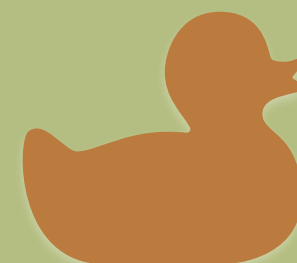
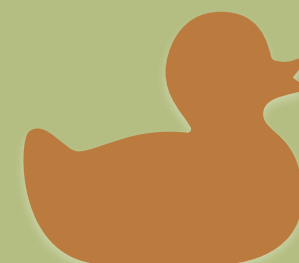
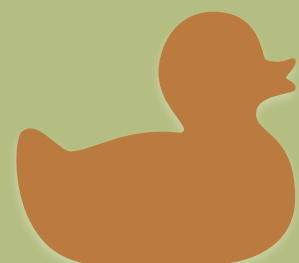


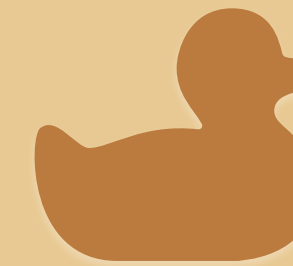
СТЕК, ДЕК, ОЧЕРЕДЬ, ДИНАМИЧЕСКИЙ МАССИВ, ОДНОСВЯЗНЫЙ И ДВУСВЯЗНЫЙ СПИСОК, ОЧЕРЕДЬ С ПРИОРИТЕТОМ

СДЕЛАНО ПРИ ПОДДЕРЖКЕ КАПРИЗНОЙ УТОЧКИ



ДИНАМИЧЕСКИЙ МАССИВ

`std::vector`



Хочу массив на 5, нет
7, нет, на 8 элементов!

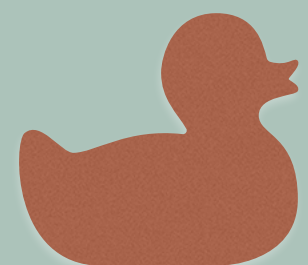
```
int main()
{
    int size;
    std::cin >> size;
    int *array = new int[size];
    delete [] array;
    return 0;
}
```

- `vector <int> vector_first(3);` заполнит нулями
- `vector <int> vector_second;`
`vector_second.reserve(3);` ничем не заполнит -> там мусор
- `(vec_first == vec_second);` сравниваем
- `size()` и `empty()`
- `push_back()` и `pop_back()`
- `front()` и `back()`
- `insert()` // `v.insert(v.begin() + i, valueToInsert);`
- `operator[]`

ОДНОСВЯЗНЫЙ СПИСОК

```
struct Node {  
    string val;  
    Node* next;  
    Node(string _val) : val(_val),  
    next(nullptr){}  
};
```

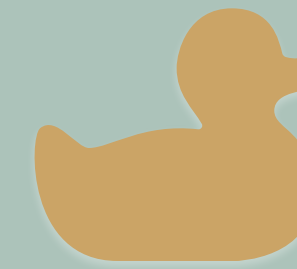
Хотим найти элемент,
который предшествует элементу x



Есть ли в односвязном
списке цикл?

std::forward_list

- push_front()
- pop_front()
- front()



