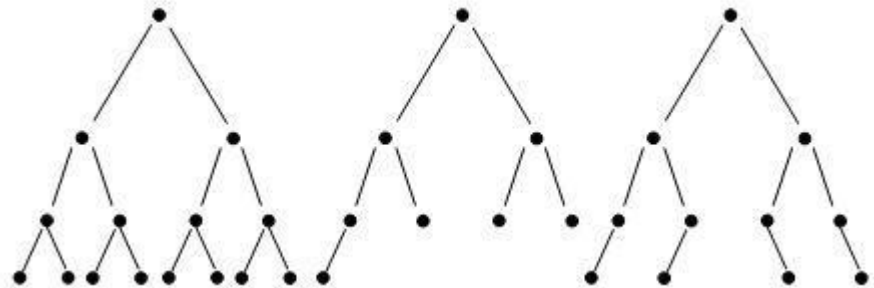


Деревья 2

Идеально сбалансированные бинарные деревья

- * Бинарное дерево назовем *идеально сбалансированным*, если для *каждой* его *вершины* количество *вершин* в *левом* и *правом поддереве* различаются не более чем на 1.



Теорема

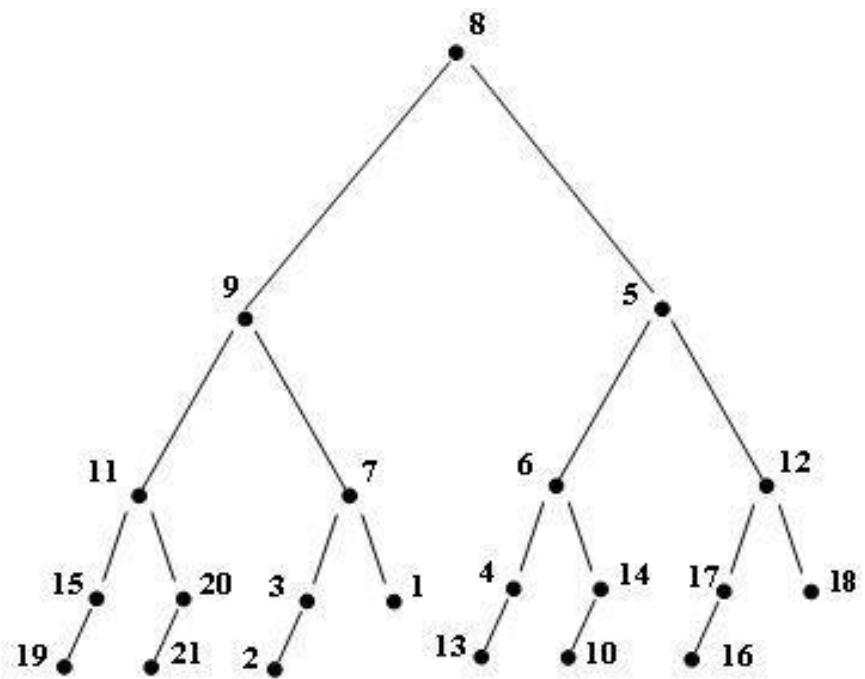
- * Длина внутреннего пути в идеально сбалансированном дереве, содержащем n вершин, не превосходит величины:
- * $(n+1)[\log_2 n] - 2 * 2^{\lceil \log_2 n \rceil} + 2$

Алгоритм построения идеально сбалансированного дерева при известном числе вершин n

- * взять одну вершину в качестве корня.
- * построить левое поддерево с $nl = n \text{ DIV } 2$ вершинами тем же способом.
- * построить правое поддерево с $nr = n - nl - 1$ вершинами тем же способом.

```
* node *Tree (int n, node **p)
* // Построение идеально сбалансированного дерева с n вершинами.
* // *p - указатель на корень дерева.
* {
*   node *now;
*   int x,nl,nr;

*   now = *p;
*   if (n==0) *p = NULL;
*   else
*   { nl = n/2; nr = n - nl - 1; cin>>x;
*     now = new(node);(*now).Key = x;
*     Tree (nl,&((*now).Left)); Tree (nr,&((*now).Right)); *p = now;}
* }
```



Балансированные по высоте деревья (АВЛ-деревья)

- * Бинарное дерево поиска называется **балансированным по высоте**, если для каждой его вершины высота ее двух поддеревьев различается не более, чем на 1. Деревья, удовлетворяющие этому условию, часто называют **АВЛ-деревьями** (по первым буквам фамилий их изобретателей **Г.М.Адельсона-Вельского** и **Е.М.Ландиса**).
- * Показателем балансированности вершины бинарного дерева мы будем называть **разность высоты его правого и левого поддерева**.

