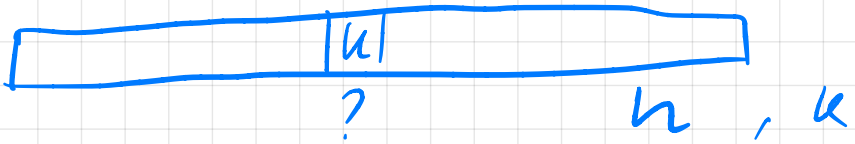
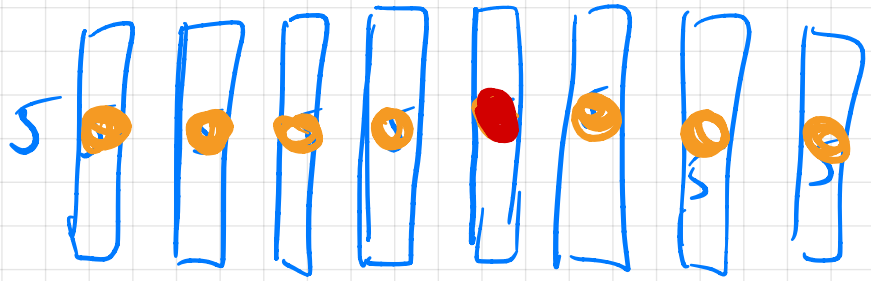
Median of Medians

k-я порядковая ст: $O(n)$





$n/5$

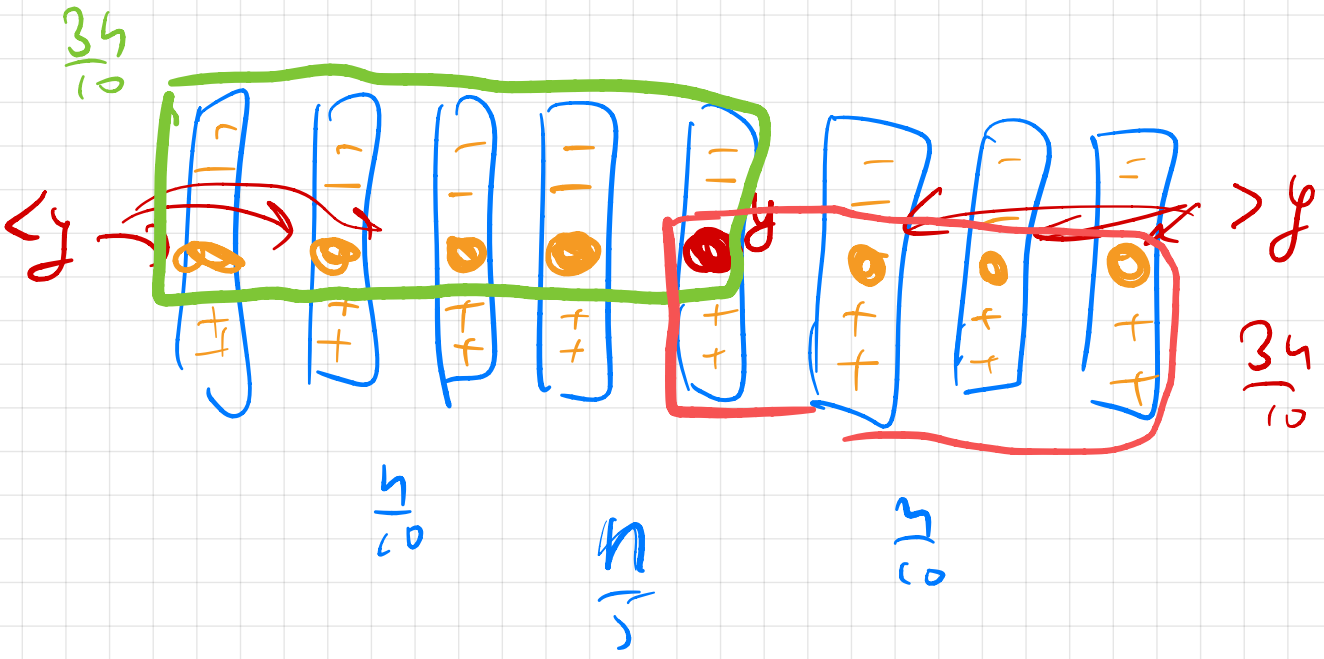
$\frac{n}{5} \cdot Sort(5) = O(n)$

$\frac{n}{5} \cdot$ Найти мед (5)

