База практических задач

Алгоритм Евклида

Нашей задачей было...

- Организовать работу для всей команды и расставить приоритеты.
- Распределить нагрузку между всеми участниками команды.
- Подобрать несколько обязательно смежных с темой заданий, чтобы ввести студента в олимпиадное программирование.
- Найти, отобрать, решить и разобрать основной костяк задач по теме "Алгоритм Евклида".
- Классифицировать весь подготовленный материал по сложности и собрать все в единую базу практических задач для студентов по теме "Алгоритм Евклида".
- Отобрать несколько задач для контеста и принять в нем участие.
- Подготовить удобную статистику для рецензирования и самим прорецензировать других.

Организация

- Основными средствами связи стали **VK** и **Discord**.
- Был создан репозиторий на **GitHub** для удобного взаимодействия с файлами между членами команды.
- Разборы создавались в **MS PowerPoint**, придерживаясь одного стиля.
- Задачи брались в основном с платформы **codeforces.com**, редко из других источников.
- Задачи были разделены на два уровня сложности: **A** и **B**.
- Внутри каждого уровня задачи пронумерованы по возрастанию сложности.
- К каждой задаче прилагается решение (код), разбор этого решения и комментарий от человека, который разбирал.

Алгоритм Евклида

Алгоритм предоставляет удобный и понятный способ вычисления **НОД** от двух чисел. На различных ЯП существует большое множество способов его реализации. Если рассматривать только C/C++, то здесь уже больше 5 вариаций.

Мы выделяем 3 основных:

- Классический
- Классический укороченный
- Бинарный

Также обязательно нужно отметить Расширенный, который выполняет дополнительную и важную функцию: поиск частного решения Диофантового уравнения.

Бинарный	Упрощенный	Классический
337	333	337
344	314	317
336	316	305
336	314	311
335	339	304

Сравнение

Это время, за которое алгоритм справляется с 10^6 пар чисел, где сами числа от 0 до 10^7.

Результаты представлены в миллисекундах.

Возникшие трудности

Трудности:

- Небольшое недопонимае между участниками до момента основательного изменения подхода к работе и ее организации.
- Хороших и интересных задач на алгоритмы Евклида оказалось не так уж и много на **codeforces.com.**

Их решения:

- Организационное совещание.
- Использование других источников, в том числе и сочинение своих задач.

Уровень A

- Задачи на ознакомление с природой олимпиадного программирования.
- Задачи на теорию чисел и свойства **НОД**.
- Сложность колеблется от легкого уровня до среднего.

Пример задачи уровня A

Манао имеет экран с отношением ширины на высоту **A : B**. Он собирается посмотреть фильм, картинка которого имеет соотношение **C : D**. Манао настроит просмотр так, чтобы фильм сохранил оригинальное соотношение сторон, но при этом занимал максимальное пространство на экране и умещался на нем полностью.

Вычислите, какая часть дисплея останется незанятой во время просмотра. Выведите ответ в виде несократимой дроби, **НОД** числителя и знаменателя которой равен **1**.

