Вариант заключительного этапа
9–10 классы

Всероссийской олимпиады школьников «Высшая проба»
по профилю «Промышленное программирование»
2024/2025 уч. г.

Блок «Теория»

Задача 1. Каменная азбука

макс. 5 баллов

Вам не приходилось, прогуливаясь в лесу или в горах, видеть необычные горки камней, которые словно какая-то невидимая рука поставила друг на друга? Местные жители говорят, что это тролли так переговариваются друг с другом.

Для нашего удобства обозначим их буквами латинского алфавита.

В каменной азбуке слова с длиной 5, составленные из камней этих семи видов, идут в таком порядке:

1	AAAAA	6	AAAAF	11	AAABD
2	AAAAB	7	AAAAG	12	AAABE
3	AAAAC	8	AAABA	13	AAABF
4	AAAAD	9	AAABB	14	AAABG
5	AAAAE	10	AAABC	15	AAACA

 B
 A
 D
 G
 E

В ответ запишите только **число**, решение прикладывать не нужно.

Задача 2. Первые шаги

макс. 5 баллов

Драконы — волшебные существа, им доступно множество разных способов перемещаться в пространстве.

Дракончик Алоизий ещё очень мал, поэтому освоил только несколько таких способов. Например, он хорошо умеет делать один шаг вперёд, назад, вправо, влево, вверх, вниз, влево через стену, вправо через стену, по диагонали влево вверх.

Родители Алоизия с умилением наблюдают за его первыми шагами и заодно подсчитывают объём информации, который займёт на диске запись его маршрута в килотритах.

Надо сказать, что информацию драконы измеряют в тритах. *Один трит — это количество информации, которое мы получаем при выпадении одного события из трёх возможных.*

Тогда если различных событий, например, 81, то понадобится 4 трита для хранения номера каждого из этих событий.

Килотрит содержит в себе 3^{10}=59049 тритов.

Определите количество килотритов, которое понадобится для хранения информации о 2480058 шагах малыша Алоизия.

В ответ запишите **только число** без единиц измерения. Ничего, кроме числа, писать в ответе не нужно.

Блок «Алгоритмы»

Задача 1. Трибуна

макс. 10 баллов

Ограничение времени	1 секунда
Ограничение памяти	256 Мб
Ввод	стандартный ввод
Вывод	стандартный вывод

Трибуна с болельщиками представляет собой прямоугольник из нескольких рядов с одинаковым числом мест в каждом ряду. Расстояние между рядами и между местами в ряду составляет 1 метр. Площадь трибуны — S квадратных метров.

Все места на трибуне заняты болельщиками. Известно, что X болельщиков со всех сторон окружены другими болельщиками (то есть их места находятся строго внутри прямоугольника-трибуны), а все остальные болельщики находятся по периметру трибуны.

Найдите количество болельщиков, сидящих по периметру трибуны.

Формат ввода
Первая строка содержит целое число S ($1 \le S \le 10^9$) — площадь трибуны в квадратных метрах.

Вторая строка содержит целое число X ($0 \le X \le 10^9$) — количество болельщиков, которые со всех сторон окружены другими болельщиками.

Гарантируется, что трибуна площади S квадратных метров с X болельщиками внутри существует.

Формат вывода
Выведите одно целое число — количество болельщиков, сидящих по периметру трибуны.

Система оценивания
Решения, работающие для случаев, где трибуна имеет форму квадрата, будут набирать не менее 4-х баллов.

Пример 1

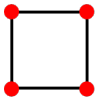
Ввод	Вывод
1	4
0	

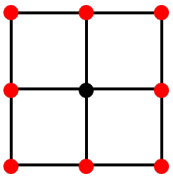
Пример 2

Ввод	Вывод
4	8
1	

Примечания

Трибуны для примеров из условия показаны на рисунке.





