- Задача G

  **Формаольное условие:**

  Приходят числа. Каждое можно либо пропустить, либо положить в стек. Также можно вытаскивать числа из стека. Необходимо выпустить все числа в порядке возрастания.

  **Решение:**

  Очевидно, что если очередное число минимальное то его нужно пропустить дальше. Не менее очевидно, что если минимальное число лежит на вершине стека то нужно его оттуда вытащить. Если ни одно условие не выполняется, то у нас нет выбора, кроме как положить число в стек. Осталось быстро находить минимальное число из оставшихся. Это легко сделать мультисетом.

- Задача H

  **Наблюдение:**

  После $(1000 + 999 + ... + 1) = 500500$ секунд рыбки в любом аквариуме будут рождаться каждую секунду $=>$ ответ не привосходит 500501.

  **Решение:**

  Будем поддерживать сет пар `(момент_когда_родится_очередная_рыбка, номер_аквариума)`, номер текущего аквариума, у которого мы стоим и текущий момент времени. Будем брать первый элемент сета и проверять, успеем ли мы до него дойти. Если нет – ответ найден, если да – поменяем момент времени и позицию, удалим первый элемент из сета и добавим новый. Итераций будет не больше, чем максимальный ответ, поэтому времени хватит.

- Задача I

  Типичная задача на rope.

  **Решение:**

  Заведем rope. Будем вырезать отрезок и вставлять в начало. Всё.

- Задача J

  **Формаольное условие:**

  Нужно добавлять элементы по коду, удалять по коду, менять занчение надежности и искать $n$–ный элемент по паре `(надежность, id)`.

  **Решение:** Заведем map (обычный или unordered) из кода в пару `(reliability, id)`. Тогда первые 3 вида запросов очевидны. Чтобы делать последний запрос используем tree из pbds. При запросах добавления/удаления будем добавлять/удалять элениент. При изменении – удалять старый и добавлять новый. Для последнего запроса будем делать `find_by_orderer`. Всё.