Математический анализ AVL-деревьев

Теорема

- * обозначим T_h - AVL-дерево высотой h с количеством узлов N_h . Тогда

$$\log_2(N_k + 1) = k + 1 \leq \frac{1}{\log_2 \frac{1 + \sqrt{5}}{2}} \log_2 \left\{ \frac{\sqrt{5}(3 - \sqrt{5})}{2} (N_k + 1) + 9 - 4\sqrt{5} \right\}$$

Лемма

- * Наибольшая длина ветвей (**$h+1$**) в AVL-дереве, содержащем N_h узлов, определяется неравенством

$$h+1 \leq \frac{1}{\log_2 \frac{1+\sqrt{5}}{2}} * \log \left[\frac{\sqrt{5}(3-\sqrt{5})}{2} * (N_h + 1) + 9 - 4\sqrt{5} \right]$$

Лемма

- * Наименьшая длина ветвей (**$h+1$**) в АВЛ-дереве, содержащем N_h узлов определяется формулой **$h+1 = \log_2(N_h + 1)$**

Теорема

- * Пусть T_h - AVL-дерево высоты h , имеющее N_h узлов. Тогда для **средней длины ветвей дерева** S_h при $h \rightarrow \infty$ имеет место следующая асимптотическая оценка:

$$S_h \sim \frac{1+\sqrt{5}}{2\sqrt{5}} \frac{\log_2 N_h}{\log_2 \frac{1+\sqrt{5}}{2}} \sim 1.04229776903821 \log_2 N_h$$

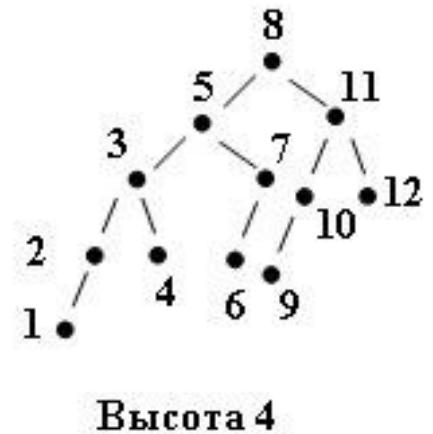
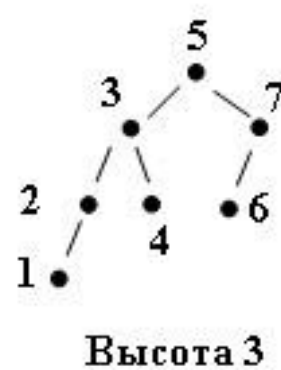
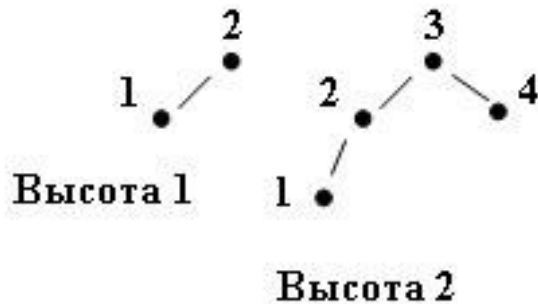
Деревья Фибоначчи

- * **Наиболее асимметричное AVL-дерево T_h высоты h имеет наиболее асимметричное AVL-дерево T_{h-1} высоты $h-1$ в качестве одного из своих поддеревьев и наиболее асимметричное AVL-дерево высоты $h-2$ в качестве другого. Подобные деревья называются деревьями Фибоначчи.**

Дерево Фибоначчи порядка k

- * если $k=0$, то дерево Фибоначчи *пусто*;
- * если $k=1$, то дерево Фибоначчи состоит из единственного узла, ключ которого содержит 1;
- * если $k \geq 2$, то корень дерева Фибоначчи содержит ключ F_k , левое поддерево есть **дерево Фибоначчи порядка $k-1$** , правое поддерево есть **дерево Фибоначчи порядка $k-2$** с ключами в узлах, увеличенными на F_k .
- * F_k - k -е число Фибоначчи: $F_0=1, F_1=1, F_2=2, F_3=3, F_4=5, F_5=8, F_6=13$,

Приммер деревьев Фибоначчи



показатель сбалансированности

- * показатель сбалансированности узла = = высота **правого** поддерева - высота **левого** поддерева.