Задача 2. Идеальная команда

макс. 10 баллов

Ограничение времени	1 секунда
Ограничение памяти	256 Мб
Ввод	стандартный ввод
Вывод	стандартный вывод

В одной известной компании работают N сотрудников, пронумерованных от 1 до N.

Новоиспечённый глава HR-отдела Феофан предполагает, что единственный фактор, который влияет на то, смогут ли сотрудники сработаться друг с другом, — это их номера. А именно: Феофан считает, что сотрудники с номерами a и b смогут сработаться друг с другом, только если НОД(a, b) = 1, то есть если наибольший общий делитель их номеров равен единице.

Теперь Феофан хочет собрать идеальную команду из сотрудников так, чтобы любые два сотрудника в ней сработались. Помогите ему и найдите максимально возможный размер идеальной команды.

Формат ввода
Ввод содержит целое число N ($1 \le N \le 10^7$) — количество сотрудников в компании.

Формат вывода
Выведите одно целое число — максимально возможный размер идеальной команды.

Система оценивания
Решения, корректно работающие для $N \le 15$, будут набирать не менее 3-х баллов.

Пример 1

Ввод	Вывод
3	3

Пример 2

Ввод	Вывод
4	3

Примечания
В первом примере N = 3: можно взять всех трёх сотрудников, так как НОД(1, 2) = НОД(1, 3) = НОД(2, 3) = 1.

Во втором примере N = 4: можно взять сотрудников под номерами 1, 3, 4.

Задача 3. Битовые любимцы

макс. 10 баллов

Ограничение времени	1 секунда
Ограничение памяти	256 Мб
Ввод	стандартный ввод
Вывод	стандартный вывод

У Феофана есть два битовых любимца — числа A и B.

Когда Феофан видит некоторое число x, он сразу вычисляет в уме числа A AND x и B OR x, где AND обозначает побитовое И, OR обозначает побитовое ИЛИ.

Феофан радуется, если два полученных числа совпадают. Определите, сколько есть разных значений x от 0 до $(2^{32} - 1)$ включительно, после проведения вычислений, с которыми Феофан будет радоваться.

Формат ввода
Первая строка содержит целое число T ($1 \le T \le 10^4$) — количество тестовых случаев.

Следующие T строк описывают тестовые случаи. Каждая из них содержит целые числа A_i и B_i ($0 \le A_i, Bi \le 2^{32} - 1$) — битовые любимцы Феофана.

Формат вывода
Для каждого тестового случая выведите одно целое число — количество значений x, для которых A_i AND x и B_i OR x совпадают.

Система оценивания
Решения, корректно работающие для $A_i, B_i \le 10^4$, будут набирать не менее 5-ти баллов.

Пример

Ввод	Вывод
2 0 0 1 0	1 2

Примечания
В первом тестовом случае примера A AND x = 0 и B OR x = x, поэтому равенство выполняется только при x = 0.

Во втором тестовом случае примера A AND x=0 или 1 в зависимости от чётности x. B OR x = x, поэтому подходят x = 0 и x = 1.

Блок «Данные»

Задача 1. Вековые колебания

макс. 10 баллов

В банке «Гоблин и Со» есть множество секретных пещер в недрах горы для хранения богатств избранных клиентов. Разберитесь в структуре информации в банке и автоматизируйте получение данных.
База данных банка «Гоблин и Со» состоит из нескольких таблиц.