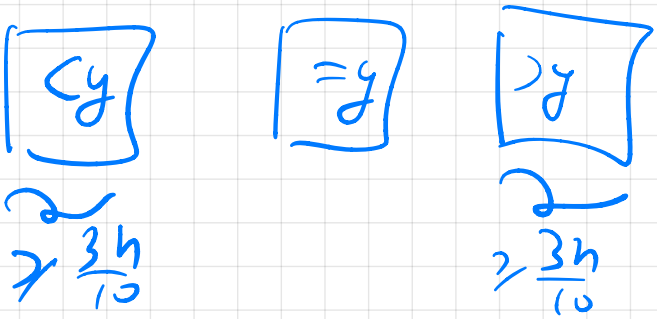
11 сравнений

● = median($n/5$ медиан)

↑ Задача поиска k-й порядк. для $k = n/10$



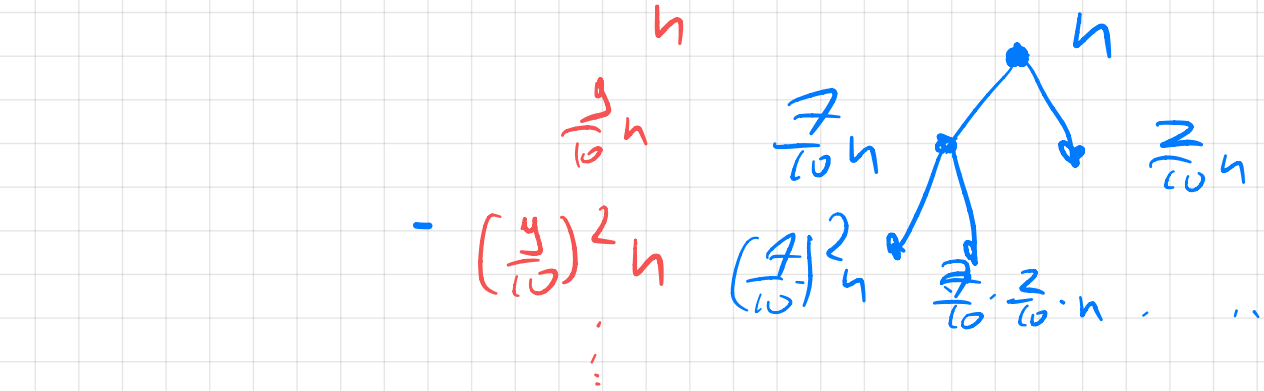
pivot()



wie noch median(,,)

$$T(n) \le \frac{11}{5}n + T\left(\frac{7n}{10}\right) + T\left(\frac{n}{5}\right)$$

z.B.: $T(n) = O(n)$



$$T(n) \le \frac{11}{5}\left(n + \frac{9}{10}n + \left(\frac{9}{10}\right)^2 n + \dots\right)$$

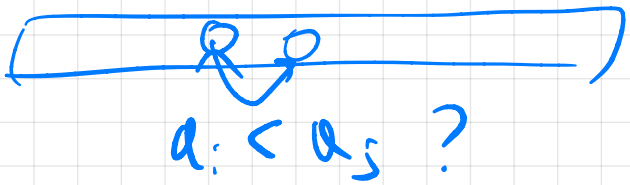
$$\frac{1}{1-\frac{9}{10}} = 10$$

$$T(n) \le 22n$$

Нижняя оценка на сортировку

n^2	$n \log n$
InsertionSort	QSort
SelectionSort	mergesort
	HeapSort

(*) Предположение: Пусть алгоритм сортировки использует только сравнения.



