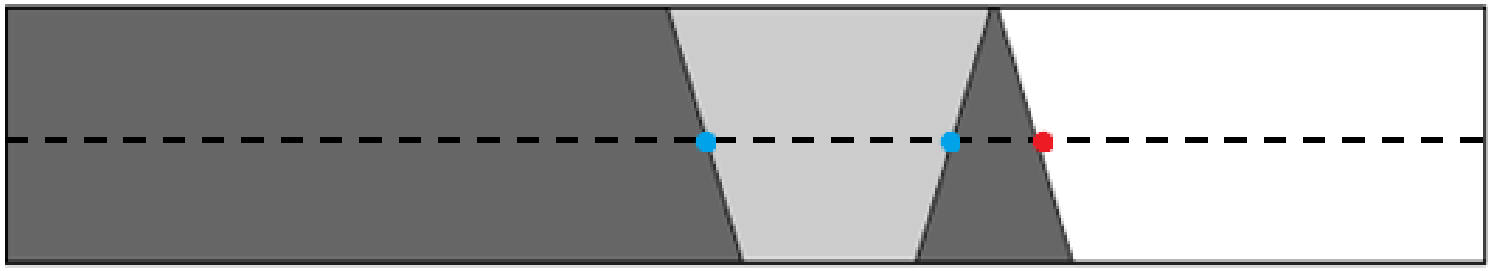


- Отрезки ломаной могут либо полностью пересекать подпрямоугольник, либо не пересекать его совсем. Также, по условию, отрезки ломаной не пересекаются.
- Если количество отрезков ломаной, пересекающих данный подпрямоугольник нечетное, то выберем самый правый отрезок и прибавим к ответу площадь трапеции, состоящей из выбранного отрезка, правой, верхней и нижней границ подпрямоугольника.

Решение для подпрямоугольника

- Если данный подпрямоугольник пересекает четное количество отрезков ломаной, то самая правая часть подпрямоугольника принадлежит швабре, и пол под данным подпрямоугольником чист.

Решение для подпрямоугольника



- При выборе самого правого отрезка удобно их сравнивать по x координате точек пересечения отрезков с горизонтальной прямой проходящей через центр подпрямоугольника.

Асимтотика

- Количество подпрямоугольников — $O(n)$, поиск самого правого отрезка выполняется за $O(n)$ операций, поэтому время работы алгоритма — $O(n^2)$

Решение за $O(n \log(n))$

- Будем рассматривать подпрямоугольники в порядке увеличения y координаты их верхних границ. Также будем поддерживать список отрезков, пересекающих текущий подпрямоугольник.
- При переходе к новому подпрямоугольнику могут произойти два типа событий: добавление отрезка и удаление отрезка.
- Также при переходе к очередному подпрямоугольнику порядок отрезков относительно правой границы прямоугольника не меняется.
- Поэтому, если в TreeSet-е хранить отрезки, пересекающие подпрямоугольник, поиск самого правого отрезка и обработка каждого события будут занимать $O(\log(n))$ операций, и время работы алгоритма будет $O(n \log(n))$.

Задача K
Вампирские числа

- Автор задачи – Никита Иоффе
- Условие – Никита Иоффе
- Подготовка тестов – Никита Иоффе
- Разбор – Никита Иоффе, Аксенов Виталик

Постановка задачи

- Дано определение вампирских чисел.
- Дано k и n.
- Нужно вывести k различных n-значных вампирских чисел.

Как решать? (маленькие n)

- При n равном 4 – найдём отдельно.
- Заметим, что для n равного 6 мы получаем уже 148 шестизначных вампирских чисел.

Как решать?

- Как построить $2n+2$-значные вампирские числа, если имеются $2n$-значные?
- Возьмём любое $2n$-значное число и умножим его на 100.
- Пример: $z = x \cdot y \rightarrow z00 = x0 \cdot y0$.
- Так как мы имеем больше 100 шестизначных вампирских чисел, мы сможем с помощью них с лёгкостью получить больше 100 $2n$-значных вампирских для любого n большего трёх.

Спасибо за внимание! Вопросы?

http://neerc.ifmo.ru/school