

# Декартовы деревья

Борис Золотов (Б09, Б11)

Михаил Иванов (Б07, Б08)

Иван Казменко (Б01, Б05)

Владислав Макаров (Б02, Б03)

Мария Радионова (Б06)

Арина Филимонова (Б10)

Санкт-Петербургский Государственный Университет  
Факультет математики и компьютерных наук

Среда, 19 ноября 2025 года

# Содержание

## 1 Введение

- Деревья поиска
- Декартово дерево

## 2 Базовые операции

- Операции
- Иллюстрации
- Операция `split`
- Операция `merge`
- Дальнейшее занятие
- Дополнительное чтение

# Содержание

## 1 Введение

- Деревья поиска
- Декартово дерево

## 2 Базовые операции

- Операции
- Иллюстрации
- Операция `split`
- Операция `merge`
- Дальнейшее занятие
- Дополнительное чтение

# Деревья поиска

Декартово дерево — это разновидность сбалансированного двоичного дерева поиска.

# Деревья поиска

Декартово дерево — это разновидность сбалансированного двоичного дерева поиска.

- Дерево — связный граф без циклов.

# Деревья поиска

Декартово дерево — это разновидность сбалансированного двоичного дерева поиска.

- Дерево — связный граф без циклов.
- Двоичное дерево — корневое дерево, в котором у каждой вершины не более двух потомков — левый и правый.

# Деревья поиска

Декартово дерево — это разновидность сбалансированного двоичного дерева поиска.

- Дерево — связный граф без циклов.
- Двоичное дерево — корневое дерево, в котором у каждой вершины не более двух потомков — левый и правый.
- Дерево поиска — корневое дерево с числами в вершинах: число в вершине больше всех чисел в левом поддереве, но меньше всех чисел в правом поддереве.
  - При равенстве можно оказаться в любом поддереве.
  - Числа → любые объекты, на которых определён порядок.
  - Поиск: начать с корня, идти в правильное поддерево.

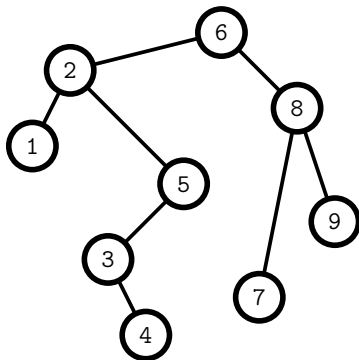
# Деревья поиска

Декартово дерево — это разновидность сбалансированного двоичного дерева поиска.

- Дерево — связный граф без циклов.
- Двоичное дерево — корневое дерево, в котором у каждой вершины не более двух потомков — левый и правый.
- Дерево поиска — корневое дерево с числами в вершинах: число в вершине больше всех чисел в левом поддереве, но меньше всех чисел в правом поддереве.
  - При равенстве можно оказаться в любом поддереве.
  - Числа → любые объекты, на которых определён порядок.
  - Поиск: начать с корня, идти в правильное поддерево.
- Сбалансированное дерево — корневое дерево, в котором максимальная глубина вершины *невелика*.
  - Глубина вершины — расстояние от неё до корня.
  - Время на поиск в худшем случае — глубина дерева.

# Пример

Пример:



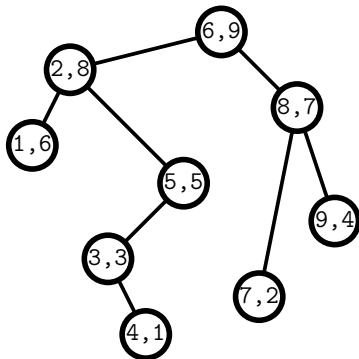
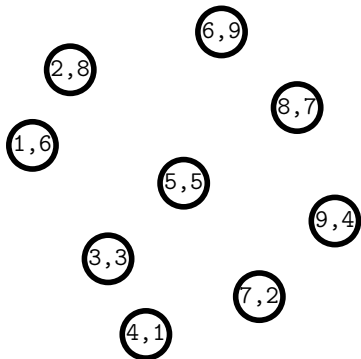
# Декартово дерево

Декартово дерево — это дерево, в котором каждая вершина содержит пару чисел  $(x, y)$ :

- По координате  $x$  это двоичное дерево поиска: число в вершине больше всех чисел в левом поддереве, но меньше всех чисел в правом поддереве.
- По координате  $y$  это двоичная куча: число в вершине больше всех чисел в поддеревьях.

