XV командная олимпиада школьников по программированию

30 ноября 2014 года

Задача A. «Профессор Хаос»

Задача A. «Профессор Хаос»

- Идея задачи — Антон Гардер, Андрей Станкевич
- Подготовка тестов — Демид Кучеренко
- Разбор задачи — Демид Кучеренко

Постановка задачи

Даны числа a, b, c, d.

- Каждый день значение a изменяется на $ab - c$;
- Если в какой-то день итоговое значение a стало меньше нуля, то a приравнивается к нулю;
- Если в какой-то день итоговое значение a стало больше d, то a приравнивается к d;
- Определить значение a к концу k-ого дня.

Решение

- Итоговая формула пересчета:
  $a = max(0, min(ab - c, d))$;
- Заметим, что если после какого-то дня значение a не изменится, то оно уже никогда не изменится;
- Также заметим, что если значение a увеличилось, то в будущем оно будет только увеличиваться, либо дорастет до d;
- Аналогично с убыванием.

Решение

- Таким образом, будем пересчитывать a, пока значение не зафиксируется;
- Поскольку $d \leqslant 1000$, то может быть не более 1000 изменений a;
- Итого, если должно пройти p дней до момента, когда a прекратит изменяться, то нам нужно смоделировать $min(p, k)$ дней.

Задача B. «Золотые монеты»

Задача B. «Золотые монеты»

- Идея задачи — Елена Андреева
- Подготовка тестов — Никита Иоффе
- Разбор задачи — Никита Иоффе

Постановка задачи

- Дана сетка 3×4. Всего 17 рёбер и 12 вершин;
- На каждом ребре сетке лежит некоторое количество монет;
- Пусть на ребре лежит x монет;
- При проходе по ребру игрок получал $\lceil x/2 \rceil$ монет;
- На ребре после оставалось $\lfloor x/2 \rfloor$ монет;
- По рёбрам, на которых не осталось монет, проходить запрещено;
- Цель — собрать как можно больше монет.

Переформулируем задачи

- Пусть $cnt(x)$ — количество ходов, чтобы забрать все x монет;
- $cnt(x) \approx \log x$;
- Заменим каждое ребро (i, j) в исходном графе на $cnt(c_{i,j})$ новых;
- Цель: удалить некоторое подмножество рёбер, и выбрать среди оставшихся Эйлеров путь.

Переформулируем задачу

- Условие существование Эйлерова пути — не более двух вершин нечётной степени;
- Для каждого ребра (i, j) следует удалить не более, чем одну его копию в новом графе;
- Итого 2^{17} вариантов.

Решение

- Переберём у каких рёбер мы удалим одну копию;
- Граф распадается на компоненты связности;
- Для каждой компоненты проверим, есть ли в ней Эйлеров путь;
- Среди всех Эйлеровых путей выберем тот, у которого суммарное число монет максимально;

Задача C. «Вещественные числа»

Задача C. «Вещественные числа»

- Идея задачи — Андрей Станкевич
- Подготовка тестов — Виталий Демьянюк
- Разбор задачи — Виталий Демьянюк

Постановка задачи

- Для каждого целого x от 1 до r проверить: равно ли единице выражение $\left(\frac{1}{x}\right) \cdot x$;
- При делении результат округляется до n двоичных знаков после запятой, при умножении до k двоичных знаков;
- При округлении к n знакам после запятой следующие за ними двоичные знаки отбрасываются. В случае, если $n+1$-ый знак равен 1, к числу прибавляется 2^{-n}.

Решение

- Моделирование деления и умножения двоичных чисел;
- Используется длинная арифметика.

Решение без длинной арифметики

- Исходное выражение равно единице только в том случае если $-2^{-k-1} < 1 - v \cdot x \leqslant 2^{-k-1}$, где v равно округленной $\frac{1}{x}$;
- Пускай $q = \lfloor \frac{2^n}{x} \rfloor$, $r = 2^n \bmod x$, тогда $\frac{1}{x} = \frac{q}{2^n} + \frac{r}{x \cdot 2^n}$;
  - $2 \cdot r < x$. Тогда $v = \frac{q}{2^n}$ и $1 - v \cdot x \geqslant 0$. Проверим что $(1 - \frac{q}{2^n} \cdot x) \leqslant 2^{-k-1}$ или $r \leqslant 2^{n-k-1}$;
  - $2 \cdot r \geqslant x$. Тогда $v = \frac{q+1}{2^n}$ и $1 - v \cdot x \leqslant 0$. Проверим что $(1 - \frac{q+1}{2^n} \cdot x) > -2^{-k-1}$ или $x - r < 2^{n-k-1}$.

Задача D. «Игра»

Задача D. «Игра»

- Идея задачи — Николай Карпов, Андрей Станкевич
- Подготовка тестов — Артем Васильев
- Разбор задачи — Артем Васильев

Постановка задачи

- Имеется n карт, на каждой из них написано число от 1 до m;
- Можно брать в руку верхнюю карту в колоде, сбросить карту или выложить ее;
- Нельзя держать в руке больше, чем k карт;
- Карты надо выкладывать в порядке номеров, начиная с 1;
- Требуется определить максимальное число карт, которое можно выложить.

Решение

- Если возможно выложить x карт, то также возможно выложить $x - 1$ карту;
- Двоичный поиск по ответу;
- Нужно реализовать функцию проверки того, сможем ли выложить x карт.

Решение

- Для каждого номинала определим самый последний момент, когда эту карту с таким номиналом можно будет выложить;
- Введем понятие «*ожидаемая карта*»;
- Будем рассматривать карты в обратном порядке, начиная с конца колоды;
- Изначально ожидаемыми картами будут x, $x - 1$, $\ldots$, $x - k + 1$.