Балансированные по весу деревья (WB-деревья)

- * Класс бинарных деревьев, в которых ограничения на высоты поддеревьев заменено ограничением на число вершин в поддеревьях.
- * От AVL-деревьев **WB**-деревья отличаются тем, что содержат параметр, который может изменяться так, что можно произвольно ограничивать длину самого длинного пути из корня в висячую вершину за счет увеличения дисбаланса.

* *Корневым балансом $b(T_n)$ бинарного дерева $T_n=(T_l, v, T_r)$ называется величина*

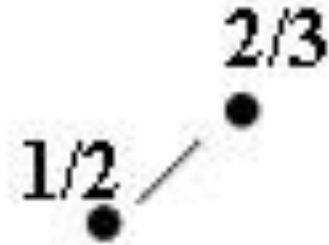
$$n_l+1$$

* $b(T_n) = \frac{n_l+1}{n+1}, n \geq 1$

$$0 < b(T_n) < 1$$

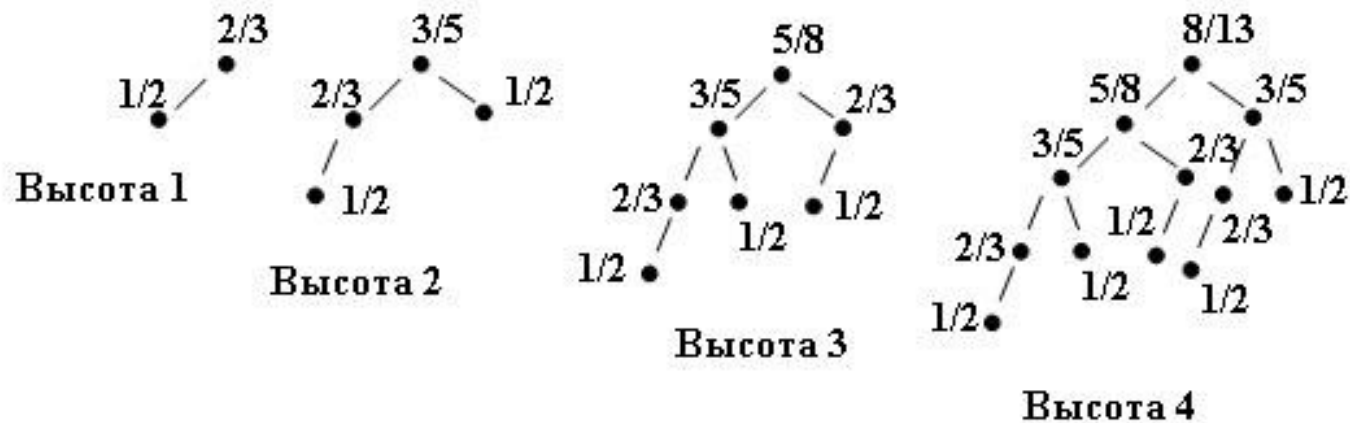
Определение

- * *Дерево T_n называется балансированным по весу с балансом A , $0 < A < 1/2$, если оно удовлетворяет следующим условиям:*
- * *$A \leq b(T_n) \leq 1 - A$;*
- * *T_l, T_r - балансированные по весу деревья с балансом A .*



- * Класс бинарных деревьев с балансом A - $WB[A]$.
- * Пустое бинарное дерево T_0 , по определению, входит в $WB[A]$ для любого A .
- * Класс $WB[A]$ становится все более ограниченным по мере того, как A меняется от 0 до $1/2$.
- * **Случай $1/2$**
- * либо левое и правое поддеревья каждой вершины содержат одинаковое число вершин, поэтому классу $WB[1/2]$ принадлежат полностью сбалансированные деревья с $n=2^k-1$ вершинами,
- * Либо см. рисунок

Деревья Фибоначчи



Теорема

- * Высота дерева T_n из класса $WB[A]$ не превышает

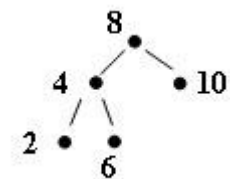
$$\frac{\log_2(n+1) - 1}{\log_2 \frac{1}{1-A}}$$

