Куча

Как построить кучу:

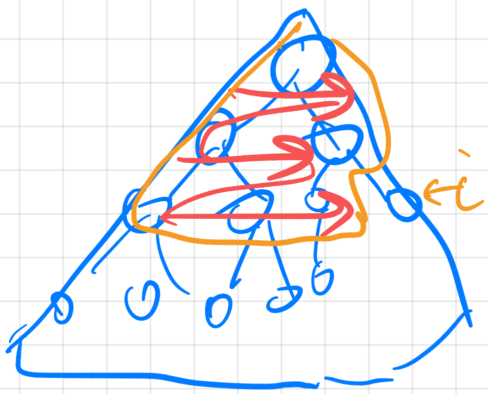
$a = \{5, 7, 3\}$

add()
$O(\log n)$

$H = []$
for x : a:
    add(H, x)

$O(n \log n)$

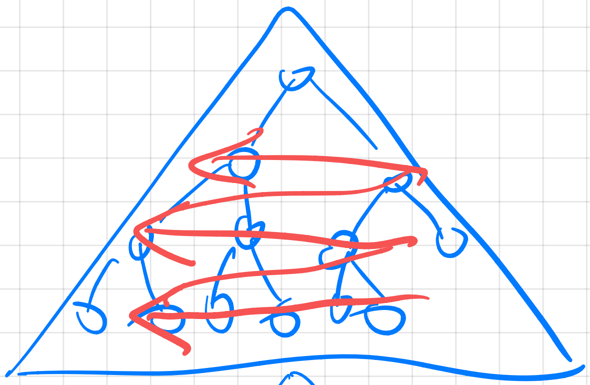
Построение быстрее



$H = [\ldots]$

for $i = 0..n-1$
    siftUp(i)

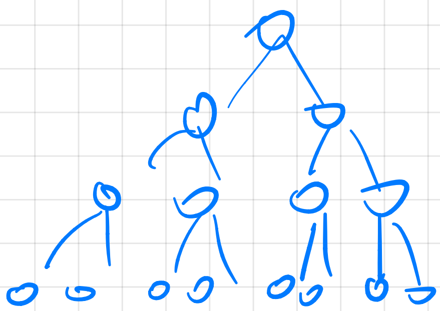
$O(n \log n)$
$(\Theta(n \log n)$:($)$



for i = n-1..0:
siftDown(i)

$O(n \log n)$

Оценка лучше



1	2^0	
2	⋮	$\times 3$
4	2^{k-1}	$\times 2$
8	2^k	$\times 1$

15

$$n = 2^{k+1} - 1$$

$$T = 2^k \cdot 1 + 2^{k-1} \cdot 2 + 2^{k-2} \cdot 3 + \ldots + 2^0 \cdot k =$$

$$= \sum_{i=0}^{k} 2^{k-i}(i+1)$$

$$= \underset{\frac{n+1}{2}}{\underset{\shortparallel}{2^k}} \underbrace{\sum_{i=0}^{k} \frac{i+1}{2^i}}_{\Theta(1)} = \Theta(n)$$

$$S = \sum_{i=0}^{\infty} \frac{i+1}{2^i} = \underbrace{\sum_{i=0}^{\infty} \frac{i}{2^i}}_{S'} + \underbrace{\boxed{\sum_{i=0}^{\infty} \frac{1}{2^i}}}_{2}$$

$$S' = \sum_{i=0}^{\infty} \frac{i}{2^i}$$

$$2S' = \sum_{i=0}^{\infty} \frac{i}{2^{i-1}} = \sum_{i=0}^{\infty} \frac{i-1}{2^{i-1}} + \boxed{\sum_{i=0}^{\infty} \frac{1}{2^{i-1}}}$$

$$\sum_{i=0}^{\infty} \frac{i-1}{2^{i-1}} = \sum_{i=-1}^{\infty} \frac{i}{2^i} = S' + \frac{-1}{2^{-1}} = S' - 2$$