Решение

- Как только встретилась ожидаемая карта, она перестает быть ожидаемой;
- Ожидаемой становится карта с номером $p - 1$, где p — минимальная ожидаемая карта;
- Если в какой-то момент множество ожидаемых карт стало пустым, то x карт выложить можно;
- В противном случае выложить x карт невозможно.

Пример

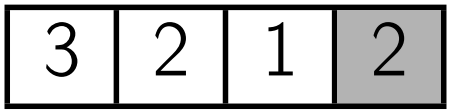
$n = 4, m = 3, k = 2, x = 3$

3	2	1	2

Ожидаемые карты: 2, 3

Пример

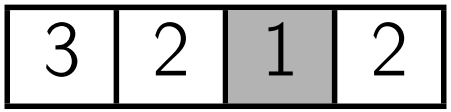
$n = 4, m = 3, k = 2, x = 3$



Ожидаемые карты: 1, 3

Пример

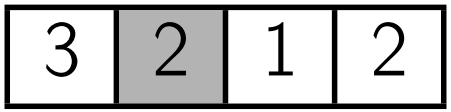
$n = 4, m = 3, k = 2, x = 3$



Ожидаемые карты: 1, 3

Пример

$n = 4, m = 3, k = 2, x = 3$



Ожидаемые карты: 3

Пример

$n = 4, m = 3, k = 2, x = 3$

3	2	1	2

Множество ожидаемых карт пусто, поэтому выложить 3 карты возможно.

Время работы

- Проверку того, можно ли выложить x карт можно реализовать за $O(n + m)$;
- Всего $O(\log m)$ итераций двоичного поиска;
- Итоговое время работы: $O((n + m) \log m)$.

Задача E. «Гипотеза об обобщенном коне»

Задача E. «Гипотеза об обобщенном коне»

- Идея задачи — Михаил Дворкин
- Подготовка тестов — Михаил Дворкин
- Разбор задачи — Михаил Дворкин

Постановка задачи

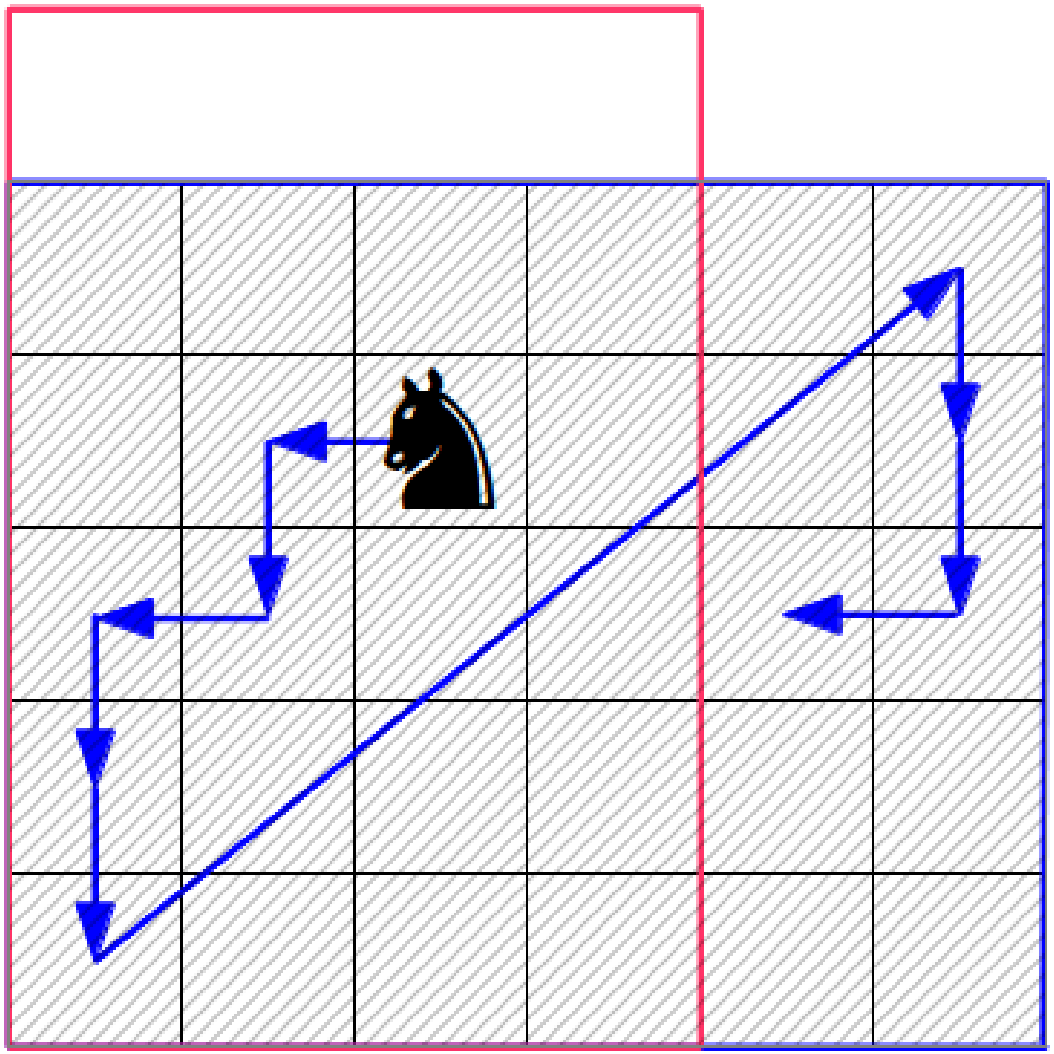
- Обобщенный конь — шахматная фигура, заданная множеством возможных перемещений;
- «Прекрасно перемещается» по доске — может дойти от любой клетки до любой;
- Даны числа a, b, c, d. Проверить гипотезу:

Если обобщенный конь прекрасно перемещается по доске $a \times b$, то он прекрасно перемещается по доске $c \times d$.