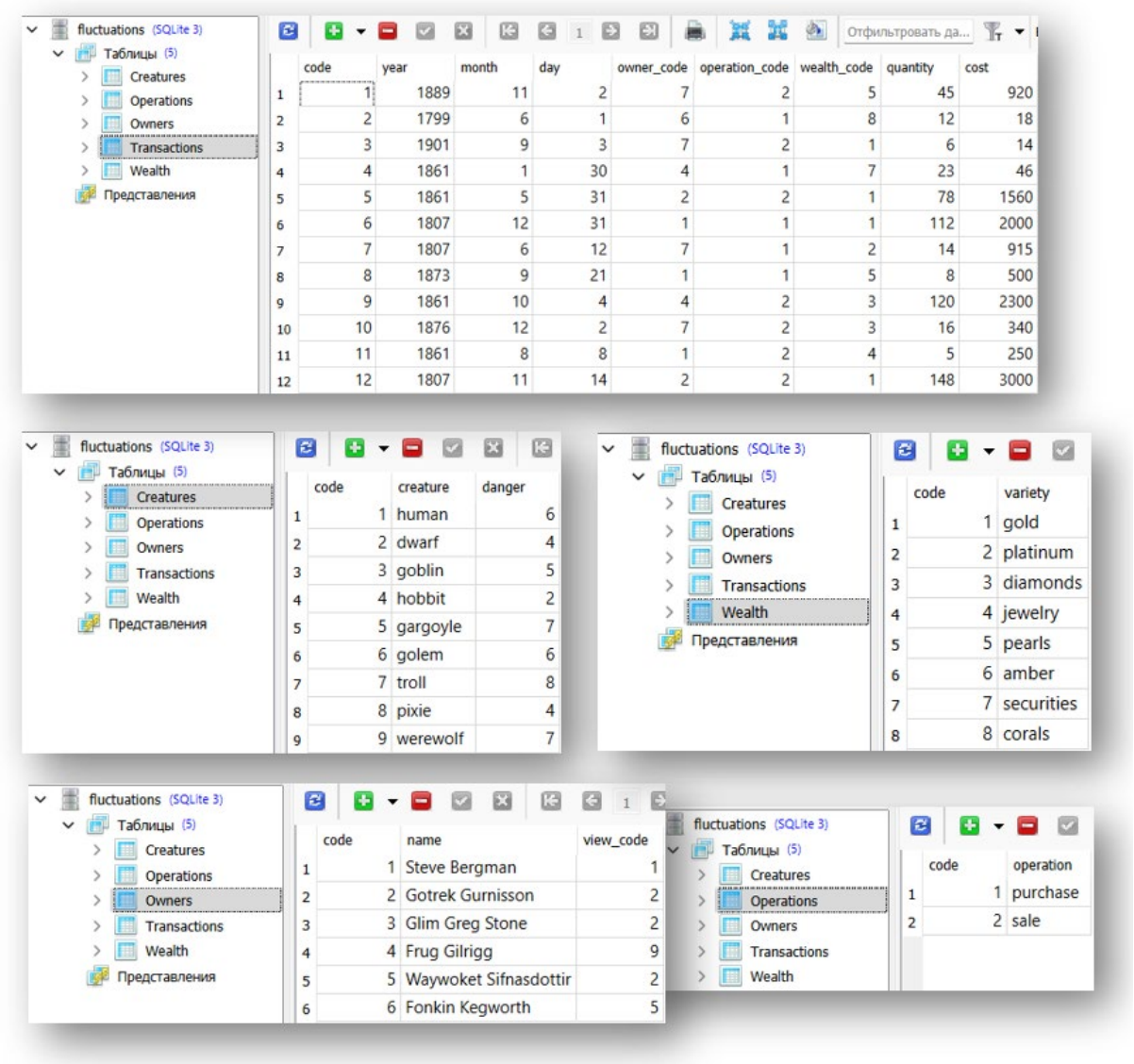
Transactions (Транзакции)
код, год, месяц, день, код_владельца, код операции, код богатства, количество, стоимость этого количества
code, year, month, day, owner_code, operation_code, wealth_code, quantity, cost

Owners (Владельцы)
код, имя, код_вида
code, name, view_code

Wealth (Виды богатства)
код, разновидность
code, variety

Creatures (Виды существ)
код, существо, опасность
code, creature, danger

Operations (Виды операций)
код, операция
code, operation



По этой БД можно узнать, как колебалась цена на золото.

Напишите SQL-запрос, который определит наибольшую цену, по которой была продана (sale) гномами (dwarf) каждая единица золота (gold) в каждый из годов, представленных в БД, в период с 1800 по 1900 год включительно.
Результаты — год, цена (округлена до сотых) — должны быть отсортированы по годам по возрастанию.

