

Лекция 13.

02.12

B-дерево, Декартово дерево.

t - фикс число; $t \approx 1000$

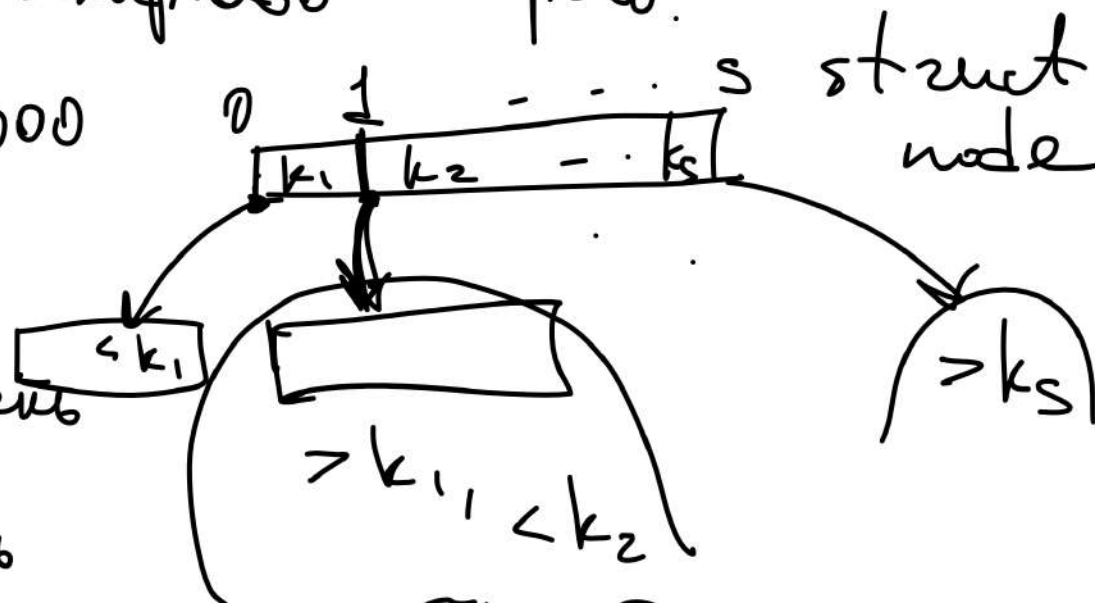
$k_1 \mid k_2 \mid \dots \mid k_s$

$$t-1 \leq s \leq 2t-1$$

$$1 \leq s \leq 2t-1$$

не корень

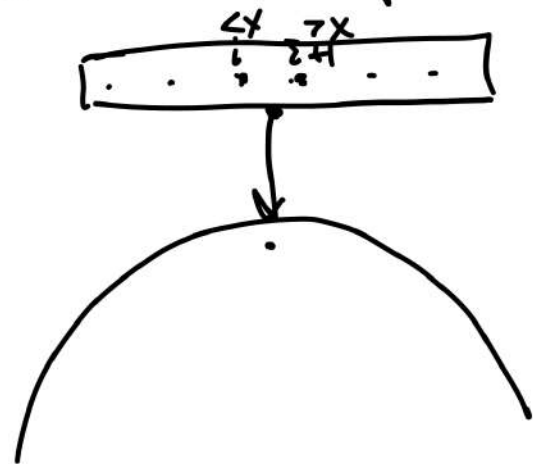
корень



Все листья на одной глубине

$\{B\}$ - блок памяти
Алгоритмы во внешней памяти

find в B-дереве.



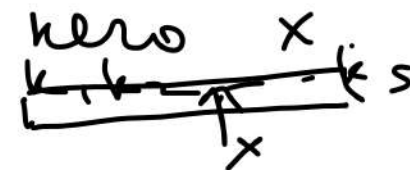
x

Асимптотика:

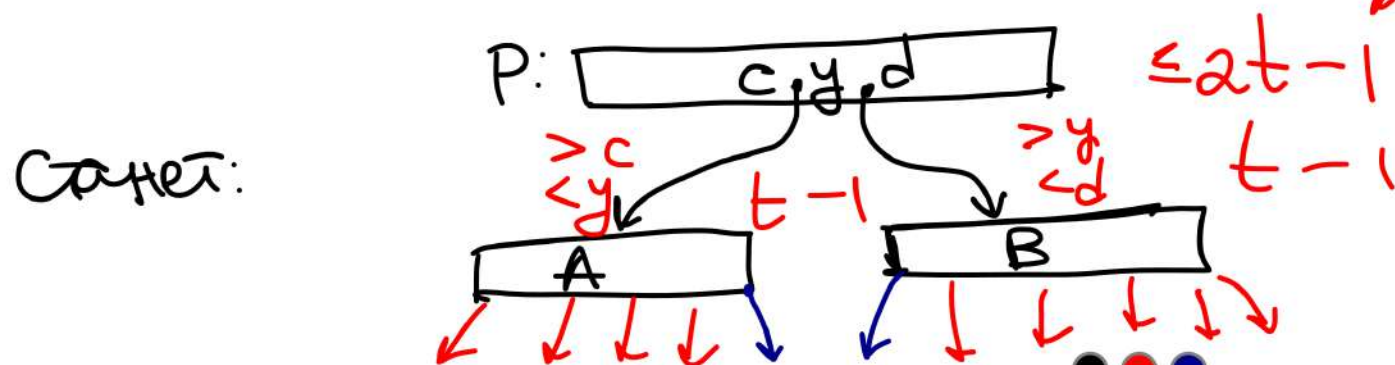
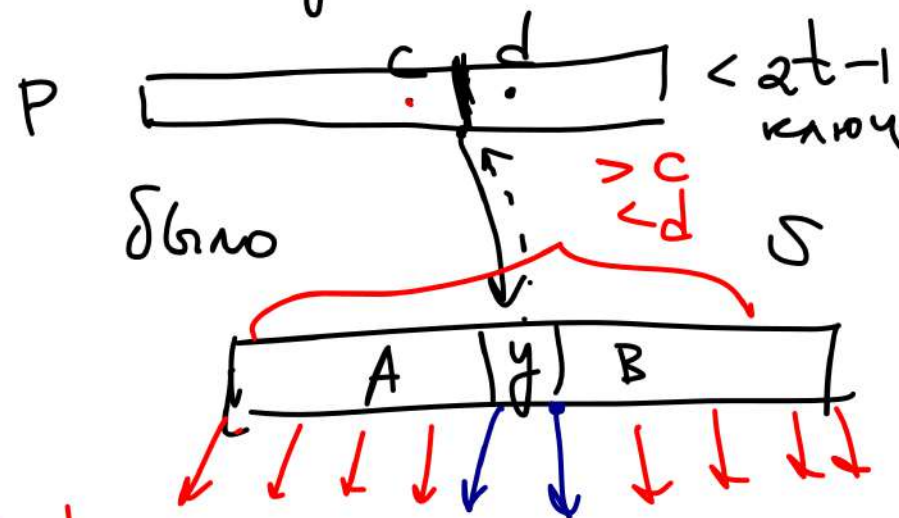
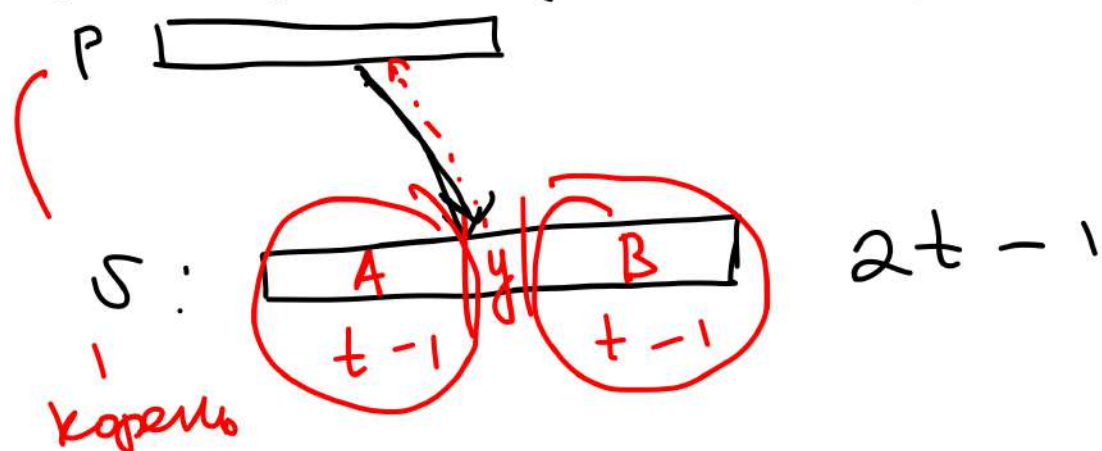
$$O(t \cdot \underbrace{\log_{ft} n}_{\text{глубина}}) \approx O(\log n),$$

если $t = \text{const}$

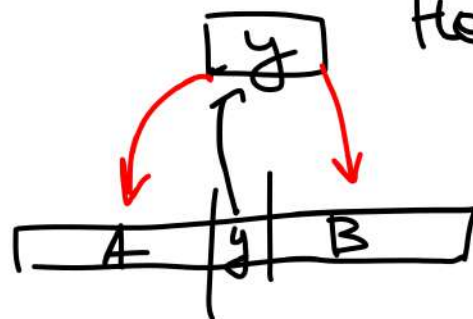
insert x. Запускаем find x. Если x есть в дереве, то делать ничего не нужно. Иначе мы спускаемся до места. Если в нём $< 2t - 1$ узлов, просто добавим в него x.



Дайте каждому ргу, когда алгоритм
попадает в заполненную верш. ($2t-1$ ключ)
делать так, чтобы она стала незаполненной.

