

1 Стек (LIFO - last in first out)

Стек — структура данных, поддерживающая следующие методы:

- `push(x)` — добавляет элемент `x` в конец структуры
- `pop()` — удаляет последний добавленный элемент из структуры, иногда возвращает его значение
- `last()` — возвращает значение последнего добавленного элемента
- `empty()` — проверяет структуру на пустоту
- `size()` — возвращает число элементов в структуре

2 Очередь (FIFO - first in first out)

Очередь — структура данных, поддерживающая следующие методы:

- `push(x)` — добавляет элемент `x` в конец структуры
- `pop()` — удаляет элемент, который был добавлен раньше всех из структуры, иногда возвращает его значение
- `last()` — возвращает значение элемента, который был добавлен раньше всех
- `empty()` — проверяет структуру на пустоту
- `size()` — возвращает число элементов в структуре

3 Дек

Дек — структура данных, позволяющая добавлять элементы как в начало, так и в конец, и удалять элементы как из начала, так и из конца.

4 Динамика: ближайший меньше слева (справа)

Пусть, для каждого элемента массива, нам нужно найти какой элемент слева меньше.

Будем добавлять элементы в стек слева-направо, перед добавлением очередного элемента выталкиваем(удаляем) все элементы большие его. Оставшийся элемент и есть ответ для нового. Полезно в стеке хранить не значения элементов, а их индексы.

5 Минимум на очереди

Пусть нужна структура, которая умеет:

- добавлять элемент в очередь
- удалять элемент из очереди
- запрашивать минимум в очереди

Понятно, как решить эту задачу если бы вместо очереди был стек. (Либо храним в стеке пару значений (значение элемента и минимум), либо храним два стека).

Можно свести очередь к двум стекам: добавлять в стек₊, а удалять из стек₋.

С добавлением всё понятно, что делать, когда нужно удалить элемент, а стек₋ пуст? Нужно взять все элементы из стек₊ и переложить в стек₋ т.е. : стек₊.push(стек₋.pop())