Ваше решение должно содержать **только скрипт**, написанный на языке SQL (диалект SQLite), работающий с базой данных, аналогичной по структуре fluctuations.db.

Ваш запрос не должен изменять существующую базу данных.

Пример базы данных fluctuations.db для отладки решения можно скачать здесь.
Ваш запрос должен вернуть данные по годам и ценам, названия возвращаемых полей могут быть любыми, важно, чтобы их было два: год и цена.

Примечания

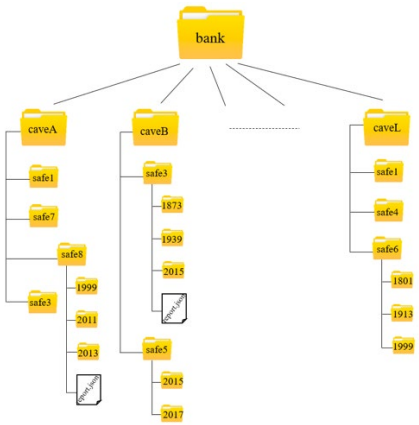
Ваш запрос будет запущен на различных тестовых данных. Для базы данных из примера для отладки выводимый результат должен быть таким:

Year	Price
1807	20.27
1861	20

Задача 2. Гоблинский банк

макс. 10 баллов

В банке «Гоблин и Со» есть множество секретных пещер в недрах горы для хранения богатств избранных клиентов. В бухгалтерии на каждую такую пещеру заведена своя папочка. А в каждой папочке — сейфы с файлами с идентификационными данными и папки по транзакциям каждого года хранения богатств.



На рисунке показана иерархическая структура хранения данных в банке (папка верхнего уровня). Для каждой пещеры есть своя папка (второй уровень), а в каждой пещере находится несколько сейфов, информация о владельце каждого из которых записана в файле report.json (словарь с ключами: id, owner, who is Mr, benefits, address, contact (id, владелец, кто он, льготы, адрес, контакт для связи)) и папки с транзакциями по годам в виде набора csv-файлов, в названии которых записана дата транзакций в формате *trans<YYYY-MM-DD>.csv* с заголовками (разделители — запятые):

id, expense, wealth, amount, outcome
id, купля/продажа, вид богатства, количество, сумма

Может оказаться, что сейф меняет владельца. Тогда файл json удаляется, а папки с транзакциями по годам могут ещё оставаться некоторое время до перезаписи под нового владельца.

Вам доступен архив bank.zip со всеми папками и файлами данных. Определите, сколько (суммарное значение по полю amount) золота в любом виде (gold присутствует в названии вида богатства) положили в банк (транзакция purchase) гномы (dwarf по ключу who is Mr в json-файле) в период с 1900 по 2020 год включительно.

В ответ запишите **только число** — количество золота. Решение прикладывать не нужно.

Пример
Для архива bank_example.zip ответ должен быть 87.

Задача 3. Уплата налогов

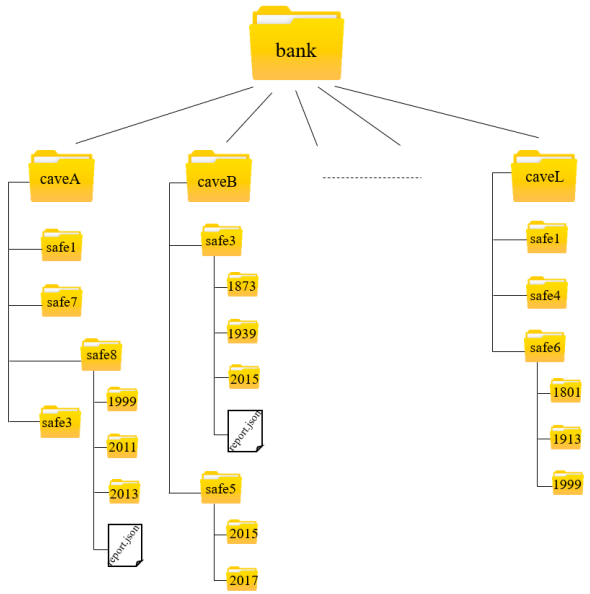
макс. 10 баллов

Ограничение времени	1 секунда
Ограничение памяти	64 Мб
Ввод	стандартный ввод или input.txt
Вывод	стандартный вывод или output.txt