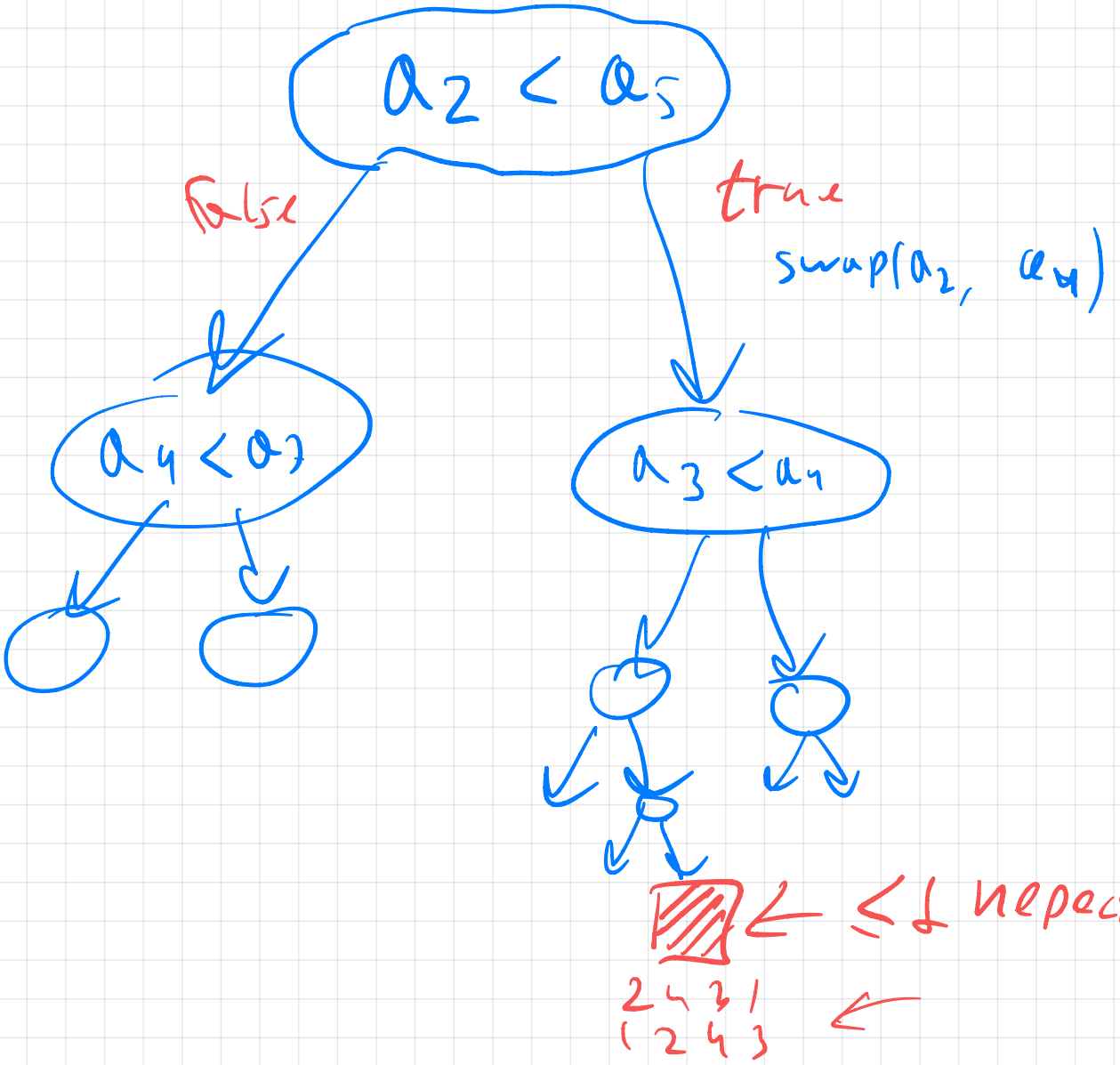
Thm: Алгоритмы удовлетв. (*) работают за $\Omega(n \log n)$

Д-во: [4, 1, 2, 3, 5, 7, 6] – перестановки.