Решение

- Если $c > a = 1$, то контрпримером является конь, ходящий только по вертикали: $\{(0, 1), (0, -1)\}$;
- Если $d > b = 1$, то контрпримером является конь, ходящий только по горизонтали: $\{(1, 0), (-1, 0)\}$;
- Если $a > c$ или $b > d$, то контрпример: $\{(a - 1, b - 1), (0, -1), (-1, 0)\}$;
- Иначе, гипотеза верна;
- Особый случай: $c = d = 1$. Из любого утверждения следует истина; гипотеза верна.

Контрпример в общем случае



Задача F. «ЕГЭ»

Задача F. «ЕГЭ»

- Идея задачи — Максим Ахмедов
- Подготовка тестов — Дмитрий Филиппов
- Разбор задачи — Дмитрий Филиппов

Постановка задачи

Входные данные:

- Дано целое число n от -10^{18} до 10^{18};
- Вывести представление числа n в (-2)-ичной системе счисления;
- Представление — такие числа $a_0, a_1, \ldots, a_{k-1}$ ($a_i \in \{0, 1\}$), что

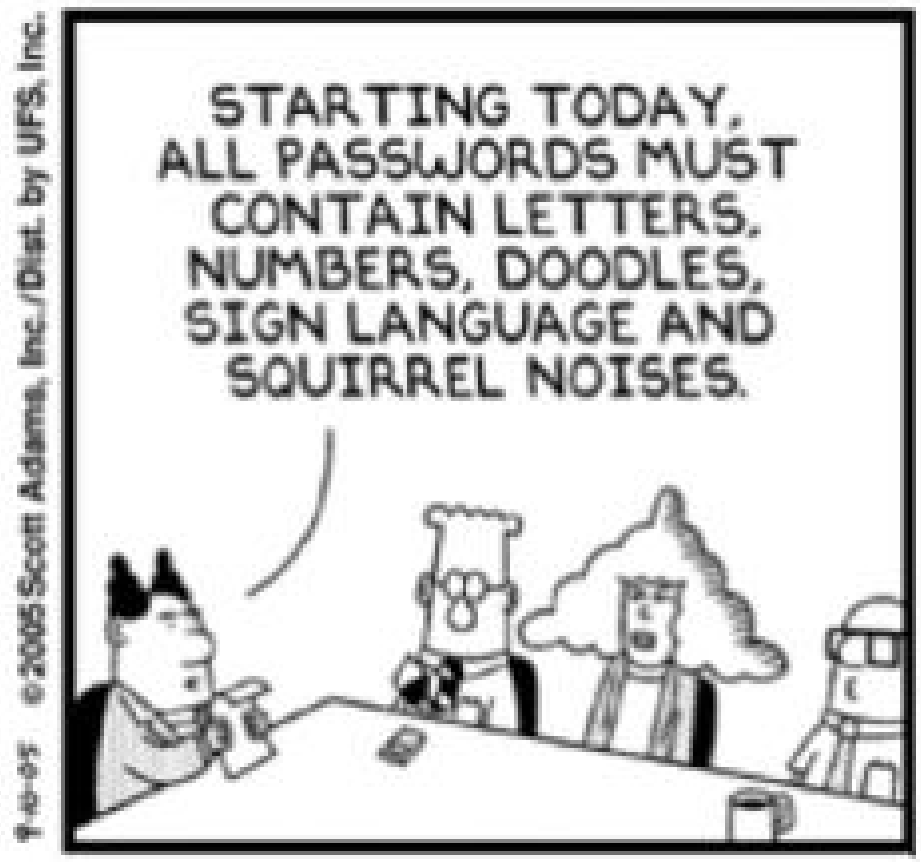
$$n = \sum_{i=0}^{k-1} a_i (-2)^i$$

Решение задачи

- Найдем a_0, после этого, заменив n на $\frac{n-a_0}{-2}$, найдем a_1, и так далее;
- Когда n будет равно 0, завершаем алгоритм, найденные числа a_i — ответ;
- Чтобы найти a_i, нужно правильно взять остаток числа n по модулю -2;
- Остаток равен 1, если число нечётное, а иначе — 0;
- Следует аккуратно следить за знаками;
- Время работы: $O(\log n)$.

Задача G. «Генератор паролей»

Задача G. «Генератор паролей»



- Идея задачи — Андрей Станкевич
- Подготовка тестов — Андрей Комаров
- Разбор задачи — Андрей Комаров

Постановка задачи

Даны числа n, a, b, c. Придумать любой пароль, в котором:

- Ровно n символов;
- Хотя бы a больших букв;
- Хотя бы b маленьких букв;
- Хотя бы c цифр;
- Соседние символы различны.

Решение

Возможное решение:

- $\underbrace{ABAB\ldots AB}_{a}\underbrace{abab\ldots ab}_{b}\underbrace{0101\ldots01}_{n-a-b}$
- Повторить AB столько раз, чтобы было ровно a символов;
  - Возможно, в конце не будет B: $ABABA$;
- Аналогично повторять ab, чтобы получить b маленьких букв;
- Остальные $(n - a - b)$ символов дополнить повторением 01.

Неправильные решения

- $\underbrace{ABAB\ldots AB}_{a}\underbrace{abab\ldots ab}_{b}\underbrace{0101\ldots 01}_{\mathbf{c}}$
  - Условие допускает, что $a + b + c \neq n$.
- Выбрать a случайных больших букв, b случайных маленьких букв, c случайных цифр, $n - a - b$ случайных символов, перемешать
  - Тест: $a = 0$, $b = 0$, $c = 100$, $n = 100$;
  - 100 случайных цифр;
  - Каждая 2..100 цифра не равна предыдущей;
  - Вероятность этого равна $0.9^{99} \approx 3 \cdot 10^{-5}$;
    - Довольно маловероятно.