Некоторые условия

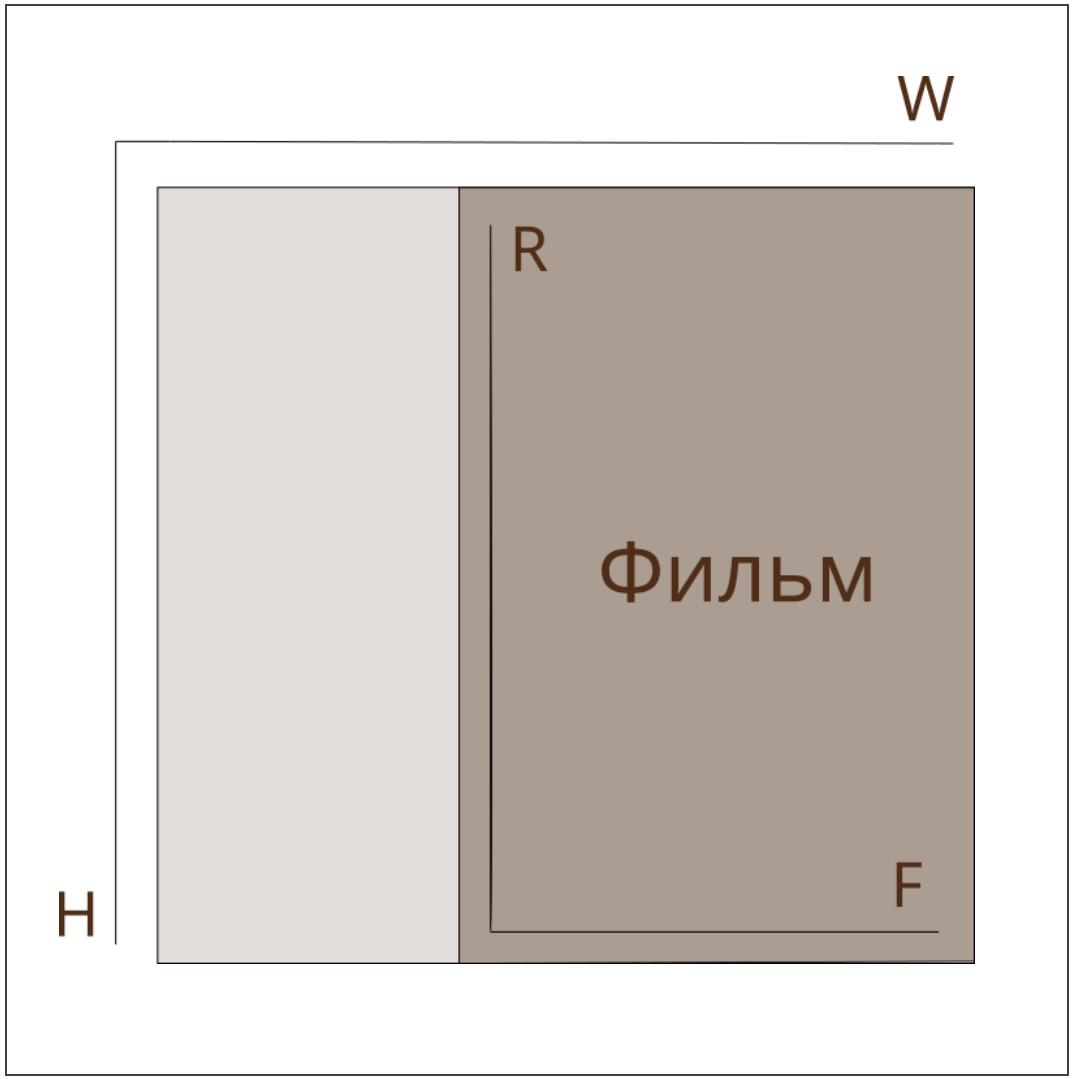
Входные данные:

Четыре целых числа **A, B, C, D** (**1 ≤ A, B, C, D ≤ 1000**), записанных через пробел.

Выходные данные:

Выведите ответ в виде несократимой дроби p/q, где p — целое неотрицательное число, q — целое положительное число.

Ограничения: 1 секунда, 256 мегабайт.



Разбор

Пусть **W** и **H** будут шириной и высотой экрана, а **F** и **R** - шириной и высотой фильма.

Это значит, что

W : H = A : B,

F : R = C : D.

Из этого нетрудно получить выражения:

$$W = \frac{HA}{B}, \quad F = \frac{RC}{D}.$$

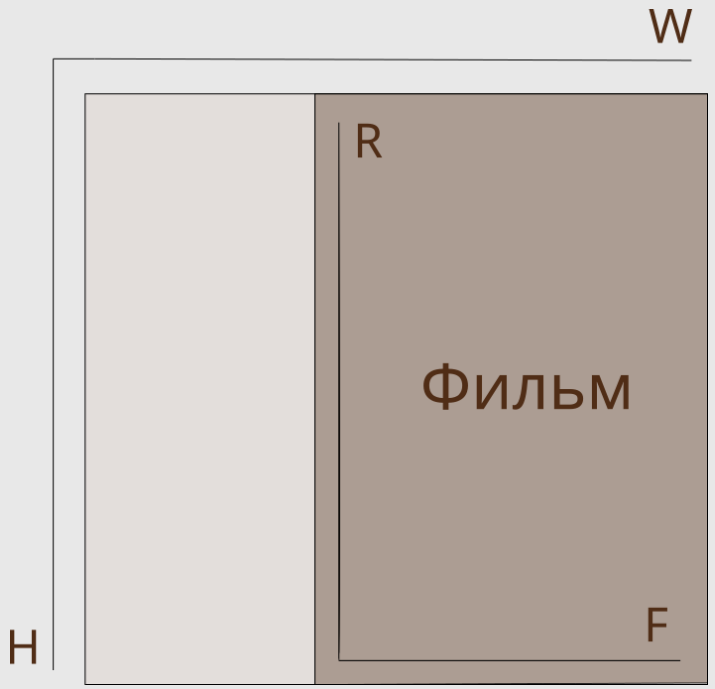
Случай первый

- **A : B = C : D**

Очевидно, что фильм может полностью уместиться на дисплее и занять его полностью.