Массивов.

$\Omega(n \log n)$

Перестановок размера n: n! штук.



← ≤ 1 перестановка

2 4 3 1
1 2 4 3 ←

Output = перестановка (input)

Если алгоритм корр., то

число листьев $\geq n!$

k = Max depth дерева.

число листьев $\leq 2^k$.

$2^k \geq$ число листьев $\geq n!$

$k \geq \log_2(n!)$

$n \cdot (n-1) \cdot \ldots$

$k \geq \sum_{t=1}^{n} \log_2(t)$

$k \geq \frac{n}{2} \log_2\left(\frac{n}{2}\right) =$

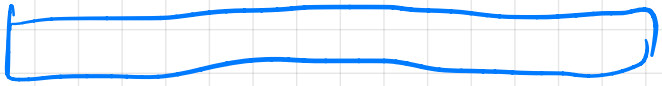
$= \frac{n}{2}(\log_2 n - \log_2 2)$

$= \Omega(n \log n)$

$n! = \sqrt{2\pi e}\left(\frac{n}{e}\right)^n \cdot (1 + O(\frac{1}{n}))$

$k = \Omega(n \log n)$

Count Sort



n эл-тов

$a_i \in [0; k) \cap \mathbb{Z}$

```
for i = 0..n-1:
    count[a_i] += 1
```

$O(n+k)$

```
ans = []
for i = 0..k-1:
    for (count[i] раз)
        ans.append(i)
```

Count Sort (2)

Пары (a_i, s_i)

a_i: $[0; k) \cap \mathbb{Z}$

s_i: "об"

Отсортировали только по a_i.

$O(n+k)$

for i=0..n-1:

list[a_i].append((a_i, s_i))

Склеить списки list[0], ..., list[k-1]

Digit/Radix Sort:

Отсортир. (a_i, b_i)

$a_i, b_i \in [0, k) \cap \mathbb{Z}$. (?)

Stability

Def: Сортировка стабильная, если она оставляет одинаковые с точки зрения сравнения эл-ты в том же порядке, что во входе.

$[1_a, 2_a, 1_b,]$

$\Downarrow$

$[1_a, 1_b, 2_a]$, не $[1_b, 1_a, 2_a]$

Heap Sort: −

Count Sort: +

Merge Sort: +

Q. Sort: +*

* – в наших версиях, обычно −

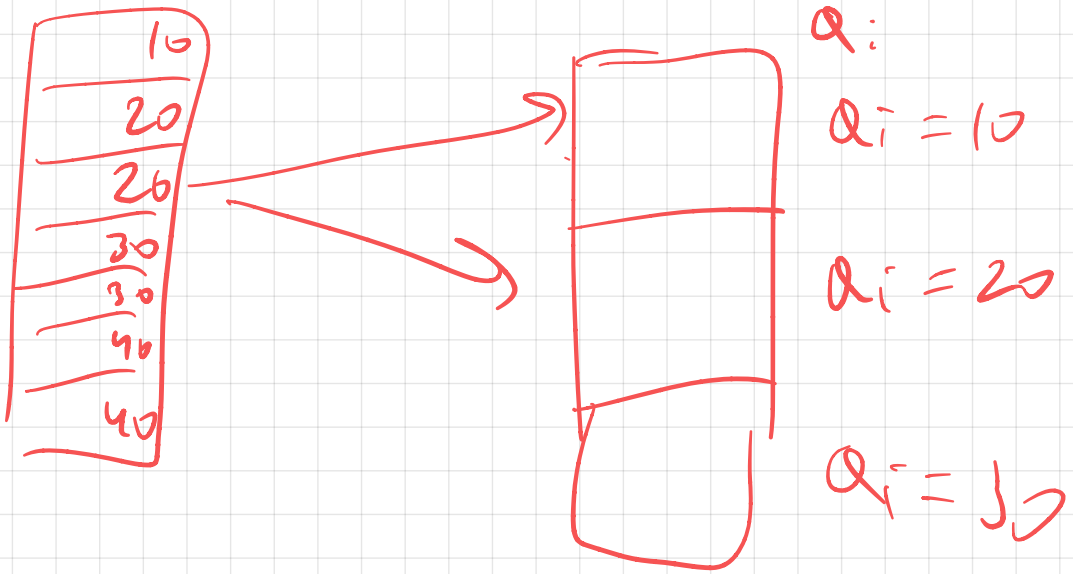
Digit / Radix Sort

Отсортир. (a_i, b_i)