$$\sum_{i=0}^{\infty} \frac{1}{2^{i-1}} = 4$$

$$2S' = S' - 2 + 4 = S' + 2$$

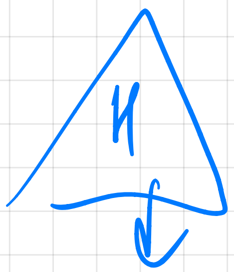
$$S' = 2$$

$$S = S' + 2 = 4$$

Heap Sort

$a = \{2 \quad 4 \quad 5 \quad 3 \quad 6\}$

↓ Build heap $\qquad O(n)$

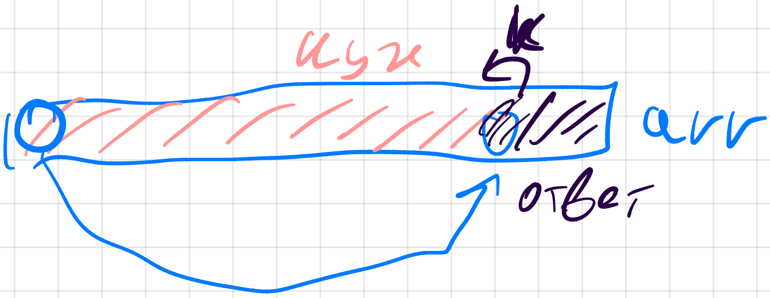


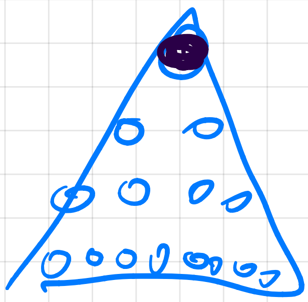
$n \times$ ExtMin $\qquad O(n \log n)$

$O(n \log n)$

```
H = Build heap(arr)
ans = [ ]
for i = 0..n-1 :
    ans.append(extMin(H))
```

Без доп памяти





Куча на MAX

Квадратичные сортировки

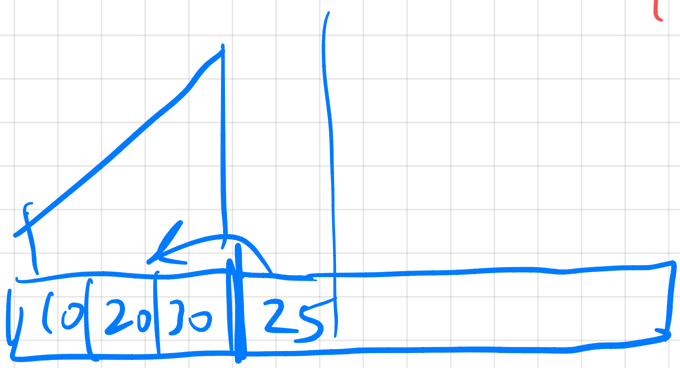
- Selection Sort (выбором)

$\Theta(n^2)$

```
for i=0..n-1:
    p=i
    for j=i+1..n-1
        if a[j]<a[p]
            p=j
    swap(a[p], a[i])
```

- Insertion Sort (Вставками) $\Theta(n^2)$

1960 frank, Lazarus



i

10 20 25 30

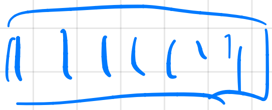
$[0; i)$ отсорт

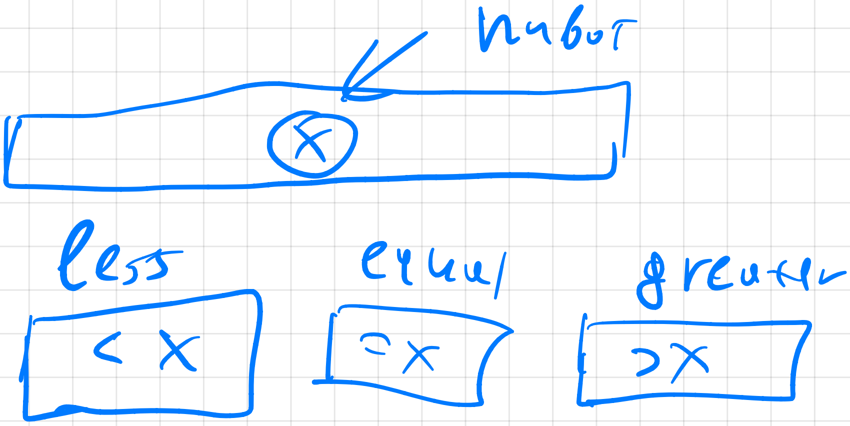
$\Downarrow$

$[0; i+1)$ отсорт.

```
for i = 0..n-1
    j = i
    while a[j] > a[j-1]:
        swap(a[j], a[j-1])
        --j;
```