Банк «Гоблин и Со» оказывает услуги по хранению богатств в хорошо укреплённых и охраняемых драконами пещерах внутри горы.

Банк предлагает новую услугу: уплату налогов за клиентов. Для этого ему нужно уметь рассчитывать их персональный налог на прибыль в течение этого года, то есть на суммарное значение по полю outcome для проданных (sale) в этом году богатств.

Процентная ставка налога может быть снижена, если у владельца есть льготы (поле benefits в json-файле). Льгота вычитается из ставки налога, в результате налог может быть снижен, но не до нуля, минимальная остающаяся ставка — 5%.



В папке верхнего уровня находятся каталоги с сейфами по каждой пещере; в папках каждого сейфа находятся папки транзакций владельца по годам: имя каталога — год; также может быть файл json, если владелец актуальный и это его транзакции хранятся в сейфе, тогда файл report.json содержит словарь с ключами:
id, owner, who is Mr, benefits, address, contact
id, владелец, кто он, льготы, адрес, контакт для связи

В папках по годам находятся csv-файлы с транзакциями в текущий день, имена файлов записаны в формате trans<YYYY-MM-DD>.csv с заголовками (разделители — запятые):
id, expense, wealth, amount, outcome
id, купля/продажа, вид богатства, количество, сумма

Напишите для банка такую программу.

Базу данных из примера можно скачать здесь: bank0.zip

Формат ввода
Вводятся:
имя файла архива с каталогами, имя владельца, год, процент налога на прибыль в этом году.

Гарантируется, что такой владелец сейфа и такой год в сейфе существуют, но один владелец может иметь несколько сейфов в одной или нескольких пещерах. У одного и того же владельца одинаковые json-файлы с данными.

Формат вывода
Для указанного владельца для введённого года нужно найти персональный налог: из введённого значения налога в этом году вычесть его личную льготу, но снижение не может быть ниже 5%. Найти абсолютное значение налога на прибыль: суммарный доход (сумму значений по полю outcome для всех sale транзакций этого года) умножить на персональное значение налога, разделить на 100 и отбросить дробную часть (то есть округлить вниз).

Вывести полученный результат.

Пример

Ввод	Вывод
bank0.zip Amergin Clornhrim 1918 20	20

Примечания
Если при обходе архива вы будете использовать временные каталоги, не забудьте их удалить после работы.

Блок «Бэкенд»

Задача 1. Коллоквиум

макс. 10 баллов

Ограничение времени	10 секунд
Ограничение памяти	159.3671875 Мб

Маги очень заняты повседневными делами, у них всегда полно заказов: кому-то нужно недоброжелателя превратить в жабу, кому-то — поменять водопровод, не разрушая дом, кому-то — превратить порцию свинца в золото. Но и спасать мир тоже нужно, а для этого нужна сила всех магов!

Напишите сервис, отвечающий на порту 8080, который определит ближайшую к указанной дату и вернёт её в формате dd.mm.yyyy, когда все маги свободны в течение всего дня (с 0 часов до 23:59 по Гринвичу) и могут собраться вместе, чтобы спасти мир. Если таких дат несколько, надо найти самую позднюю.

В файле input.txt в формате JSON-Lines в каждой строке указана дата события в поле "date" в формате dd.mm.yyyy, в поле "time" время в формате hh:mm±hh:mm, в поле "duration" продолжительность в минутах, "name" содержит название события и "participant" имя участника, кому принадлежит это событие.

