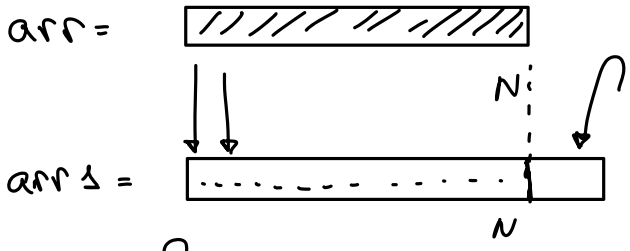
Линейные контейнеры

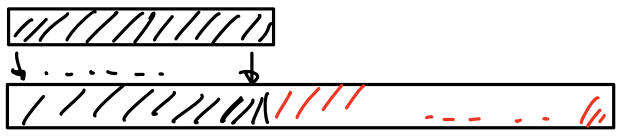
(I) Динамический расширяющийся массив.

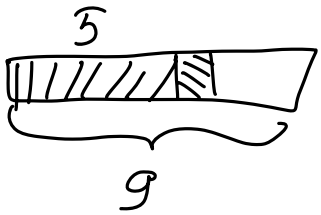


Ассимптотика: $O(N)$

```
for  i = 0... n
     arr1[i] = arr[i]
```

$\| O(N)$





class Dynamic Array

1. size – кол-во элементов в массиве
2. capacity – емкость контейнера
3. array – контейнер

метод is_empty ( ) {
    return size == 0
}

метод size ( ) {
    return size
}

Внутренний метод

метод resize ( ) {
    new_arr = [...] // размер capacity * 2
    for i = 0... capacity

O(N)

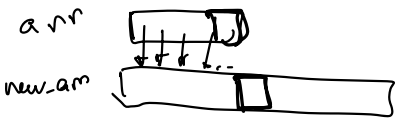
new_arr[i] = arr[i]

delete arr

arr = new_arr

capacity *= 2

}



метод push_back (elem) {

если size + 1 >= capacity

↳ resize()

arr[size] = elem

size += 1

} O(1)

O(N) } O(1)*

как часто $P = \frac{1}{N}$

O(1) в среднем.

метод pop_back ( ) {

если size = 0

↳ ошибка

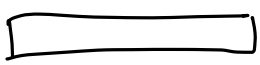
size -= 1

return arr[size] // удобно

}

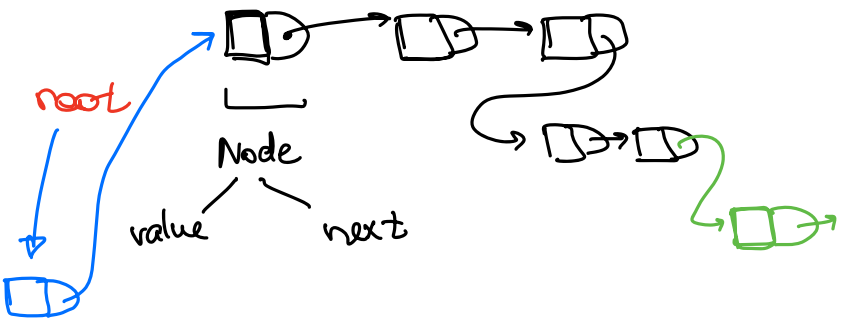
Время выполнения push_back ?

↳ O*(1)



II АТД Список → Абстрактный тип данных

(односвязный список)



push_back V

pop_back V

push_front

pop_front

class List {

- class Node
  - value
  - next
- root = Null

push_back (elem) {

- если root = null
  - ↳ root = Node (elem)
  - root.next = null
  - return
- runner = root
- while runner.next != Null
  - runner = runner.next
- new_node = Node (elem)
- new_node.next = Null
- runner.next = new_node