Удобно представлять вершины как точки  $(x, y)$  на плоскости, в декартовой системе координат.

# Декартово дерево



# Декартово дерево

**Утверждение:** если все  $x$  различны и все  $y$  различны, существует единственное декартово дерево с такими вершинами.

# Декартово дерево

**Утверждение:** если все  $x$  различны и все  $y$  различны, существует единственное декартово дерево с такими вершинами.

- Докажем конструктивно: предъявим алгоритм построения.

# Декартово дерево

**Утверждение:** если все  $x$  различны и все  $y$  различны, существует единственное декартово дерево с такими вершинами.

- Докажем конструктивно: предъявим алгоритм построения.
- Корень дерева — это вершина с наибольшим  $y$ .
- Все точки слева от неё образуют левое поддереву, а все точки справа — правое поддереву.
- Переходим к двум подзадачам, в которых поддеревья строятся рекурсивно тем же алгоритмом.

# Выбор структуры

Почему на практике мы изучаем именно декартово дерево?

- коротко пишется
- мощное: умеет решать много интересных задач
- скорее всего, нет в стандартной библиотеке
  - C++: Policy-based data structures (GCC extension)?..

# Содержание

- 1 Введение
  - Деревья поиска
  - Декартово дерево
- 2 Базовые операции
  - Операции
  - Иллюстрации
  - Операция `split`
  - Операция `merge`
  - Дальнейшее занятие
  - Дополнительное чтение

# Операции

Интерфейс двоичного дерева поиска:

- `insert (t, x)` — вставить элемент `x` в дерево `t`
- `remove (t, x)` — удалить элемент `x` из дерева `t`
- `find (t, x)` — найти элемент `x` в дереве `t` (или выяснить, что его нет)

# Операции

Интерфейс двоичного дерева поиска:

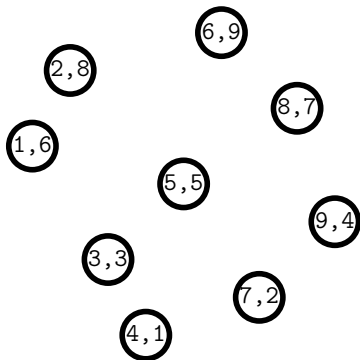
- `insert (t, x)` — вставить элемент  $x$  в дерево  $t$
- `remove (t, x)` — удалить элемент  $x$  из дерева  $t$
- `find (t, x)` — найти элемент  $x$  в дереве  $t$  (или выяснить, что его нет)

Базовые операции декартова дерева:

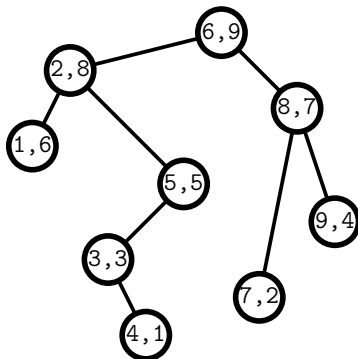
- `split (t, x)` — разбить декартово дерево на два,  $l$  и  $r$ : в одном все числа меньше  $x$ , в другом все остальные
- `merge (l, r)` — объединить два декартова дерева  $l$  и  $r$  в одно при условии, что любое число в  $l$  не больше любого числа в  $r$

Эти операции — взаимно обратные.