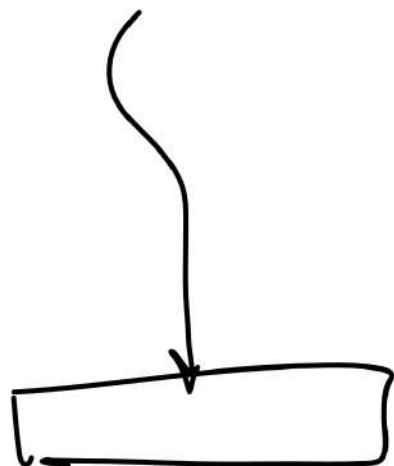


γ - корень



Новый корень
 $[1, 2t - 1]$

$2t - 1$



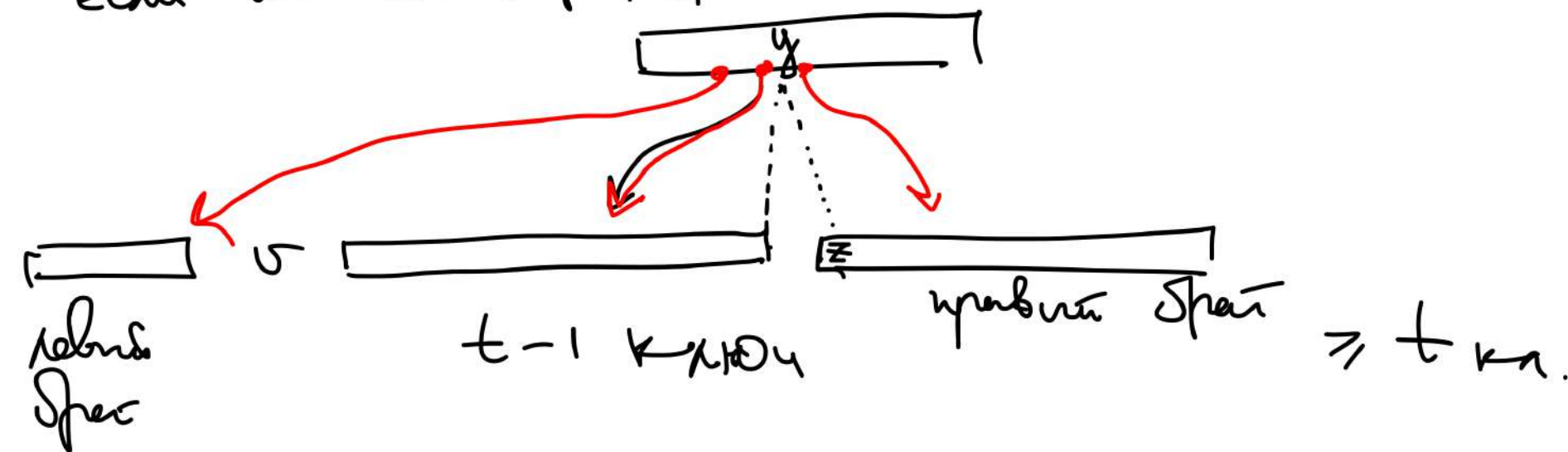
мст

незаконч. \Rightarrow в кее можно
 вставить x

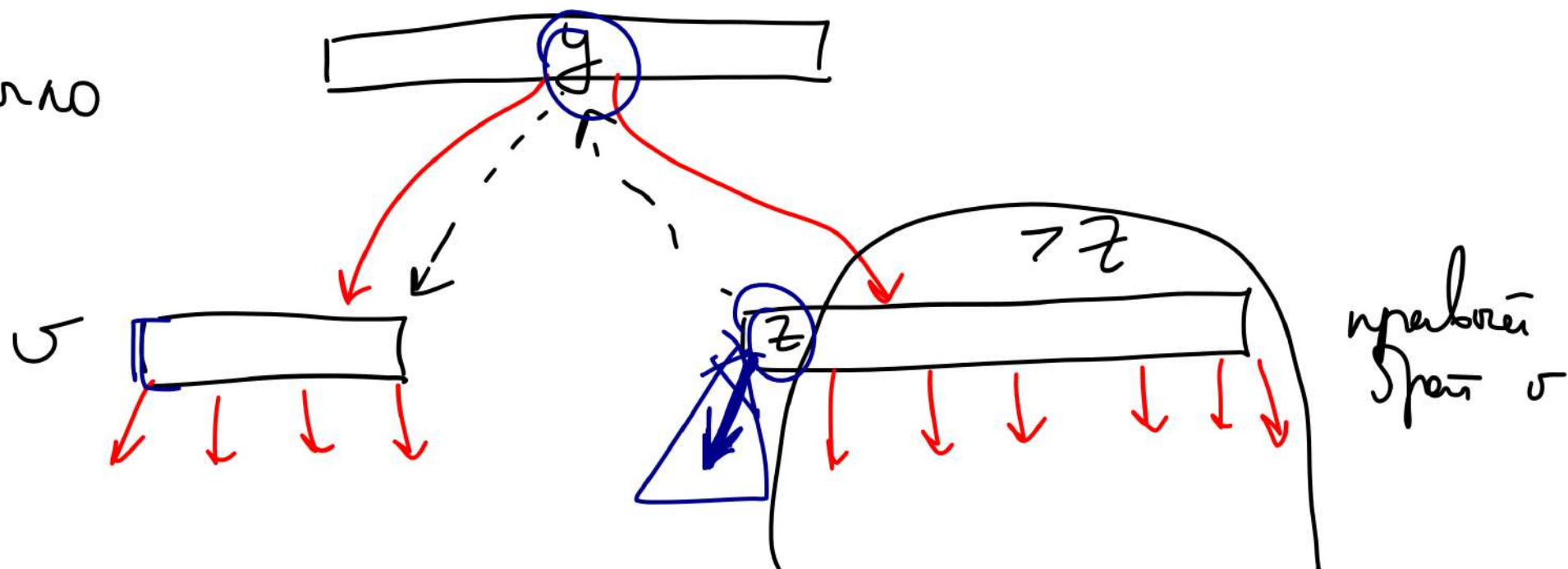
Зам Всего 1 рекурсивный спуск, без
 подёмок