Quick Sort 1959





```
def Sort(a):
    if len(a) <= 1:
        return a
    x = choice(a)
    less, equal, greater = [],[],[]
    for e in a:
        положить e в
        нужный список
    return Sort(less) +
        + equal + Sort(greater)
```

Идеальный случ.

$$T(n) = 2T\left(\frac{n}{2}\right) + n =$$

$$= \Theta(n \log n)$$

Худший случай

$$T(n) = T(n-1) + n =$$

$$= n + (n-1) + (n-2) + \ldots + 1$$

$$= \frac{n(n+1)}{2} = \Theta(n^2)$$

Средний случай

$$\Rightarrow O(n \log n)$$

Случайность в алг.

1. Randomized

Алгоритмы которые всегда работают, но время работы это случ. величина

Мат. ожидание времени работы

2. Probabilistic

Алгоритмы которые используют случ. биты и с какой-то вер. ошиб.

Quick Sort

1) Мат ожидание
2) предположим что входные данные перестановка $\{2, 3, 1, 4\}$
3) считаем число сравнений (мат. ожидание)

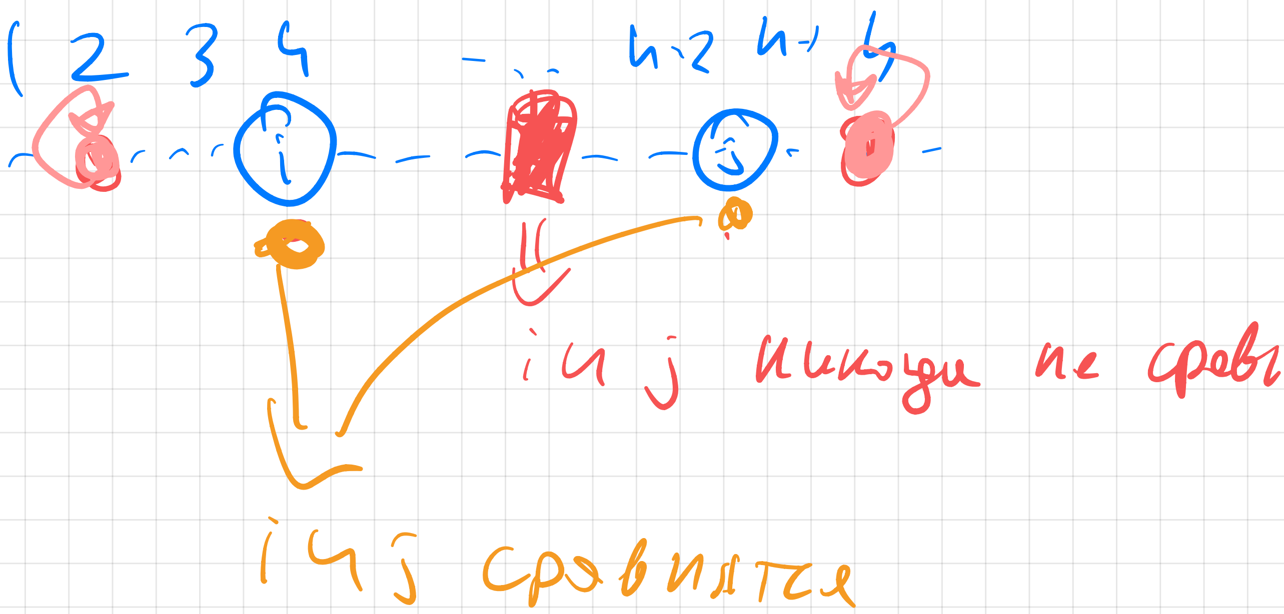
$E[$число сравнений$] =$

$$= E\Big[\sum_{1\le i<j\le n} \mathbb{1}_{\text{сравнили } i \text{ и } j}\Big] =$$