А теперь хочу
вставлять элемент в начало
массива! И удалять. И
чтобы за $O(1)$

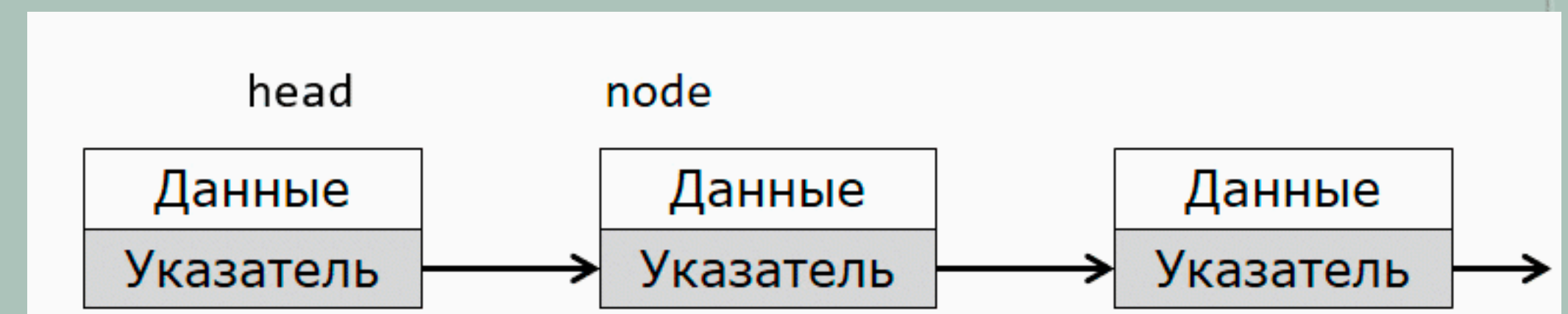
+

Вставка/удаление в начало - $O(1)$

Не зависит от наличия непрерывного
куска памяти (как для массивов)

-

Доп память для хранения указателей
Вставка узла перед определенным узлом

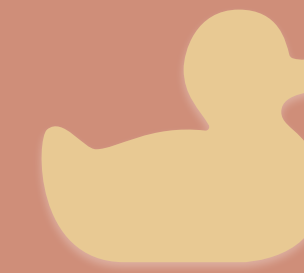


ДВУСВЯЗНЫЙ СПИСОК

```
struct Node {  
    string val;  
    Node* next;  
    Node* prev;  
    Node(string _val) : val(_val),  
        next(nullptr), prev(nullptr){}  
};
```

std::list

- push_front()
- push_back()
- pop_front()
- pop_back()
- front()
- back()



Хочу ходить в обе стороны! И еще вставлять элементы в конец массива за $O(1)$

+

Вставка/удаление в начало/конец - $O(1)$

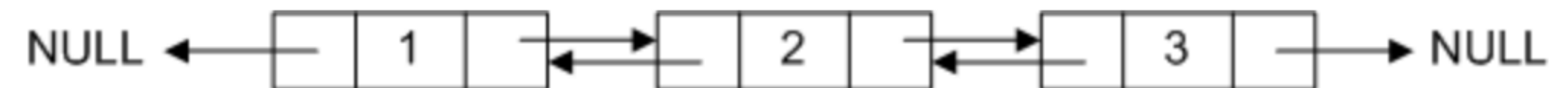
Не зависит от наличия непрерывного куска памяти (как для массивов)

-

Еще больше доп памяти для хранения указателей



Как объединить два списка?

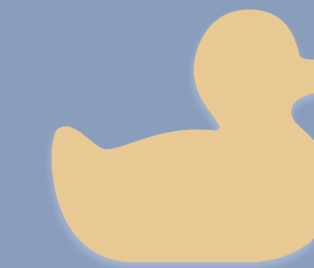


СТЕК (LIFO)