высота дерева Фибоначчи не превышает $\log_2(n+1)-1$

Алгоритмы балансировки

Включение в сбалансированное дерево новой вершины


- * сначала было $h_L = h_R$. После включения **L** и **R** станут разной высоты, но критерий сбалансированности не будет нарушен;
- * сначала было $h_L < h_R$. После включения **L** и **R** станут равной высоты, то есть критерий сбалансированности даже улучшится;
- * сначала было $h_L > h_R$. После включения критерий сбалансированности нарушится и дерево необходимо перестраивать.



```
* struct node
* {
*   int Key;
*   int Count;
*   int bal; // Показатель балансированности вершины.
*   node *Left;
*   node *Right;
* };
```

Определение

- * Показатель сбалансированности вершины
= *разность между высотой правого и левого
поддерева*

- 
- * Проход по дереву, чтобы убедиться, что включаемого значения в дереве нет;
 - * включение новой вершины и определение результирующего показателя сбалансированности;
 - * "отступление" по пути поиска и проверка в каждой вершине показателя сбалансированности. При необходимости - балансировка.

Однократный LL-поворот

- * `p1 = (*p).Left;`
- * `(*p).Left = (*p1).Right;`
- * `(*p1).Right = p;`
- * `(*p).bal = 0;`
- * `p = p1;`

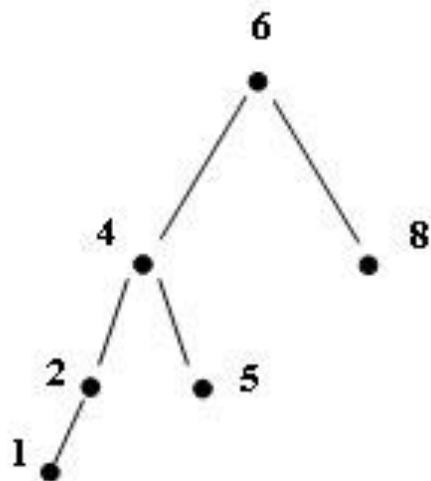
LL-поворот

- * `p1 = (*p).Left;`
- * `(*p).Left = (*p1).Right;`
- * `(*p1).Right = p;`
- * `(*p).bal = 0;`
- * `p = p1;`

RR-поворот

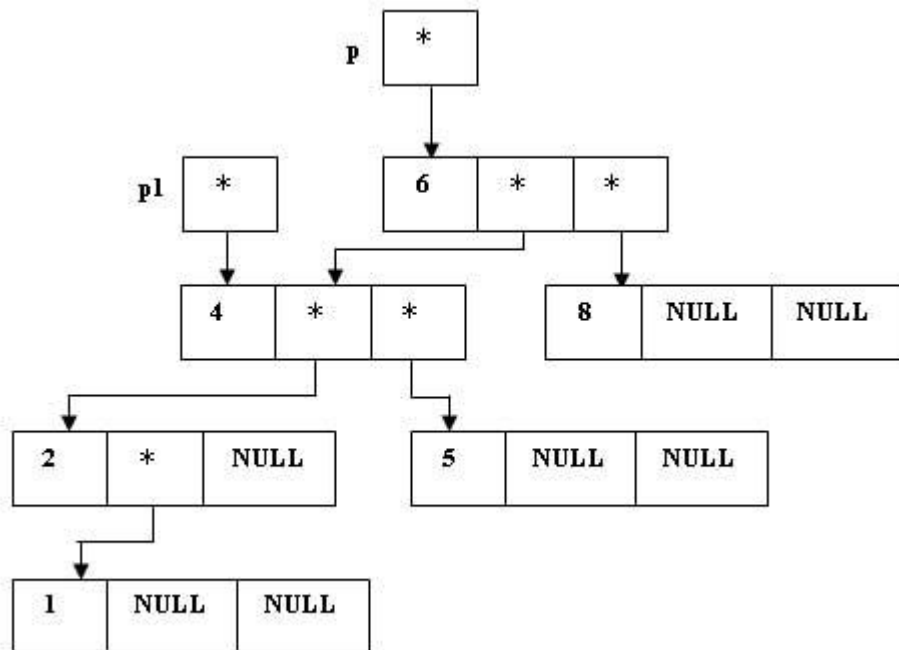
- * `p1 = (*p).Right;`
- * `(*p).Right = (*p1).Left;`
- * `(*p1).Left = p;`
- * `(*p).bal = 0;`
- * `p = p1;`