$a_i, b_i \in [0, k) \cap \mathbb{Z}$. (?)

$O(n+k)$ Count-Sort$_b$ – стабильно отсорт. по b_i

$O(n+k)$ Count-Sort$_a$ – стабильно отсорт по a_i



Строки

s_0 abce | gnnn d

s_1 abcq

⋮

s_{n-1} bcdl

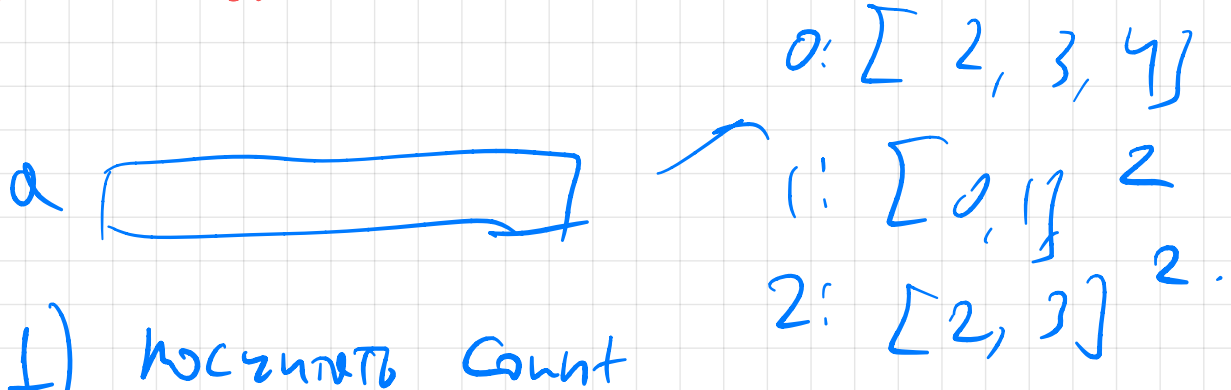
for $i = d-1 \dots 0$

отсорт по i-му разряду

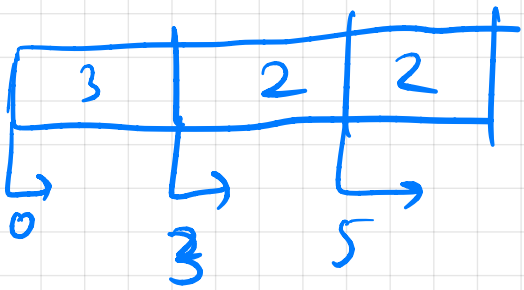
Count_Sort

$O(d(n+|\Sigma|))$.

(*): Count Sort



2)



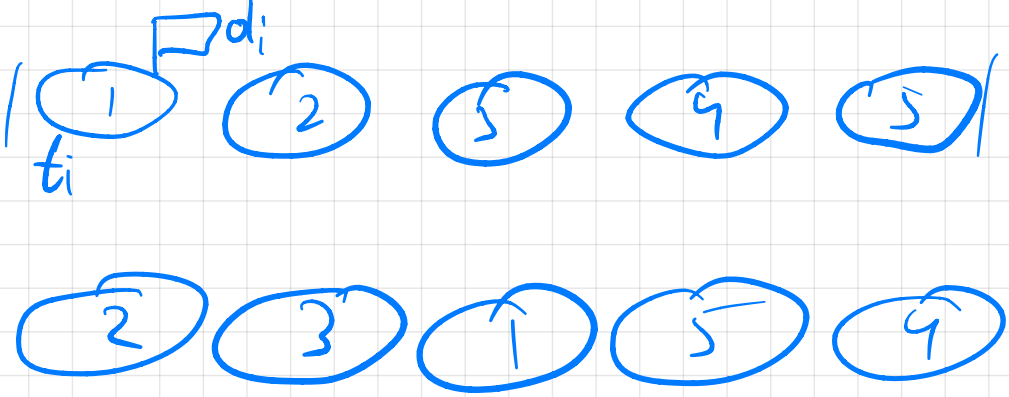
Count [3, 2, 2]:

↓

[0, 3, 5]

Жидкости

Пример задачи:

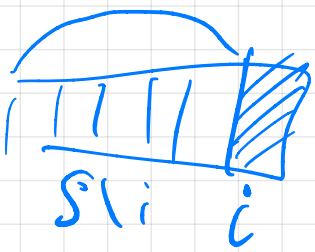


Tasks - мн-во задач

Con($S \subseteq Tasks$): bool

if $S = \emptyset$:

return true



for $i \in S$:

if Con($S \setminus \{i\}$) and $\sum_{j \in S} t_j \le d_i$:

return True

return false

$Con(\emptyset) = 1$

$Con(S) = \bigvee_{i \in S} con(S \setminus \{i\})$ and $\sum_{j \in S} t_j \le d_i$

(!) Утв: i можно выбрать как задачу с MAX d_i

Пусть $Con(S) = 1$, тогда

пусть i^* - это задача с MAX d_i

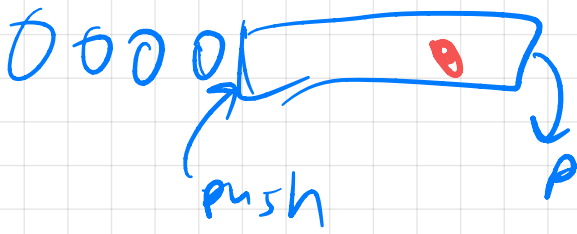
Утв: $\underbrace{(Con(S \setminus \{i^*\})}_{=1}$ and $\left[\sum_{j \in S} t_j \le d_{i^*}\right) = 1$

$Con(\emptyset) = 1$

$con(S) = Con(S \setminus \{i^*\})$ and $\sum t_j \le d_{i^*}$

Решение: Sort Задачи по d_i

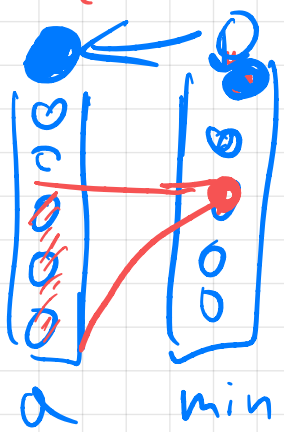
Queue (min за O(1))



O(1)

min O(1)

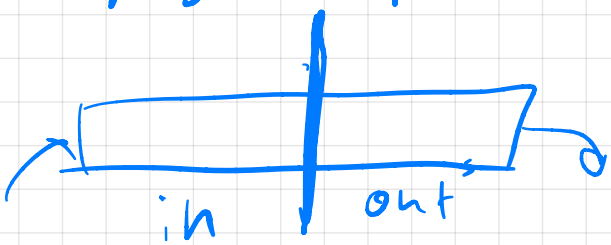
Stack (min за O(1))