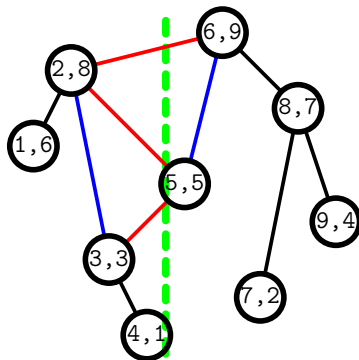
# Иллюстрации 1



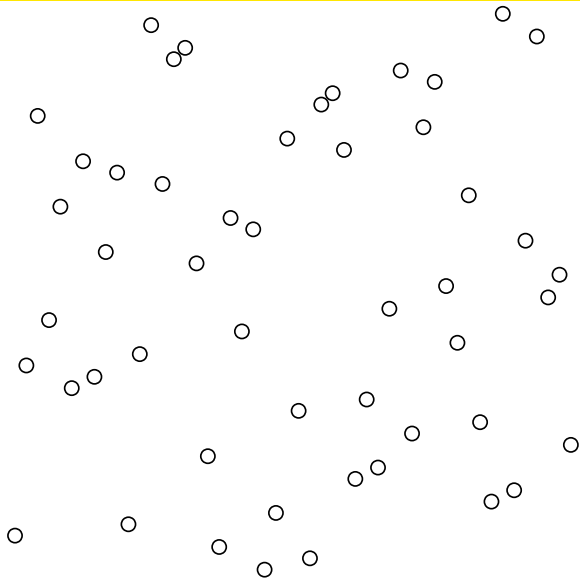
# Иллюстрации 1



## Иллюстрации 1

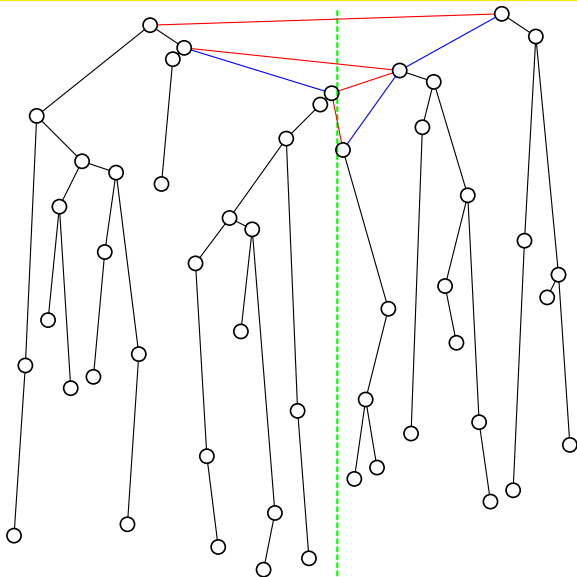


# Иллюстрации 2





## Иллюстрации 2



# Операция `split`

Научимся делать `split` дерева `t` по ключу `xs`.

- Если `t` пустое, то `l` и `r` пустые.
- Если `t.x ≥ xs`, нужно разделить `t.left` по ключу `xs`.
  - Пусть мы получили `ll` и `lr`. Ответ:
    - `l = ll`,
    - `r = t`, у которого вместо `t.left` привешено дерево `lr`.
- Если же `t.x < xs`, нужно разделить `t.right` по ключу `xs`.
  - Пусть мы получили `rl` и `rr`. Ответ:
    - `l = t`, у которого вместо `t.right` привешено дерево `rl`,
    - `r = rr`.

# Операция merge

Научимся делать merge деревьев  $l$  и  $r$ .

- Если  $l$  пустое, то ответ это  $r$ .
- Если  $r$  пустое, то ответ это  $l$ .
- Если  $l.y > r.y$ , нужно  $l$  сделать корнем.
  - Поддерево  $l.left$  останется как есть.
  - Вместо поддерева  $l.right$  будет merge деревьев  $l.right$  и  $r$ .
- Если же  $l.y \leq r.y$ , нужно  $r$  сделать корнем.
  - Вместо поддерева  $r.left$  будет merge деревьев  $l$  и  $r.left$ .
  - Поддерево  $r.right$  останется как есть.

## Дальнейшее занятие

План дальнейшего занятия — написать вместе программу, реализующую декартово дерево, а дальше изменять её, чтобы решить несколько примеров — сколько успеем. Например:

1. Множество: добавление, поиск, удаление.
2. Порядковая статистика: поиск  $k$ -го элемента в порядке возрастания.
3. Массив со следующей операцией: даны  $l_0 < m_e < h_i$ , поменять местами отрезки  $[l_0, m_e)$  и  $[m_e, h_i)$ .
4. Максимум на отрезке  $[l_0, h_i)$ .
5. Переворачивание отрезка  $[l_0, h_i)$ .
6. Приписывание отрезка  $[l_0, h_i)$  в конец массива.

## Дополнительное чтение

Рекомендуемое дополнительное чтение — декартово дерево на Хабре (автор — Александр Полозов):

- <https://habr.com/ru/post/101818/>
- <https://habr.com/ru/post/102006/>
- <https://habr.com/ru/post/102364/>

Вопросы?

Вопросы?