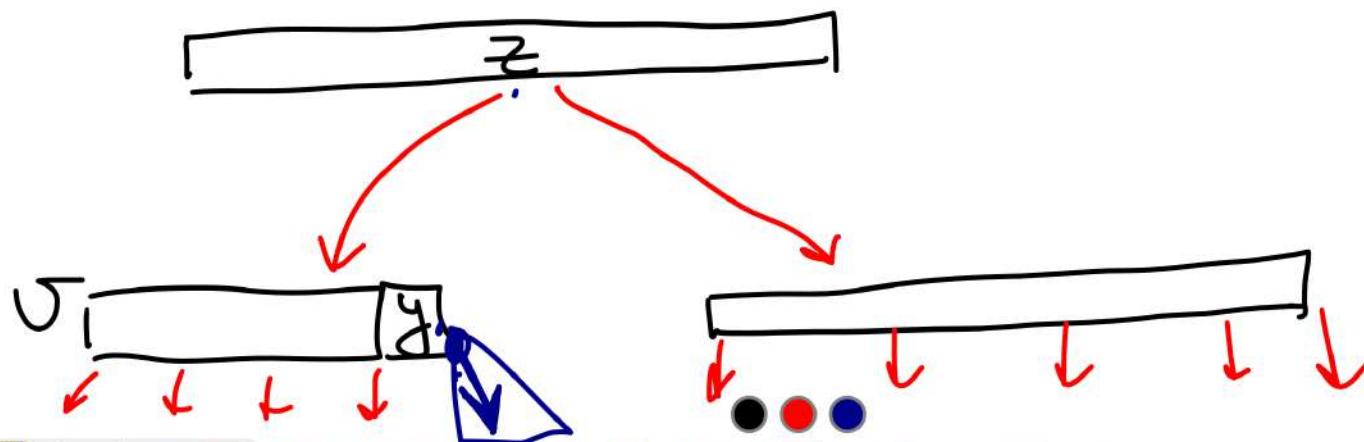
erase x . Давайте спустаться
от корня в листьях x и делать так,
чтобы в текущей вершине было $\geq t$ узлов,
если это не корень.



Σημ 10




Станет



То же происходит, если у γ
в левом брате $\geq t$ ключей

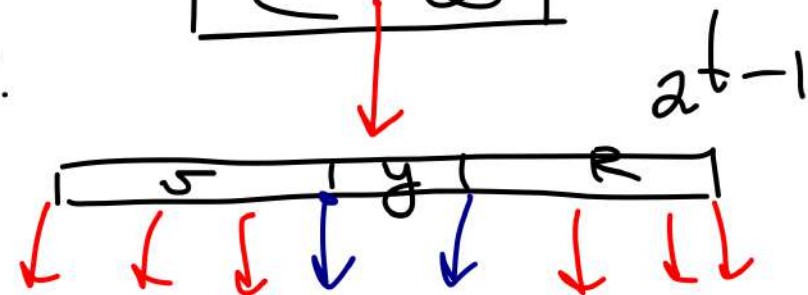
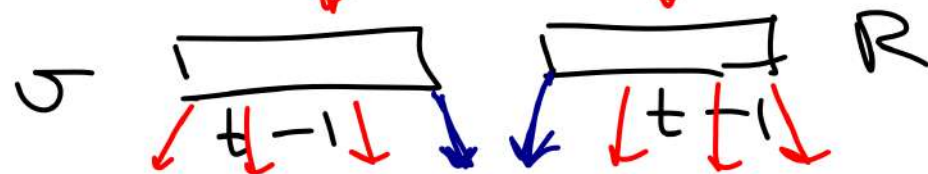
Пусть теперь в обоих братьях по $(t-1)$ ключу
(точнее, каждый брат либо не существует,
либо в нем ровно $(t-1)$ ключ).

В таком случае есть хотя бы 1 брат

брат γ  родитель

счит:





-1 ключ

Если родителем — корень
 корень

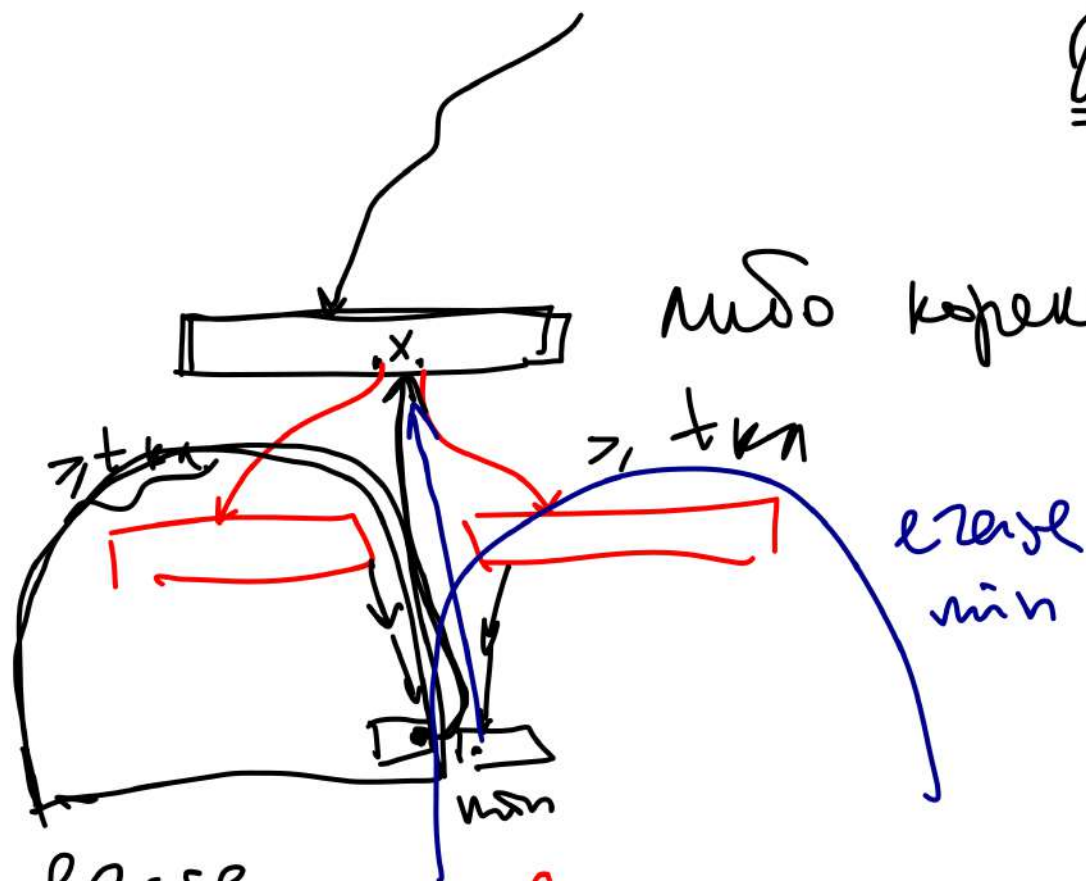




Если у бота единств., ключом в корне,
 то старый корень нужно удалить, новым
 корнем назначить γ у R .

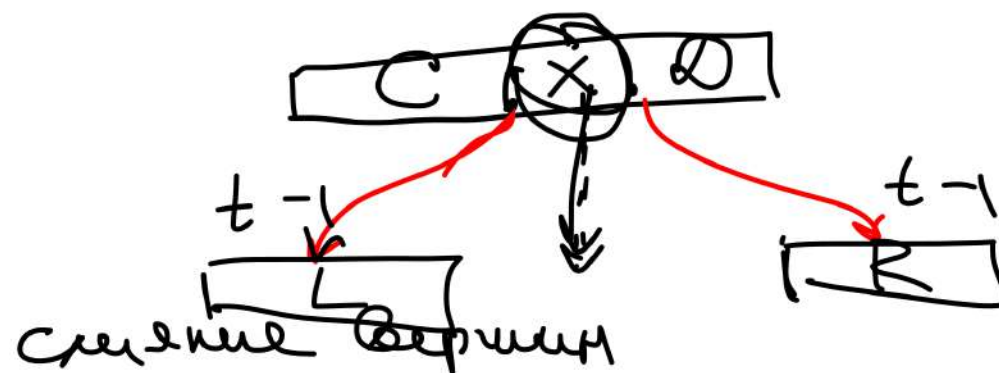
$$\underline{O}(t \cdot \log_h^n)$$

либо корень, либо $\geq t$ ключей,

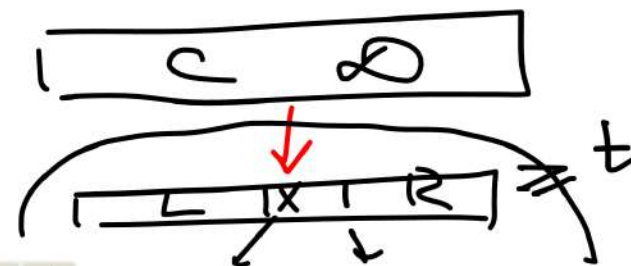


erase
max

наибольшее
дерево поиска



среднее дерево



Декартово дерево.

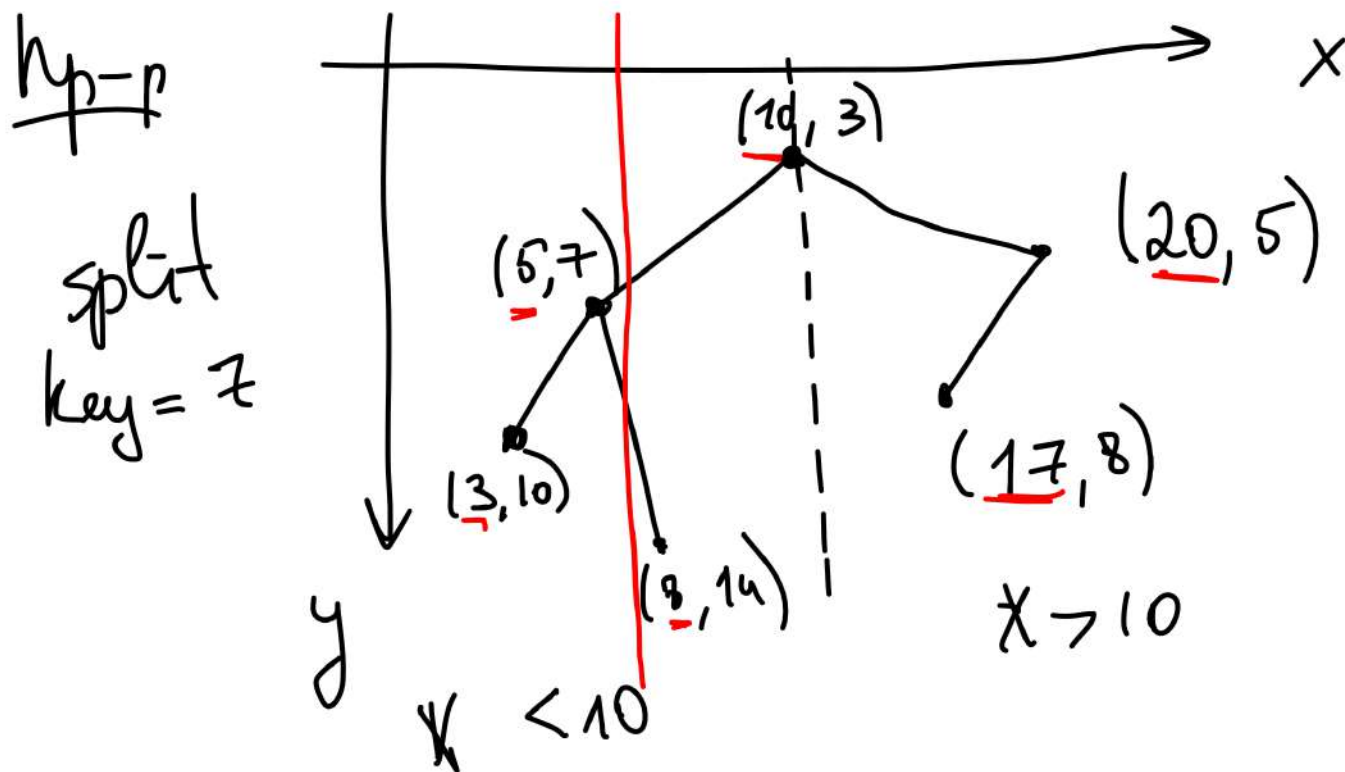
— дерево поиска средних (ожидаемая) глубина которого есть $\underline{O}(\log n)$.