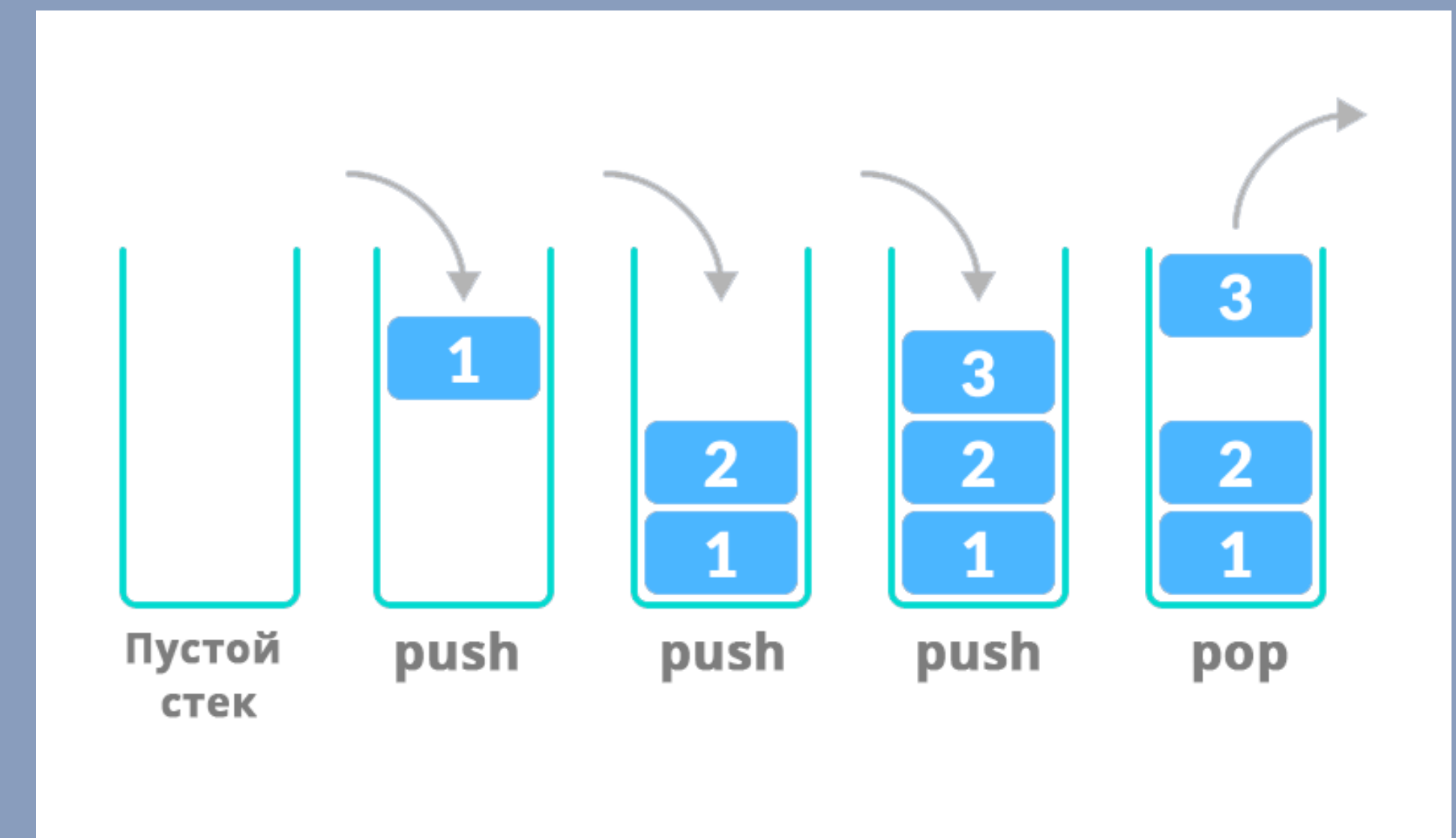
- **Массив - макс размер известен заранее**
- **Динамический массив - макс размер неизвестен**
- **Односвязный список - при добавлении элемента делаем его головой списка**
 - Задача на проверку правильности скобочной последовательности
({})(())[({})][[]]
 - Минимум в стеке - вспомогательный стек

`std::stack`

- `push()`
- `top()`
- `pop()`



Хочу структуру,
похожую на стакан!