Исходное дерево



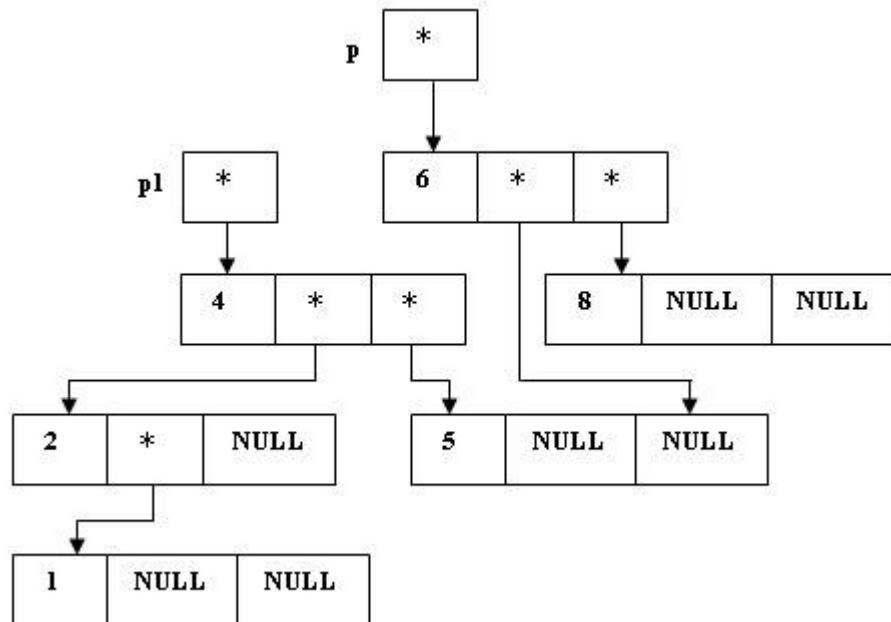
Сохранение адреса нового корня дерева

* $p1 = (*p).Left;$



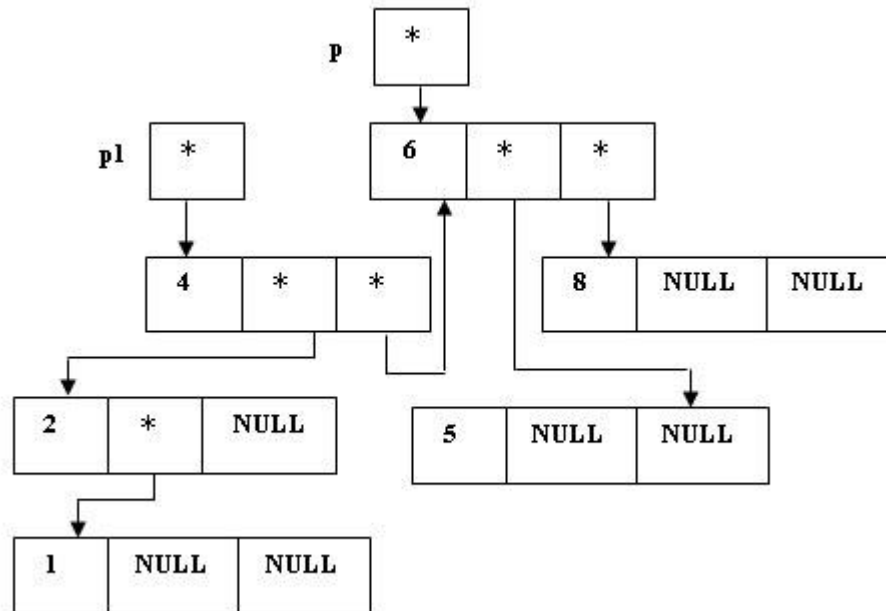
Переприкрепление

* (*p).Left = (*p1).Right;



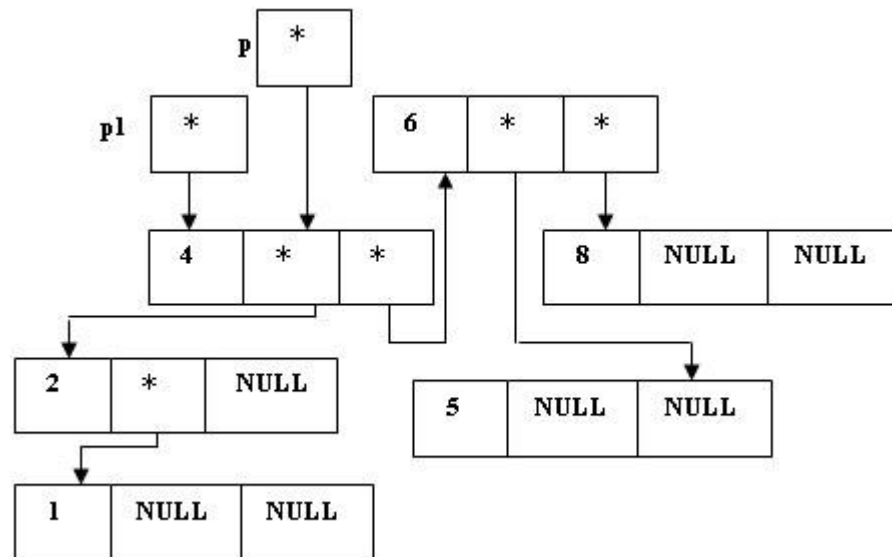
Определение правого поддерева "нового" корня