Задача H. «Уборка снега»

Задача H. «Уборка снега»

- Идея задачи — Тимур Гарипов
- Подготовка тестов — Борис Минаев
- Разбор задачи — Борис Минаев

Постановка задачи

- Облако движется над дорогой;
- Из облака падает снег;
- Посчитать общую длину дороги, которая покроется снегом.

Формализация задачи

- Облако является выпуклым многоугольником (10^5 вершин);
- Дорога является прямой на плоскости;
- Движение облака можно разбить на участки (10^5 участков);
- На каждом участке облако движется равномерно, не поворачиваясь и не изменяя форму.

Решение

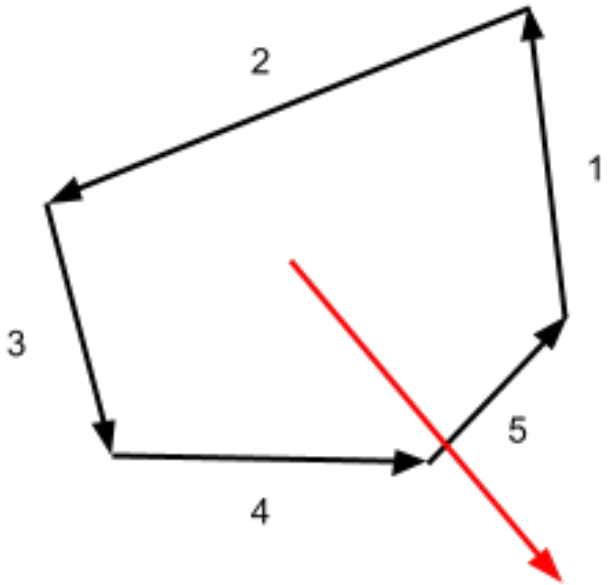
- Решим задачу отдельно для каждого участка движения;
- За время одного участка снегом будет покрыт некоторый выпуклый многоугольник;
- Чтобы его получить, необходимо «растянуть» облако вдоль вектора движения;
- Пересечение выпуклого многоугольника с дорогой — непрерывный отрезок;
- Обьединим отрезки, полученные для каждого участка, и посчитаем суммарную длину — ответ на задачу.

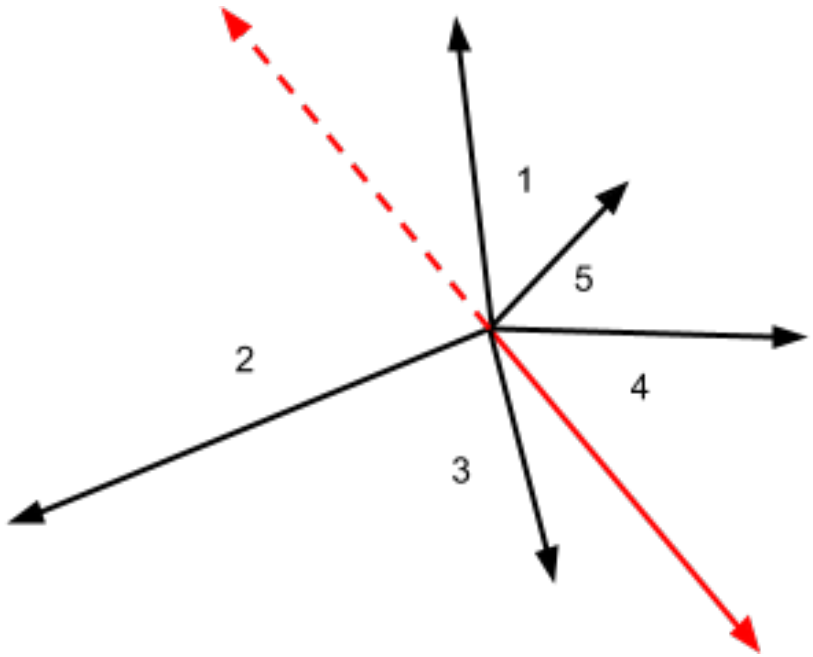
Решение. Детали реализации.

Как «растянуть» многоугольник вдоль вектора?