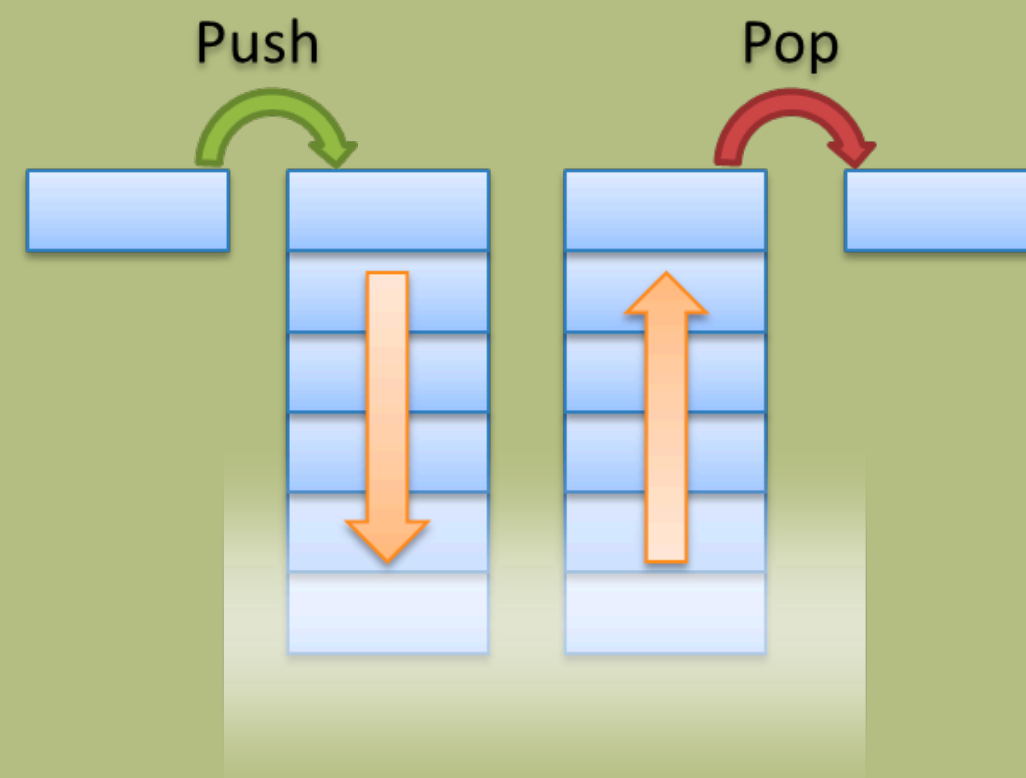


ОЧЕРЕДЬ (FIFO)

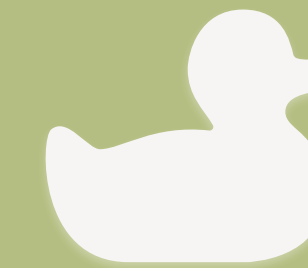
- **Массив - циклическая очередь, не более n элементов**
- **Односвязный список**
- **Два стека**

`std::queue`

- `push()`
- `pop()`
- `front()`
- `back()`



- Минимум в очереди - вспомогательная очередь



А теперь пускай как очередь!



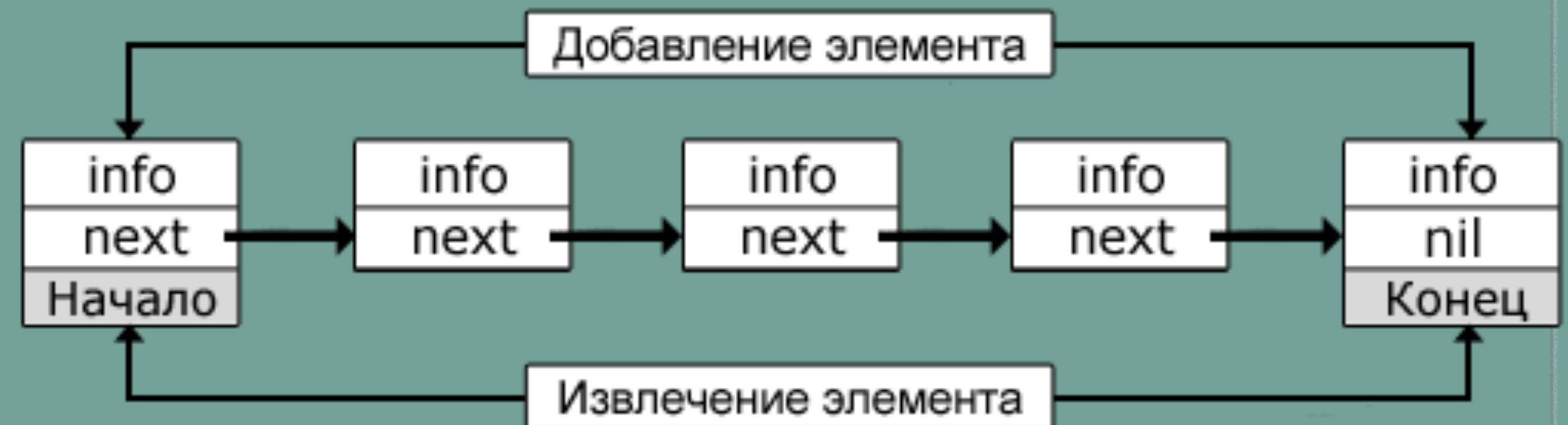
ДЕК (FIFO+LIFO)

- **Массив - циклическая очередь, не более n элементов**
- **Динамический массив**
- **Двусвязный список**
- **Два стека**

`std::deque`

- `push_front()`
- `push_back()`
- `pop_front()`
- `pop_back()`
- `front()`
- `back()`
- `operator []`

А теперь - как очередь и стек одновременно!



ОЧЕРЕДЬ С ПРИОРИТЕТОМ - ДВОИЧНАЯ КУЧА

- **Массив - циклическая очередь, не более n элементов**
- **Двусвязный список**
- **Бинарная куча**

`std::priority_queue`

- `push()`
- `pop()`
- `top()`



Ладно, что там еще
осталось?

- `sift_up()`
- `sift_down()`
- `del_max()`

