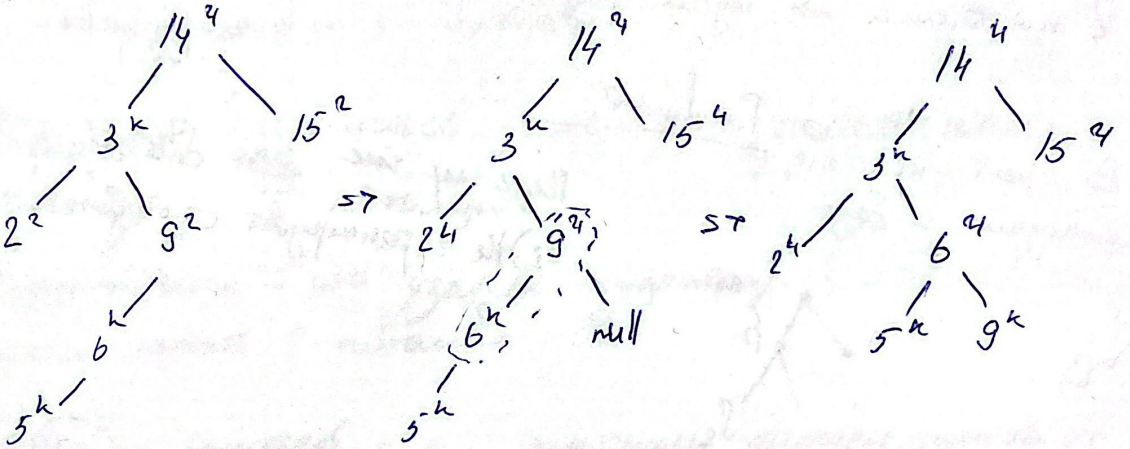
(7.) Красно-черные деревья. Основные операции. Теорема о высоте.

Это BST у которого есть дополнительный бит для каждой вершины, который говорит красная вершина или черная

Свойства RB-дерева:

1. Сразу после вставки любой вершины её нужно покрасить
2. Листами у красных узлов будут - null, а не ребенок без детей.
3. Корень всегда черный, листы всегда черные
4. У всех красных узлов всегда черные дети
5. От корня до любого листа одинаковое количество черных элементов
6. Есть черная высота, это кол-во черных ~~листов~~ узлов от листа до корня



Алгоритм поиска - такой же как в ~~BST~~ BST

Вставка: 1) Обычный поиск → 2) Создание у листика нового элемента →

3) После вставки красим в красный цвет ~~не~~ всегда! → Нарушаем 2 свойство RB-дерева:

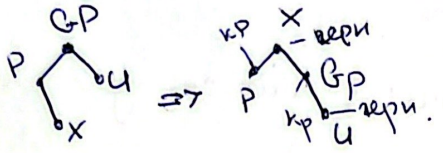
а) Корень может стать красным! → листик перекрашивается в черный цвет

б) У красного элемента может быть ребенок красным → Помним в п.4.

4) 2 случая: ↓

1) Красный ~~отец~~ дядя → переедем цвет от деда!

2) Черный ~~отец~~ дядя → повернитесь относительно деда



Перебалансировка RBT состоит из двух операций – перекраски и ротации.

Алгоритм Удаления:

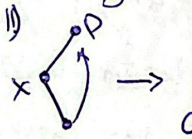
1) Поиск удаляемого элемента.

2) 3 случая:

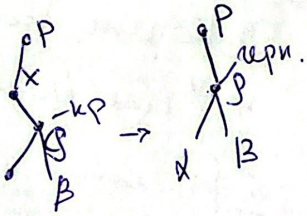
а.) Узел без детей – черный; просто удаляем

б.) Один потомок:

I)

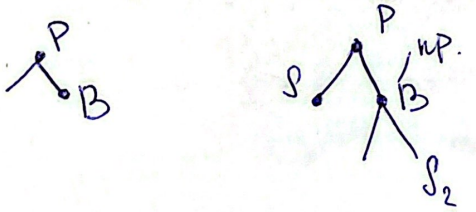


II)



в.) 2 потомка → Идем раз направо, потом лев. элемент в поддереве.

Если В брат – красный, приводим к базе



Главное:

Почему не оно сбалансированное:

1) Не гарантируют строгой сбалансированности,