Мат. ожидание глубины = $\underline{O}(\log n)$

Опр Декартово дерево — это бинарное дерево

поиска по ключам (x) и y с min в корне по приоритетам (y).

созд. сгенерировано для каждого x



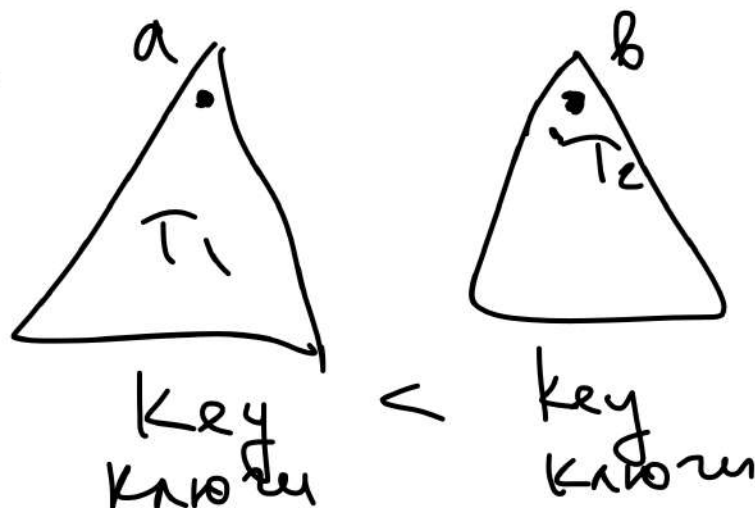
Ynp Если все ключи и
все приоритеты равны, то
Декартово Древо стр. одност.
более того, можно $\mathcal{O}(n)$

Тепр (5/2)
Если в ДД
все приоритеты
свернут. структура,
то
 $\mathbb{E} h(\text{Декарт. Древо})$
 $= \mathcal{O}(\log n)$

Плюс: простота в написании

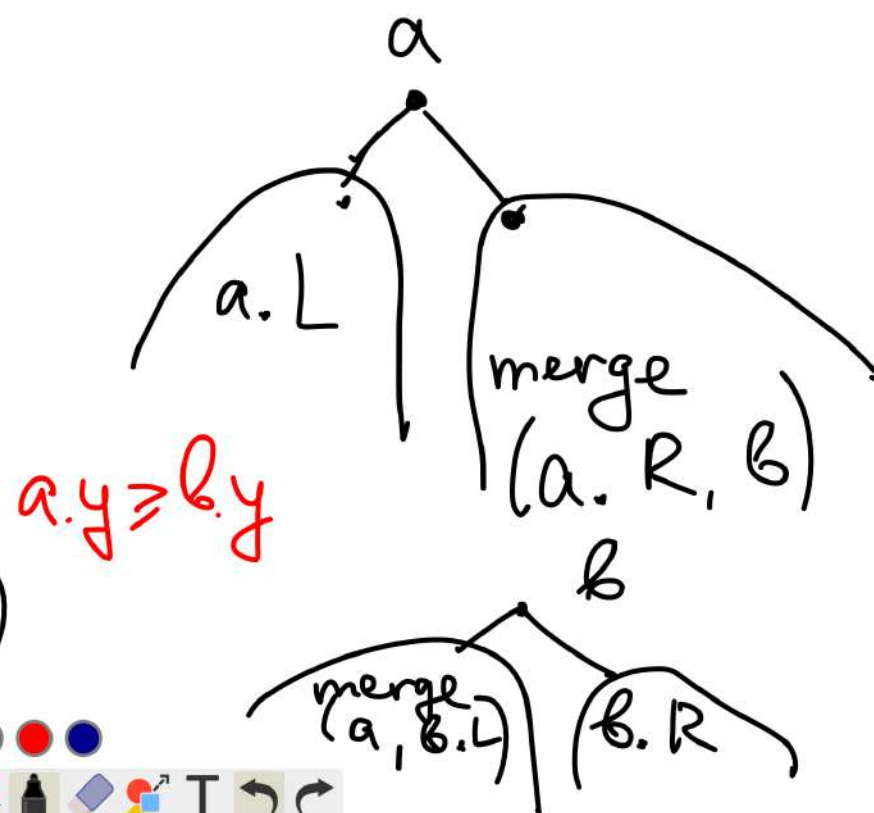
Минус: всё вероятностное
split и merge.