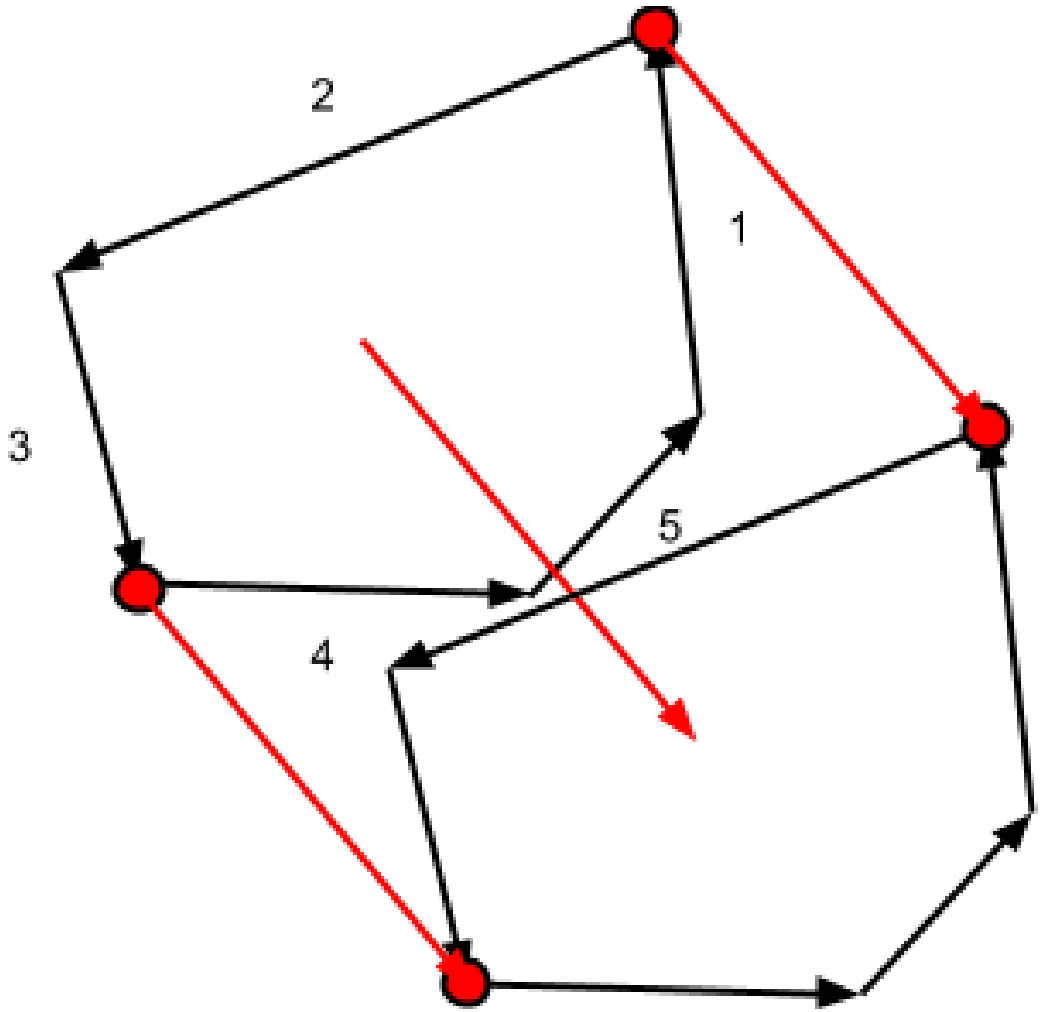
- Запишем ребра многоугольника в порядке его обхода;
- Каждое ребро можно считать вектором, который необходимо прибавить, чтобы получить из одной точки другую;
- Заметим, что все вектора отсортированы по углу;
- Необходимо добавить два вектора, которые «растянут» многоугольник, так, чтобы вектора все еще были отсортированны;
- Позиции, в которые необходимо добавлять вектора, можно найти двоичным поиском.

Растягивание многоульника





Растягивание многоульника



Решение. Детали реализации.

Как пересечь новый многоугольник с прямой?

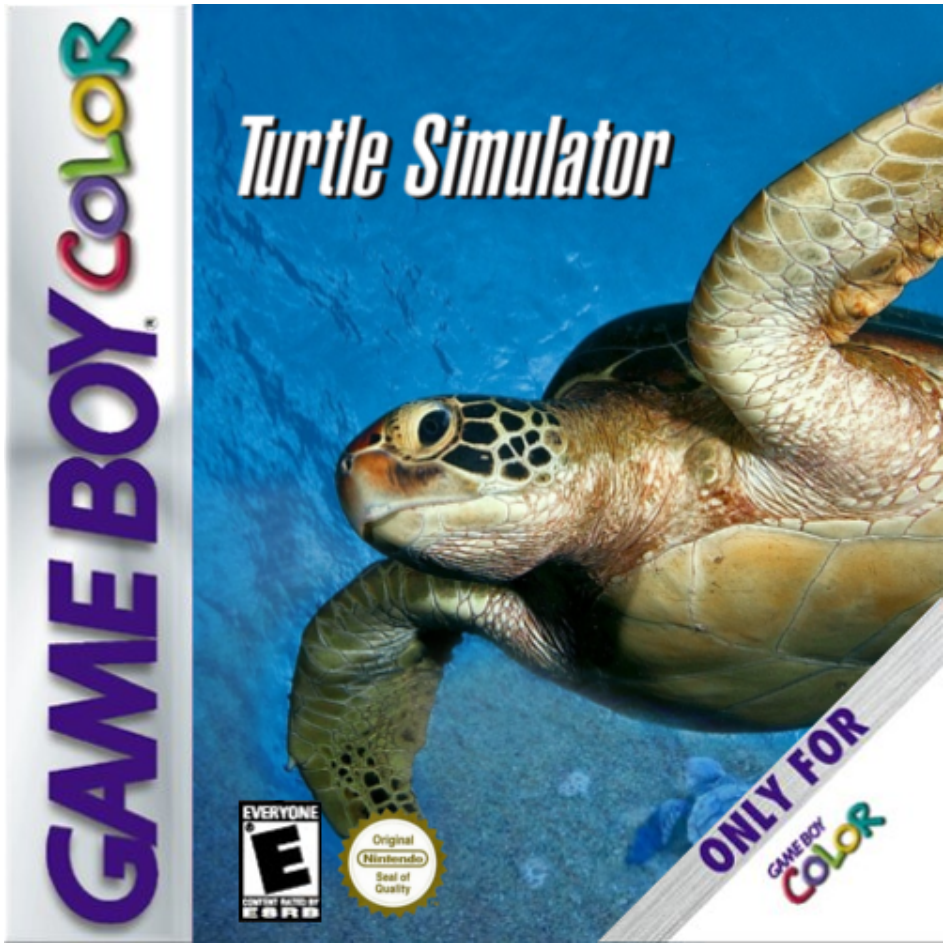
- Найдем две вершины многоугольника с наибольшим и наименьшим ориентированным расстоянием до прямой $(\frac{A \cdot x + B \cdot y + C}{\sqrt{A^2 + B^2}})$;
- Разобьем многоугольник на две части по этим вершинам;
- Заметим, что в каждой части многоугольника ориентированное расстояние от вершины до прямой монотонно изменяется при обходе вдоль многоугольника;
- Значит, можно использовать двоичный поиск, чтобы найти точки пересечения.