Так как ответом требуется несократимая дробь вида $^{p}/_{q}$, нам подойдет $^{0}/_{1}$.

Второй



◦ **A : B > C : D**

Понятно, что ширина дисплея в такой ситуации больше ширины фильма, хоть как-то умещенного на этом дисплее.

Чтобы занять им как можно больше места на экране, нужно расширить его до высоты самого экрана (**R = H**). Теперь свободную часть можно вычислить с помощью выражения:

$$\frac{W - F}{W} = \frac{\frac{HA}{B} - \frac{HC}{D}}{\frac{HA}{B}} = \frac{AD - CB}{AD} = {p'}/{q'},$$

где ${p'}/{q'}$ – это потенциально сократимая дробь.

Последний

- **A : B < C : D**

Видно, что ситуация обратная предыдущей: умещенное видео теперь меньше экрана по высоте.

Займем всю ширину экрана (**F = W**).

Свободное пространство теперь можно найти с помощью выражения:

$$\frac{H-R}{H} = \frac{\frac{WB}{A} - \frac{WD}{C}}{\frac{WB}{A}} = \frac{BC-AD}{CB} = {}^{p'}\!/_{q'}$$

где ${}^{p'}\!/_{q'}$ — это потенциально сократимая дробь.

Приведение к ответу

Мы получили дробь **p′/q′**, но мы знаем, что она может быть сократимой.

Найдем **НОД** числителя и знаменателя, чтобы на него сократить обе части.

Дробь после этого станет несократимой, ведь общих делителей у частей не осталось.

Для этого воспользуемся алгоритмом Евклида.

Комментарий: первая задача, в которой студенту выпадет возможность воспользоваться алгоритмом Евклида на практике. Является базовой задачей на этот алгоритм.

Уровень B

- Задачи более высоко уровня сложности.
- Требует лучшего понимания материала курса.
- Алгоритм Евклида зачастую не самая сложная часть задач.

Пример задачи уровня B

Алиса и Боб нашли мешок с апельсинами и яблоками. Алиса взяла себе апельсин, а Боб — яблоко. Чтобы процесс разделения оставшихся фруктов был интереснее, ребята решили поиграть в игру. Они выложили в ряд карточки и на каждой записали букву ***A*** или ***B***.

Они стали убирать карточки по одной слева направо. Когда убиралась карточка с ***A***, Алиса отдавала Бобу все свои фрукты и брала из мешка такой же набор фруктов. Таким образом количество апельсинов и яблок, имеющихся у Алисы, не менялось. Если же на карточка была с ***B***, то Боб поступал аналогично.

После того, как была убрана последняя карточка, все фрукты в мешке закончились.

Известно, сколько сначала в мешке апельсинов и яблок. Требуется найти любую последовательность карточек, которыми могли играть Алиса и Боб.

Дополнительно

Входные данные

Записанные через пробел целые **X, Y (1 ≤ X, Y ≤ 10^{18}, X * Y > 1)** — изначальное количество апельсинов и яблок в мешке.