merge



Асимпт: $O(\text{depth}(T_1) + \text{depth}(T_2))$

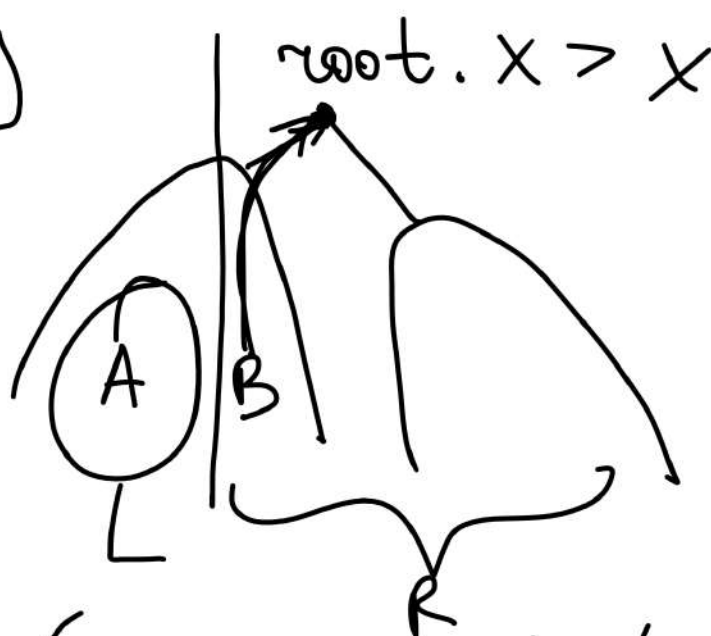
$a.y < b.y \Rightarrow a$ — корень
реж-та



Split
by
(L, R)



T, x

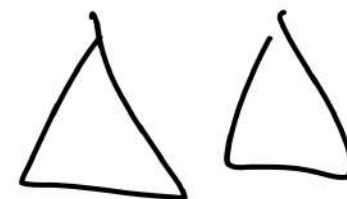


Even $root.x > x$, to $(A, B) = \text{split}(root.L, x)$
 $root.L = B$; return $(A, root)$

Unare

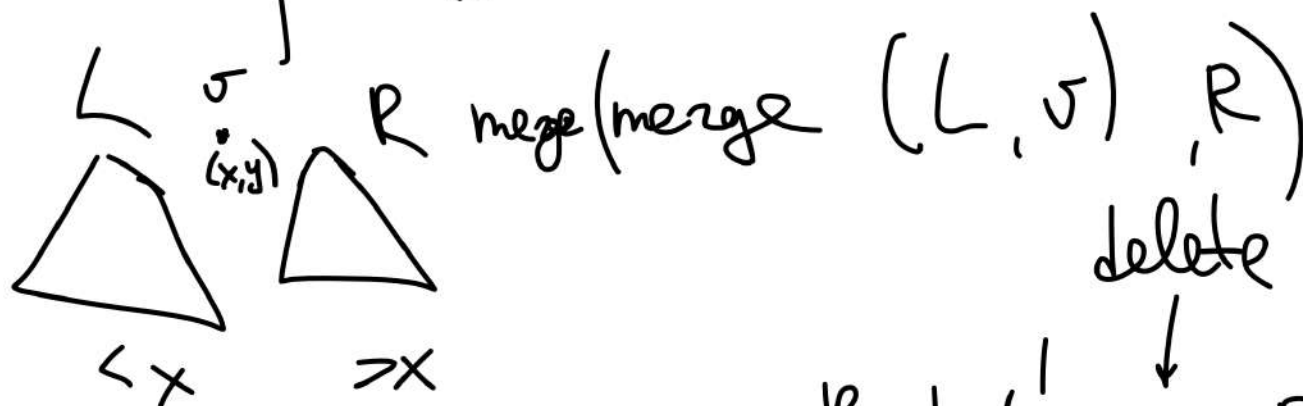
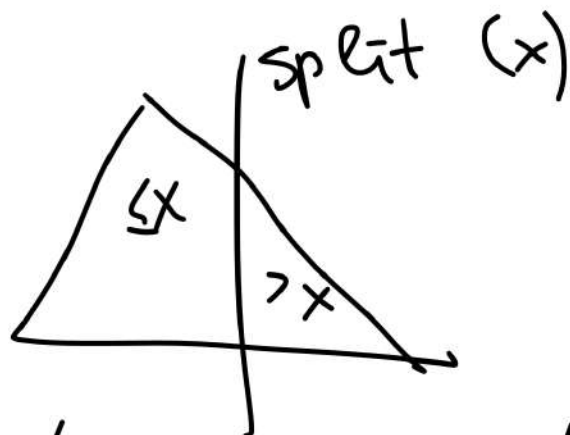
$(A, B) = \text{split}(root.R, x)$
 $root.R = A$; return $(root, B)$