Решение. Асимптотика

- n — количество вершин многоугольника;
- m — количество участков пути;
- время работы решения — $O(n + m \log n)$.

Задача I. «Черепахи в пруду»

Задача I. «Черепахи в пруду»



- Идея задачи — Максим Ахмедов
- Подготовка тестов — Максим Ахмедов
- Разбор задачи — Максим Ахмедов

Постановка задачи

- Дано клетчатое поле, на котором выбрано некоторое *связное* множество клеток;
- Множество клеток называется *удобным*, если от любой клетки можно дойти до любой «*черепашкой*», т. е. двигаясь только вверх-вправо, вверх-влево, вниз-влево или вниз-вправо;
- Поступают запросы на добавление клетки в множество с сохранением связности;
- После каждого добавления требуется определять, является ли множество *удобным*, или нет.

Условие *удобности*

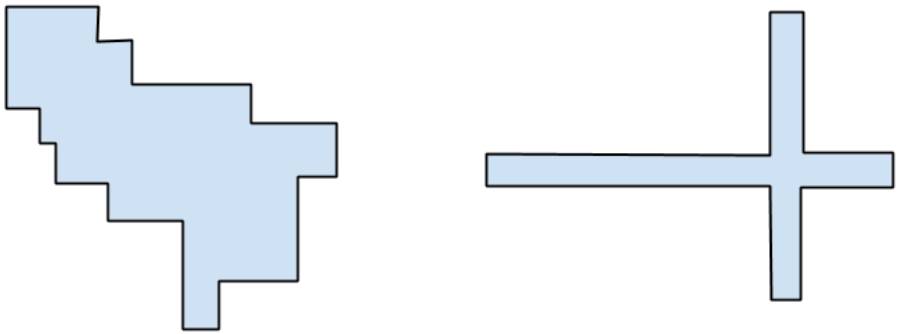
- Решение, проверяющее после каждого добавления набор клеток на *удобность* наивным образом, работает за $O(qn^2)$ и получает вердикт Time Limit Exceeded;
- Ключевым шагом в решении задачи является исследование того, что означает условие *удобности*.

Условие *удобности*

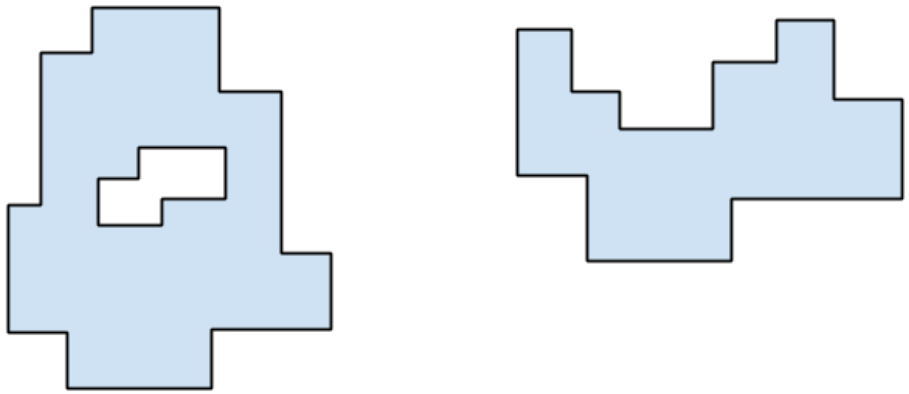
- Применим условие *удобности* к двум клеткам, расположенным на одной вертикали;
- Из одной из них можно добраться до другой, двигаясь только вертикально, следовательно, в любом столбце, с которым множество пересекается, оно представляет собой **непрерывный отрезок**;
- Аналогичное условие верно для любой строки;
- Множества клеток, удовлетворяющие этим двум условиям, иногда называют *клетчато-выпуклыми*;
- Таким образом, для *удобности* набора клеток, необходима его *клетчатая выпуклость*.

Условие *удобности*

Примеры клетчато-выпуклых множеств:



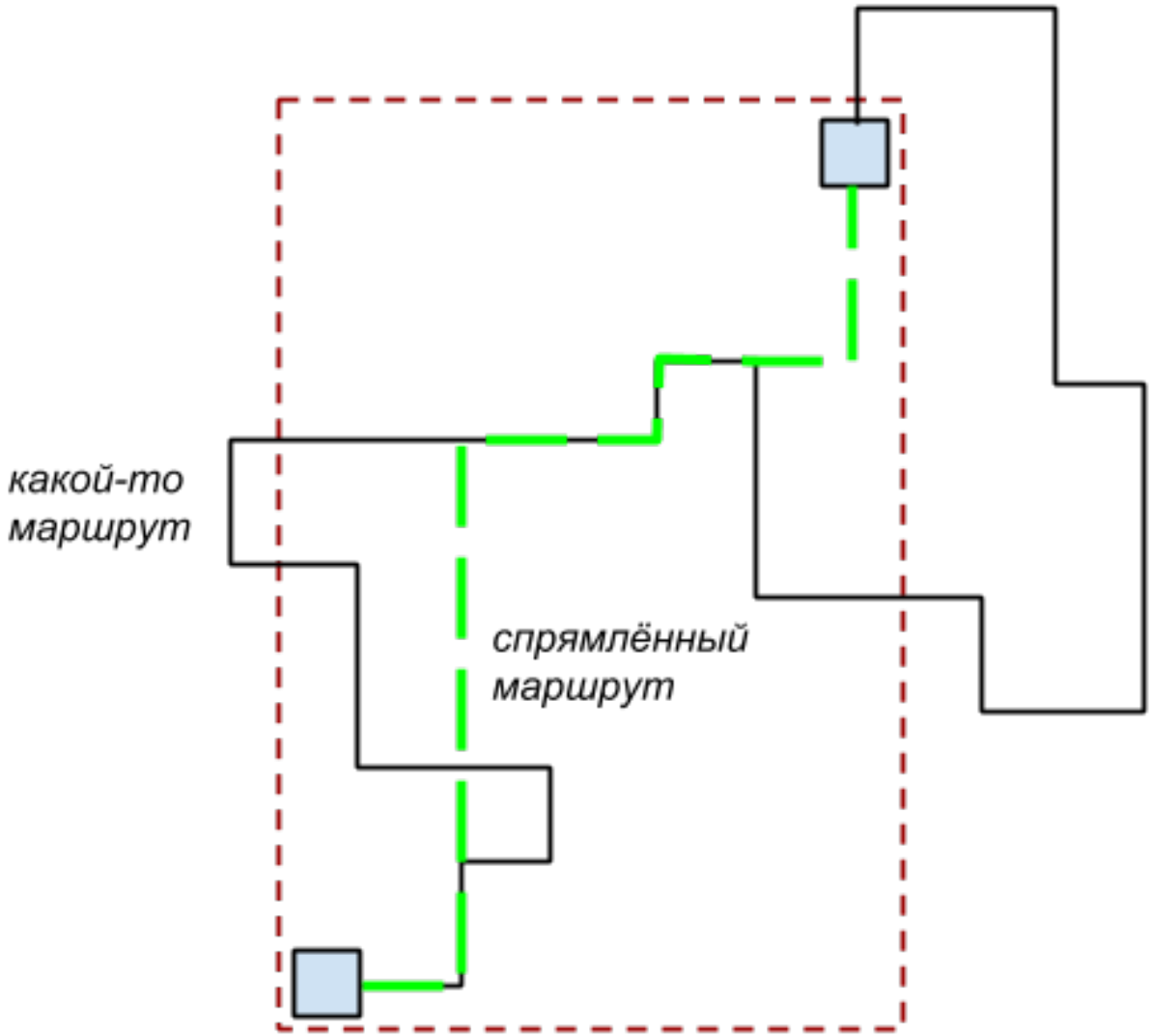
Примеры не клетчато-выпуклых множеств:



Условие *удобности*

- Условие *клетчатой выпуклости* является достаточным для *удобности* набора клеток;
- Любые две клетки соединяет какой-то маршрут;
- Его можно «спрямить» до маршрута черепашки, пользуясь условием *клетчатой выпуклости*.

Условие *удобности*



Решение

- Теперь можно решать задачу по отдельности: для каждой строки и каждого столбца;
- Для каждой строки будем поддерживать количество клеток и позиции самой левой и самой правой клетки в ней;
- Обозначим эти величины для i-й строки за $rowcnt[i]$, $rowleft[i]$ и $rowright[i]$;
- i-я строка является *хорошей*, если:
  - либо $rowcnt[i] = 0$,
  - либо $rowright[i] - rowleft[i] + 1 = rowcnt[i]$.
- Аналогично действуем для столбцов.

Решение

- Поддерживаем количество хороших строк и хороших столбцов;
- Множество является *удобным*, если хороших строк ровно w, а хороших столбцов ровно h;
- Описанное решение работает за $O(1)$ на добавление клетки

Задача J. «Починка забора»

Задача J. «Починка забора»

- Идея задачи — Григорий Шовкопляс
- Подготовка тестов — Григорий Шовкопляс
- Разбор задачи — Григорий Шовкопляс

Постановка задачи

Входные данные:

- Дан забор из n сегментов с высотами a_i;
- Есть m дополнительных досок с длинами b_i.

Нужно получить две последовательности x_i и y_i длины k, такие что:

- $1 \leqslant x_1 < x_2 < \ldots < x_k \leqslant n$;
- $1 \leqslant y_1 < y_2 < \ldots < y_k \leqslant m$.

Чтобы $\min(\min\limits_{t \in \{1,\ldots,k\}} a_{x_t} + b_{y_t}, \min\limits_{t \notin \{x_1,\ldots,x_k\}} a_t)$ был максимально возможным.

Решение задачи

Разобьем задачу на две подзадачи.

- Бинарный поиск по высоте забора.
- Проверка того, что можно получить такую высоту забора.