В запросе /find передаётся параметр: date — дата, ближе всего к которой нужно найти свободный для всех день.

В файле не более 100 000 строчек.

Пример запроса: http://127.0.0.1:8080/find?date=02.01.2024

Примечания
Пример input.txt

{"date": "01.01.2024", "time": "03:30+03:00", "duration": 120, "participant": "Merlin", "name": "Round table"}
{"date": "02.01.2024", "time": "05:15+04:00", "duration": 120, "participant": "Merlin", "name": "1x1 Merlin-Artur"}
{"date": "03.01.2024", "time": "17:00+03:00", "duration": 120, "participant": "Harry Potter", "name": "Quidditch"}

Пример запросов для предоставленного input.txt

http://127.0.0.1:8080/find?date=31.12.2023

31.12.2023
http://127.0.0.1:8080/find?date=01.01.2024
31.12.2023
http://127.0.0.1:8080/find?date=02.01.2024
04.01.2024
http://127.0.0.1:8080/find?date=03.01.2024
04.01.2024
http://127.0.0.1:8080/find?date=04.01.2024
04.01.2024

Для языка **Python** доступны библиотеки:

1. Flask
2. aiohttp
3. FastAPI

Для языка **Java** доступны библиотеки:

1. json-simple
2. jackson
3. async-http-client

Для языка **C++** доступна библиотека **httplib.h**

Шаблоны оформления задач можно скачать по ссылке.

Задача 2. Молчащая мандрагора

макс. 10 баллов

Ограничение времени 15 секунд
Ограничение памяти 159.3671875 Мб

Мандрагоры — очень капризные магические растения. Их нельзя переливать, но и пересушивание они выносят недолго. А если что, начинают кричать так, что мало не покажется.

Напишите сервис, отвечающий на порту 8080, который будет составлять расписание полива мандрагор.

Необходимо реализовать выполнение запросов:

http://127.0.0.1:8080/add/<plant:str>/<interval:int>/<maxdelay:int>/?date=01.01.2024 — добавить мандрагору (строковый идентификатор растения, интервал полива в днях, допустимое количество дней пересушивания, дата добавления).

http://127.0.0.1:8080/task/?date=01.01.2024 — вернуть список растений (идентификаторы растений в лексикографическом порядке через запятую без пробелов), которые нужно полить в указанную дату с учётом пропущенных поливов, если растение ещё живое.

http://127.0.0.1:8080/watering/<plant>/?date=01.01.2024 — полив растения в указанную дату.

Так как дата передаётся в каждый запрос текущая, то она никогда не убывает.

Гарантируется, что в системе менее 100 000 растений, а интервал не превышает 5 недель.

Пример:

http://127.0.0.1:8080/add/1/2/1/?date=01.01.2024
http://127.0.0.1:8080/add/2/2/2/?date=01.01.2024
http://127.0.0.1:8080/task/?date=01.01.2024
http://127.0.0.1:8080/task/?date=02.01.2024
http://127.0.0.1:8080/task/?date=03.01.2024
1,2
http://127.0.0.1:8080/task/?date=04.01.2024
1,2
http://127.0.0.1:8080/task/?date=05.01.2024
2
http://127.0.0.1:8080/watering/2/?date=05.01.2024
http://127.0.0.1:8080/task/?date=07.01.2024
2

http://127.0.0.1:8080/watering/2/?date=07.01.2024
http://127.0.0.1:8080/task/?date=08.01.2024
http://127.0.0.1:8080/watering/2/?date=08.01.2024
http://127.0.0.1:8080/task/?date=08.01.2024
http://127.0.0.1:8080/task/?date=09.01.2024
http://127.0.0.1:8080/task/?date=10.01.2024

Примечания

Для языка **Python** доступны библиотеки:

1. Flask
2. aiohttp
3. FastAPI

Для языка **Java** доступны библиотеки:

1. json-simple
2. jackson
3. async-http-client

Для языка **C++** доступна библиотека **httplib.h**.

