GAMINID. Тестовое задание **(Web-**разработчик)

1 Алгоритмы (используйте **C, Ruby, Python** или **AS/JS)**

(1) Напишите программу печати однонаправленного списка в обратном порядке. Какова ее алгоритмическая сложность по времени и памяти?

(2) Напишите функцию, которая из произвольного входящего массива выберет все комбинации чисел, сумма которых будет равняться 10.

2 Front-end

(1) Дан отрывок JS кода:

```
var a = {a: { a1 : 1 }, b: "string" }, c = a, b = { a : a.a, b: a.b };
delete a.a;
```

Что будут содержать `b.a` и `c.a`? Объясните результат.

(2) Реализуйте компонент постраничного листания внутри веб-страницы на HTML/JS. Поведение полностью сдублируйте из нашей игры: http://vk.com/RingKings (закладка Соревнования → Все лиги). Там во всех лигах разное количество страниц, так что можно посмотреть на поведение в различных ситуациях. Внешний вид повторять не нужно: сделайте на свой вкус. Реализовать нужно только компонент листания страниц, сами страницы достаточно схематично показать номером, просто чтобы было видно, на какую мы переключились.

3 Back-end: Erlang ↔ DB

Выберите одну или несколько известных вам NoSQL баз данных (MongoDB, Redis, RethinkDB) и выполните следующие задания:

(1) Найдите способ подключиться из Erlang-программы к базе данных (на локальном хосте без авторизации). Выполните тестовые запросы — сохранение и чтение одного/нескольких документов, изменение отдельных полей внутри документа.

(2) Реализуйте максимально эффективный (в плане нагрузки на сервер базы данных) проход по таблице: перебор всех записей небольшими фиксированными блоками (записей в таблице может быть много миллионов, поэтому нужно, чтобы клиент единовременно держал в памяти только текущий блок — несколько десятков или сотен записей, и по завершению работы с ним, обращался за следующим).

(3) Часто бывает необходимо пройтись по всей таблице, запросив через API стороннего сервера некоторые данные по каждой записи. При этом количество запросов к API в секунду ограничено, а один поток для запросов использовать недостаточно из-за долгого ответа внешнего сервера (скажем, ограничение - не более 10 запросов в секунду, а среднее время ответа - около 250 мс; таким образом, если каждый раз слать запрос, ждать ответ, слать следующий, мы не сможем отправить более 4 запросов в секунду). Поэтому для того, чтобы полностью использовать всю отведенную емкость, нужно завести небольшое количество потоков, достаточное, чтобы заполнить канал, и слать запросы параллельно, но все же не превышая заданный лимит. Используя наработки из пп. 1-2, предложите общее решение данной задачи: вытаскивая из таблицы по 100 записей, отсылать по 5 записей за раз, делая не более 10 запросов в секунду (сам запрос достаточно эмулировать простым “засыпанием” процесса на нужное количество миллисекунд). Все числа (100, 5, 10) даны просто для примера, их необходимо вынести в константы.