Answer: $\mathcal{O}(\text{depth}(T)) = \mathcal{O}(\log n)$

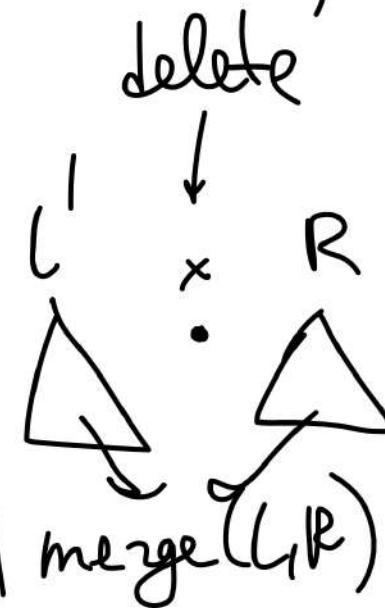
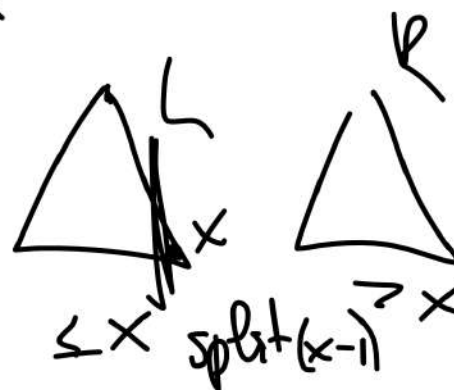
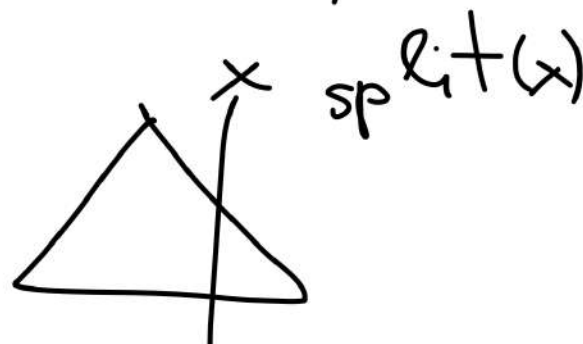


insert, erase

insert x
generate y



erase x



Нелбное древо поиска
(Дерево поиска по нелбному ключу)

Не будет крайних ключи.

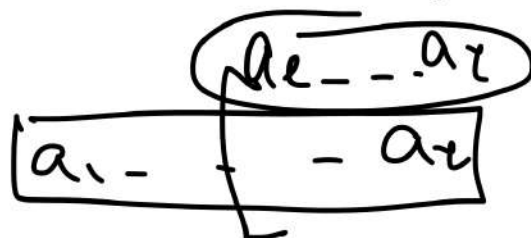
Задача



→ insert val pos

→ erase

→ найти



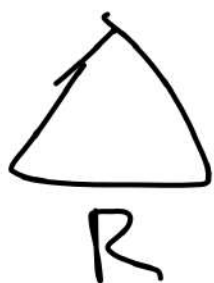
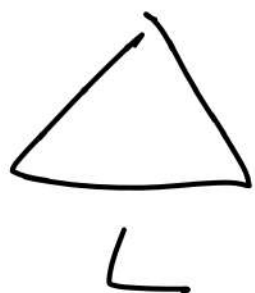
$$\sum_{i=l}^{pos} a_i$$

$$n \leftrightarrow a_{pos};$$

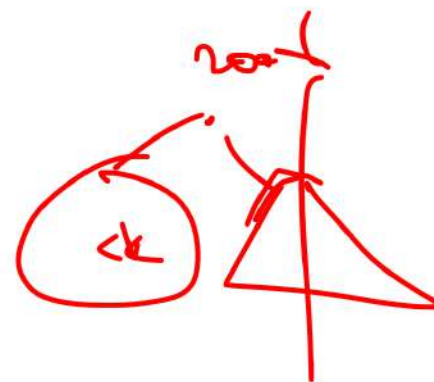
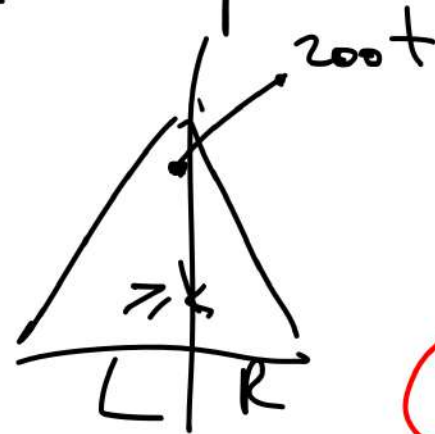
сумма
a-мек
в поддереве,
размер
поддерева



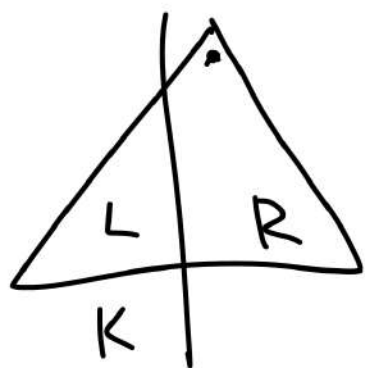
merge ke len, T, k . Or ke chhoti
ka karon,



$$O(\log n)$$



split By Size (T, k) :



if $root.L.size \geq k$:

$$(A, B) = \text{split}(root.L, k)$$

$$root.L = B; \text{return}(A, root)$$

else $(A, B) = \text{split}(root.R, k - root.L.size - 1)$

$$root.R = A; \text{return}(root, B)$$