Решение второй задачи

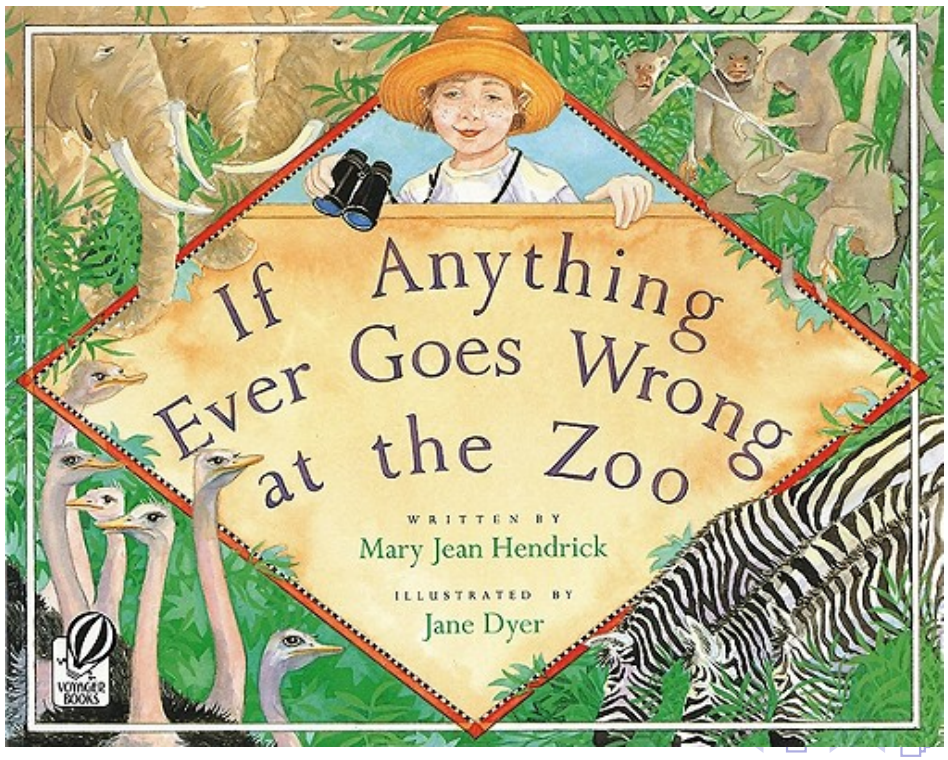
- Есть число h — желаемая высота забора.
- Будем последовательно перебирать сегменты забора. В i-ом сегменте возможны две ситуации:
  - $a_i \geqslant h$ — все хорошо, можно двигаться дальше.
  - $a_i < h$ — нужно прибить сверху доску b_j, чтобы новая высота была $a_i + b_j \geqslant h$.
  - Чтобы найти такое j, будем вытаскивать поочерёдно доски из стопки, пока не найдём подходящую.
- Если в какой-то момент стопка пустая, а $a_i < h$, значит, невозможно починить забор таким образом, что его высота будет не меньше h.
- Иначе, можно.

Вывод

В итоге мы $\log \infty$ раз выполним проверку, которая работает, за $O(max(n, m)) = O(n)$. Итоговая сложность решения — $O(n \log \infty)$. (в данной задаче $\infty = 10^9$.)

Задача K. «Прогулка по зоопарку»

Задача K. «Прогулка по зоопарку»



- Идея задачи — Андрей Станкевич
- Подготовка тестов — Николай Ведерников
- Разбор задачи — Виталий Аксенов

Постановка задачи

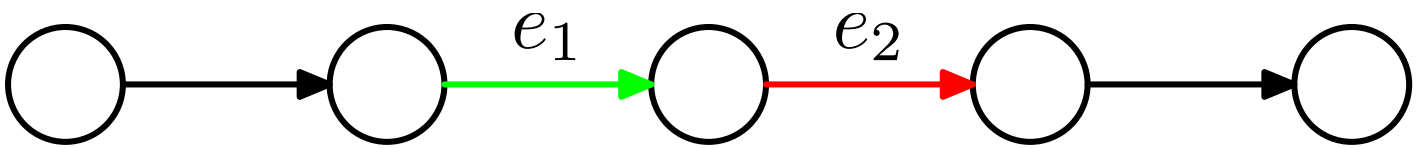
- Задан граф на n вершинах и m рёбрах с пометками;
- Все пометки различны;
- Также дано k путей, заданных начальной и конечнами вершинами и пометками.

Нужно было поменять две пометки местами (два ребра местами), чтобы можно было пройти по пути.

Решение

- Обязательно найдётся путь, по которому мы в какой-то момент не смогли пройти, так как было известно о том, что ровно две пометки были поменяны;
- Рассмотрим этот путь;
- Пройдём по нему до того места, где мы не можем пройти, то есть не можем пойти по ребру e_2;
- Рассмотрим ребро e_1, предшествующее ребру e_2 на пути;
- Очевидно, что хотя бы одно из этих рёбер находится не на своём месте;
- Проверим для каждого из них путь.

Пояснение



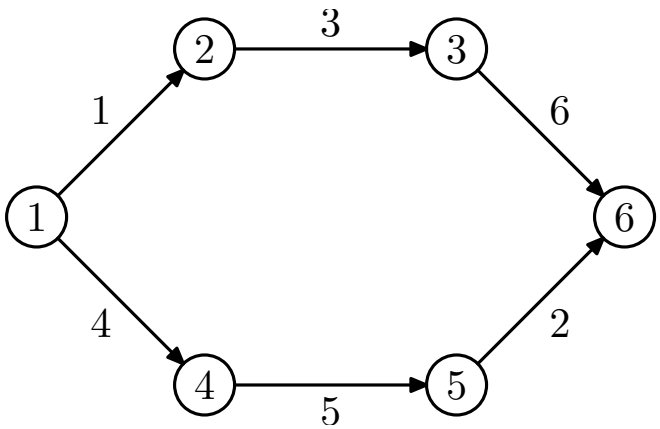
Проверка ребра

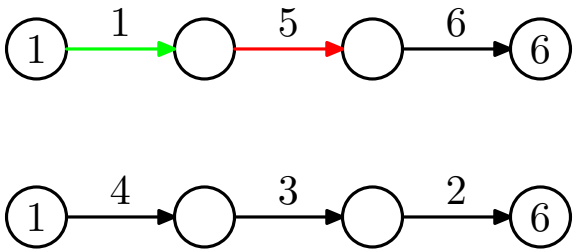
Как проверить ребро на замену?

- Если ни один сосед ребра на этом пути не будет поменян, то мы однозначно можем восстановить ребро, на которое надо заменить;
- Иначе, пытаемся его поменять на каждого из его соседей;
- Когда меняем — проверяем правильность путей на связность и простоту за суммарную длину;
- Так как всего плохих рёбер будет не более двух, а различных вариантов на замену не более трёх, то решение будет работать за $O(\text{сумма длин путей})$.

Пример

Возьмём пример из условия.





Пары, которые мы попытались поменять вторым способом: $\{(1,5),(5,6)\}$.

Пары, которые мы попытались поменять первым способом: $\{(1,4),(5,3)\}$.

Вопросы?