Шаблоны оформления задач можно скачать по ссылке.

Блок «Фронтенд»

Задача 1. Треугольная жизнь

макс. 10 баллов

Игра «Жизнь», также известная как «Жизнь Конвея», — это клеточный автомат, придуманный британским математиком Джоном Конвеем в 1970 году. Это игра с нулевым числом игроков, то есть её эволюция определяется начальным состоянием и не требует дальнейшего ввода данных.

Популярность игры «Жизнь» и даже культовый статус в 1970-х годах и позже были обусловлены её появлением в то же время, когда компьютеры становились всё более доступными. Она является полной по Тьюрингу, то есть теоретически столь же мощной, как и любой компьютер с неограниченной памятью, и без ограничений по времени.

Эта задача основана на игре «Жизнь», но имеет некоторые отличия в правилах и визуализации — вам необходимо реализовать такую симуляцию.

Данная форма жизни состоит из треугольных клеток зелёного цвета (#00a83d).

Закономерности поведения клеток:

- У каждой ячейки есть 12 соседей (3 по сторонам и ещё по 3 на каждый угол).
- Если вокруг пустой ячейки есть 4 или 8 соседей, то на следующем шаге на этом месте рождается новая клетка.
- Если вокруг любой живой клетки есть 4, 5 или 6 соседей, то на следующем шаге она выживает.
- Во всех остальных случаях (когда соседей 3 и меньше или 7, 9 и больше) клетки умирают от «одиночества» или «перенаселения» и ячейка становится пустой.

Симуляция происходит на поле из треугольных ячеек шириной 28 пикселей и высотой 24 пикселя («почти равносторонних»). Пустые ячейки раскрашены в «шахматном» порядке в цвета #d8dbe0 и #f7f8fa. См. изображение:

Поле «закольцовано» по всем сторонам — т. е., не смотря на фиксированные размеры в количестве ячеек, самые крайние ячейки являются соседями. Если что-то двигается

за правый край поля, то оно появляется слева (и наоборот). Аналогично, если что-то двигается за нижний край, то оно появляется сверху (и наоборот).

Т. к. треугольники ориентированы вершинами вверх или вниз, то левый и правый края поля «разрезают» соответствующие ячейки пополам. С точки зрения «закольцованности» поля эти половинки нужно трактовать как две половинки одной ячейки. Соответственно, поле всегда имеет чётное количество ячеек в ширину. Так же можно учитывать, что поле не может быть меньше, чем 6 x 6.

См. изображение, на котором живыми являются клетки в первой колонке ячеек:

Формат ввода

Начальная конфигурация поля задаётся в window.data.field в виде массива массивов одинаковой длины, где значениями являются числа: 0 (пустая клетка) и 1 (живая клетка).

Например:

- поле размером 2 x 2 со всеми пустыми клетками

```
window.data = {
  field: [
    [0, 0],
    [0, 0]
  ]
}
```

- поле размером 3 x 3 с живой клеткой посредине

```
window.data = {
  field: [
    [0, 0, 0],
    [0, 1, 0],
    [0, 0, 0]
  ]
}
```

Кроме этого, в части тестов определено числовое поле window.data.steps. Оно указывает на количество шагов симуляции, которые нужно сделать сразу, до вызова функции готовности решения (window.onSolutionReady()). Если количество шагов симуляции не определено, то нужно просто отрисовать начальное состояние.

Часть тестов проверяет эффективность решения на больших объёмах данных.

Примечания

Решение должно представлять из себя один HTML-файл (все нужные стили должны быть включены внутрь).

HTML-элемент с полем симуляции должен иметь класс Field.

В конце работы вашего JS-кода нужно вызвать глобальную функцию window.onSolutionReady(), которая декларируется автоматически в тестовом окружении (не нужно её переопределять).

Рекомендуется использовать заготовку в архиве. В качестве решения можно отправить доработанный файл stub.html из данного архива.