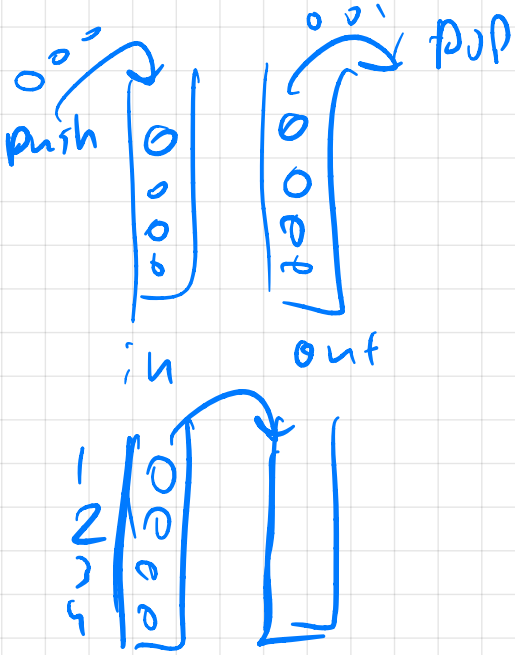


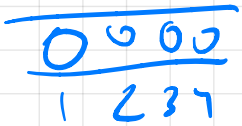
push
pop O(1)
min

```
push(x):
  st.push(x)
  st_min.push(min(x,
              self.min()))
```

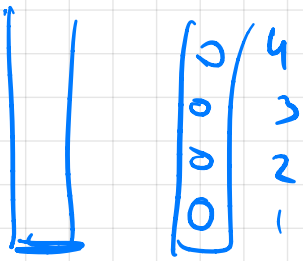
Очередь через стеки







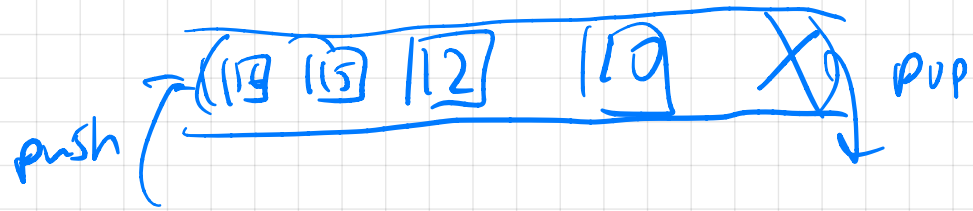
```
while !in.empty()
    out.push(in.pop())
```



push, pop, $O(1)$ в среднем.

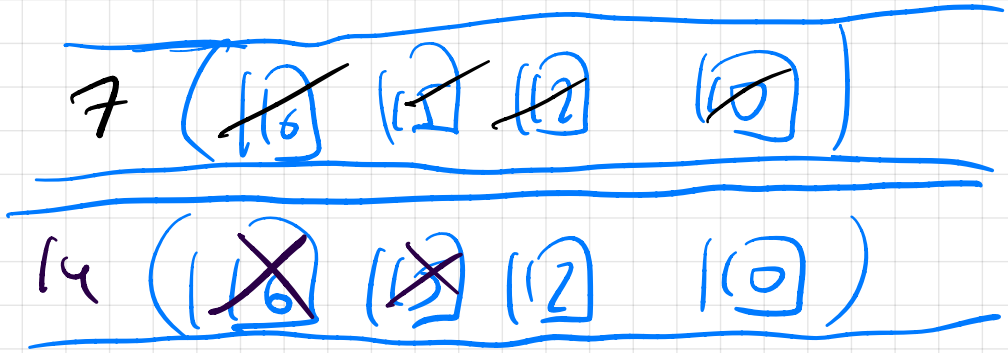
Утв! Пусть сделали k операц. (push/pop). Тогда время раб: $O(k)$.

Очередь с минимумом через дека

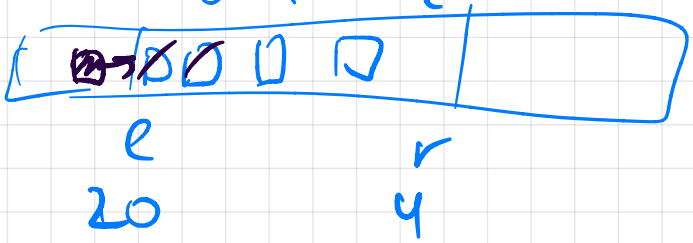


push, pop $O(1)$ в среднем.

min: $O(1)$



дека (сначала, позади).



l, r, deque ((значение, позиц.))

queue* – если хочется отвечать на pop() со значением эл-та.