$\{0,1\}$

$$= \sum_{1\le i<j\le n} E\big[\mathbb{1}_{\text{сравнили } i \text{ и } j}\big] =$$

$$= \sum_{1\le i<j\le n} P\big[\text{сравнили } i \text{ и } j\big] =$$



$$= \sum_{1\le i<j\le n} \frac{2}{j-i+1}$$

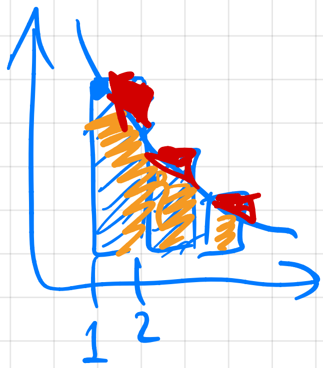
$\circ\ \circ\ \circ$ $n-2$

$$= \sum_{d=2\ldots n} \frac{2}{d} \cdot (n-d+1)$$

$$\leq 2n \cdot \sum_{d=2\ldots n} \frac{1}{d} \leq 2n(\ln(n+1)+1) = \Theta(n\log n)$$

$$\sum_{d=1}^{n} \frac{1}{d} = \ln(n) + \xi + o(1)$$

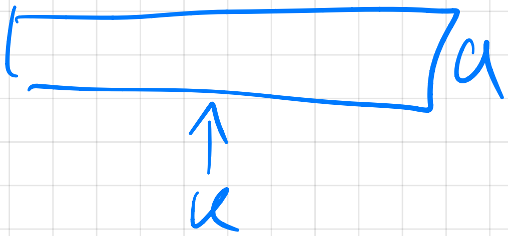
(повторить)



$$\int_1^{n+1} \frac{1}{x}\,dx$$

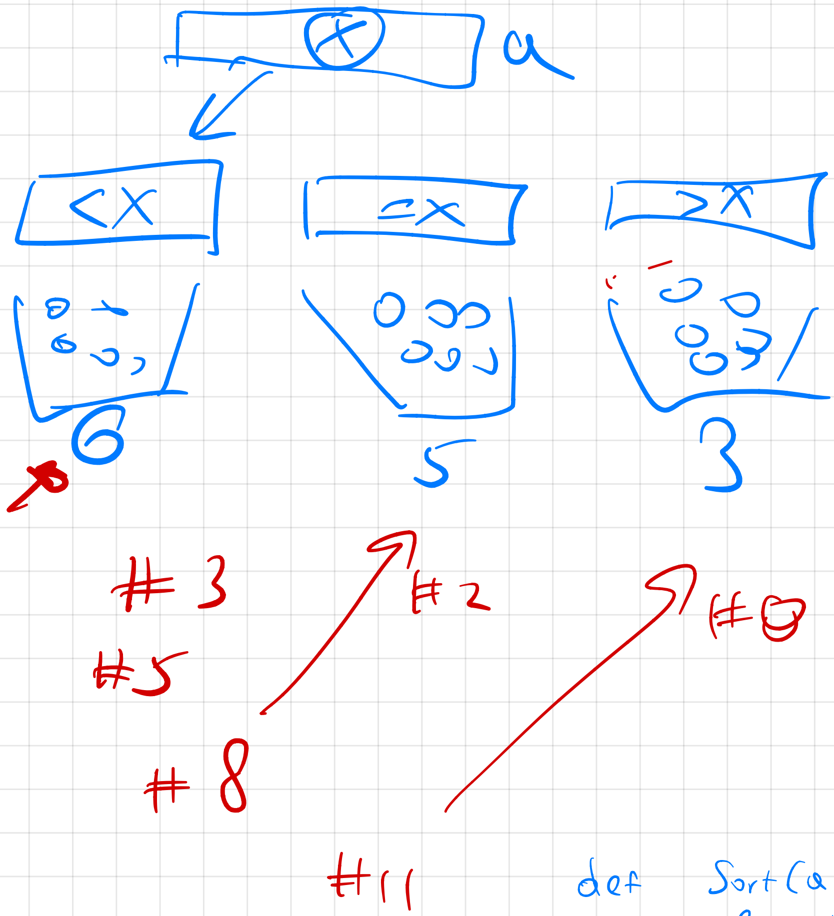
$$\underbrace{\int_1^{n+1} \frac{1}{x}\,dx}_{=\ln(n+1)} \leq \sum_{d=1}^{n} \frac{1}{d} \leq \underbrace{\int_1^{n+1} \frac{1}{x}\,dx + 1}_{=\ln(n+1)+1}$$