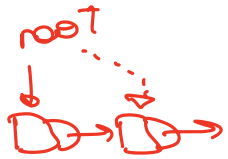
}

pop_back () {

- runner = root
- если runner = null // нет эл-в
  - ошибка
- если runner.next = null // 1 эл-т
  - delete runner
  - root = null
  - return
- while runner.next.next != null
  - runner = runner.next
- delete runner.next
- runner.next = null

}

```
push_front(elem)
{
    new_elem = Node(elem)
    new_elem.next = root
    root = new_elem
}
pop_front () {
    ecnu    root.next = null }
        delete   root
        root = null
    }
    old_root = root
    root = old_root.next
    delete   old_root
}
```



Асимптотика:

push_back pop_back	$O(N)$
pop_front push_front	$O(1)$

Dynamic Array VS List

−	+	item $O^*(1)$
+	−	обр. по индексу [дека]
+	−	элемент следующий

(+) храним разные типы данных

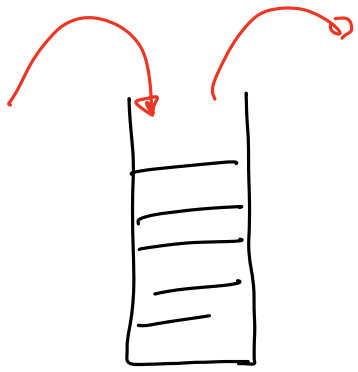
Когда integer

Свой класс → ЧМВ

(III) На подумать

1) Стек (ATД)

стопка книжек



FILO

LIFO

Реализация стека: (на списке)

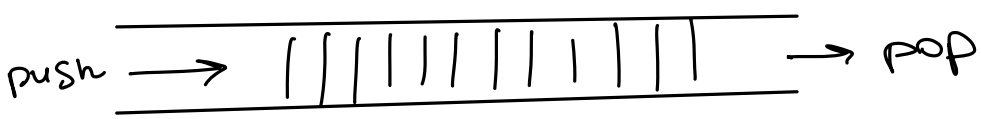
push_front O(1)

[] ... []

pop front O(1)

```
class Stack {
    || stack = List()
push(elem) {
    stack.push_front(elem)        //O(1)
}
pop() {
    stack.pop_front()             //O(1)
}
top() {
    return stack.root.value       // через конец top
}
}
```

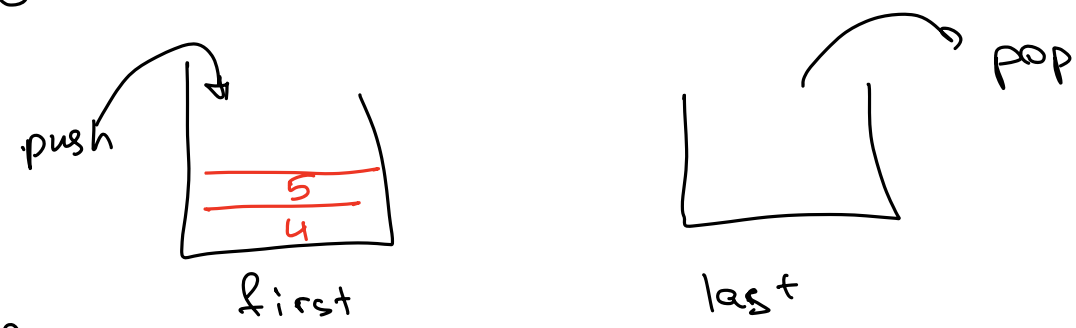
(IV) Очередь



FIFO

LILO

Очередь на 2 стеках



Queue {

stack first

stack last

push (elem) {

first.push (elem)

}

O(1)

pop ( ) {

если not last.is_empty() {

elem = last.top()

last.pop()

return elem

}

O(1)

while not first.is_empty()

last.push(first.top())

first.pop()

}

O(N)

$P = \frac{1}{N}$

если last.is_empty()

ошибка

иначе: elem = last.top()

last.pop()

return elem

}

O(1)

Асимптотика $D^*(s)$