Выходные данные

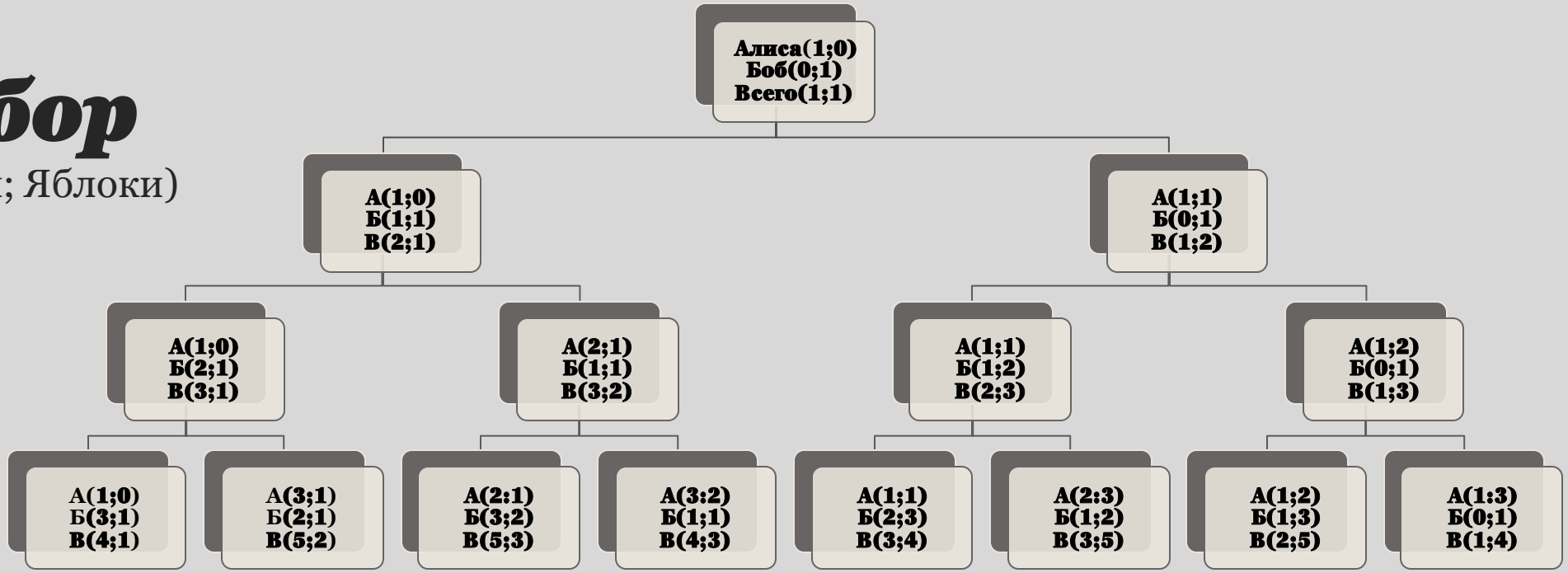
Последовательность карточек в виде строки из символов **A** и **B**, но нужно сжать строку, заменив отрезки одинаковых символов на количество повторений этого символа и его самого. Например, **AABAABB** нужно заменить на **2A1B2A3B**. Строка должна не превышать 10^6 символов.

Если ответ существует, то существует и его сжатое представление, не превышающее 10^6 символов. Если ответов несколько, выведите любой из них. Если не существует ответа, выведите **Impossible**.

Ограничения: 1 секунда, 256 мегабайт.

Разбор

(Апельсины; Яблоки)



В дереве описаны все возможные развития событий до 4-го хода. Если спускаться влево, то это поднятие карточки с **A**, аналогично с **B** вправо. Нас интересует количество **Всего** фруктов после каждого хода.

Видно, что с каждым спуском происходит обратный шаг алгоритма Евклида. Верхний листок дерева имеет **НОД = 1**, это значит, что и все листья имеют такой же **НОД**. Это следует из свойства: **НОД(u, v) = НОД(u – v, v)** при **v ≤ u**.

Алгоритм

Так как все листья имеют **НОД = 1**, то в мешке не могут лежать столько апельсинов и яблок, что их **НОД** был бы не равен **1**. В противном случае мы выводим **Impossible**.

Так как каждый спуск в дереве - это ход в игре, то количество ходов будет равно количеству шагов алгоритма Евклида, пока количества апельсинов и яблок не станут равными **1**.

Если в алгоритме вычитается из левого числа правое, то это Алиса передает свои фрукты. Если из правого левое, — Боб.

То есть сначала нам нужна будет проверка на **НОД = 1** с помощью обычного алгоритма Евклида, а потом запустится наша его модификация, которая будет считать подряд идущие вычитания.

Подводный камень

Вы знаете, что алгоритм Евклида, использующий только вычитание, медленнее такого же, но использующего деление.

Выбрав медленный, мы не уложимся во время, но, выбрав более быстрый, мы столкнемся с одной проблемой: наши числа могут поделиться нацело, тогда одно из них станет равным нулю, а мы возвращаемся к ситуации, когда у каждого по одному фрукту.

Это может возникнуть только тогда, когда делитель был равен **1 (**потому что числа внутри пары взаимно простые). В таком случае мы будем отнимать **1** от результата деления (результат деления — это кол-во повторений символа подряд).

Комментарий: Специфика задачи требует глубокого анализа вариантов ее развития. Используется алгоритм Евклида. Поможет студенту лучше анализировать все возможные варианты развития событий, потому что требует умение составлять бинарное дерево.

Задачи

Приоритетом для нас было качество разборов задач, их интересность и полезность.

Всего было полностью разобрано **16** задач, из них:

- **3** придумано нами
- **2** взято с **acmp.ru**
- **11** взято с **codeforces.com**

На контесте было нами решено **12** задач.

Участники

Афанасьев Андрей:

- Разобрал 7 задач
- Нашел или придумал **3** задачи
- Подготовил презентацию

Арутюнян Владимир:

- Разобрал **6** задач
- Нашел или придумал **5** задач

Кодуков Александр:

- Разобрал **3** задачи
- Нашел **8** задач