Порядковая статистика.



Sort(a)
a[k]
$\Theta(n\log n)$

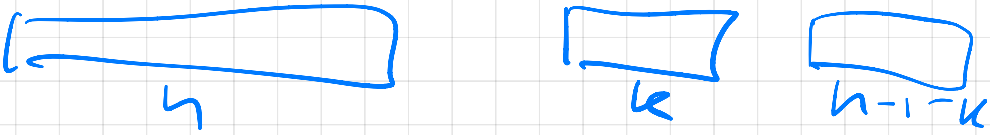
Q. Sort




```
def  Sort(a, k)
    if len(a) ≤ 1:
        return a
    x = choice(a)
    less, equal, greater = [],[],[]
    for e in a:
        положить e в
        нужный список
```

перейти в

Sort(less / equal / greater, k')



Q. Sort

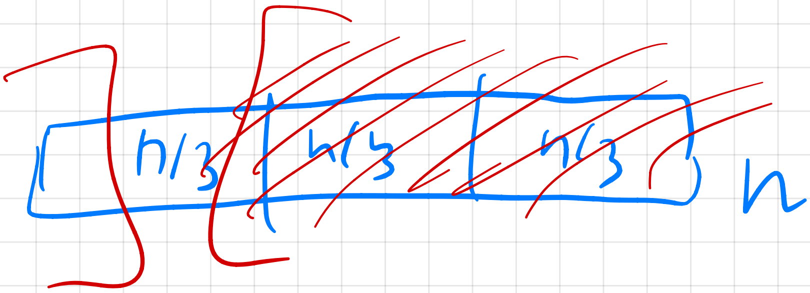
$$T(n) \le T(k) + T(n-1-k) + n$$

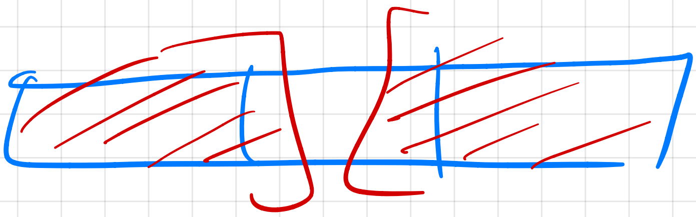
$$T(n) = \frac{1}{n}\sum_{k=0}^{n-1} T(k) + T(n-1-k) + n$$

Порядк. ст:

$$T(n) \le MAX(T(k), T(n-1-k)) + n$$

$$T(n) = \frac{1}{n}\sum_{k=0}^{n-1} max(T(k), T(n-1-k)) + n$$





$p = \frac{1}{3}$

$\le T(\frac{2}{3}n)$

$p = \frac{2}{3}$

$\le T(n)$

$$T(n) \le \frac{1}{3} T\left(\frac{2}{3}n\right) + \frac{2}{3}T(n) + n$$

$$\frac{1}{3}T(n) \le \frac{1}{3}T\left(\frac{2}{3}n\right) + n$$

$$T(n) \le T\left(\frac{2}{3}n\right) + 3n$$

$$T(n) \le 3\Big(\underbrace{n + \frac{2}{3}n + \frac{4}{9}n + \dots}_{3n}\Big)$$

$$\le 9n$$

$$T(n) = O(n)$$

$$1 + q + q^2 + \dots + q^k + \dots$$
$$= \frac{1}{1-q}$$