* (*p1).Right = p;

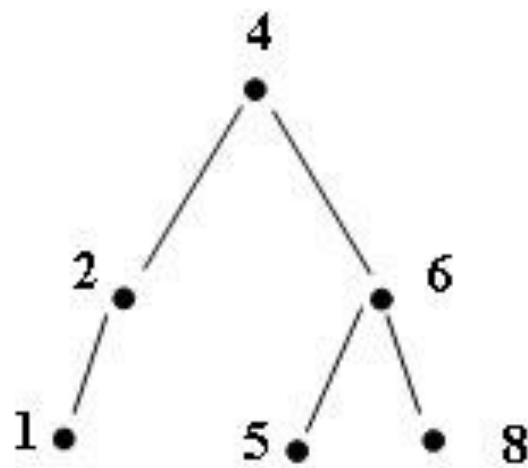


Установка начальных значений

- * $(*p).bal = 0;$
- * $p = p1;$
- *



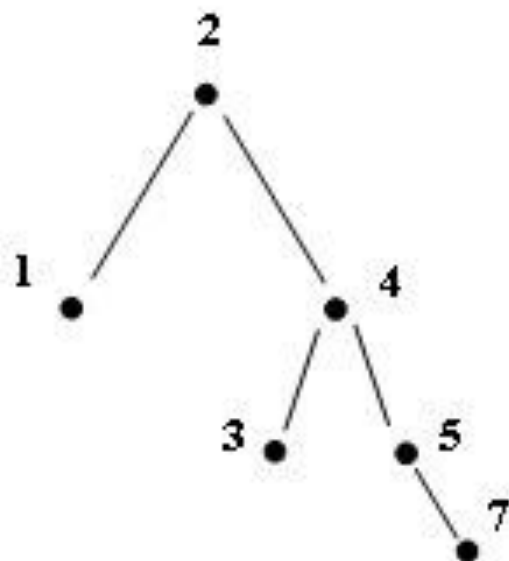
Результирующее дерево



Однократный RR-поворот

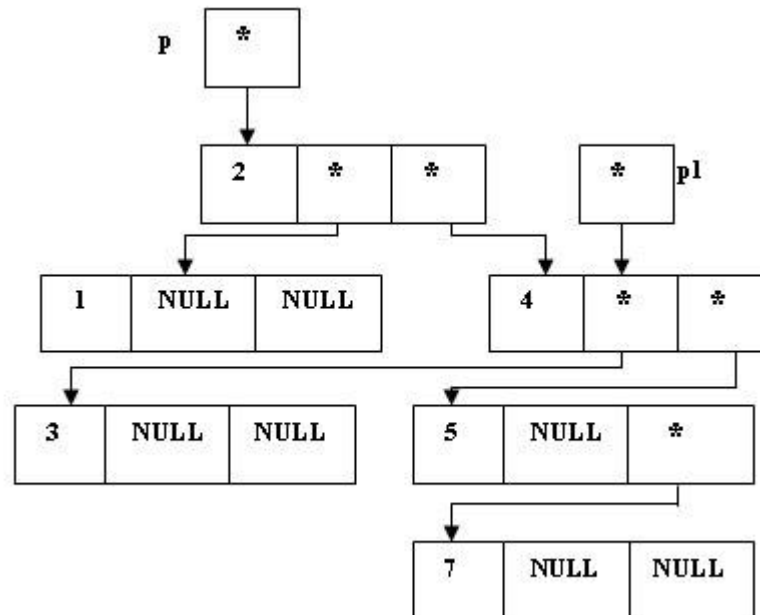
- * `p1 = (*p).Right;`
- * `(*p).Right = (*p1).Left;`
- * `(*p1).Left = p;`
- * `(*p).bal = 0; p = p1;`

Исходное дерево



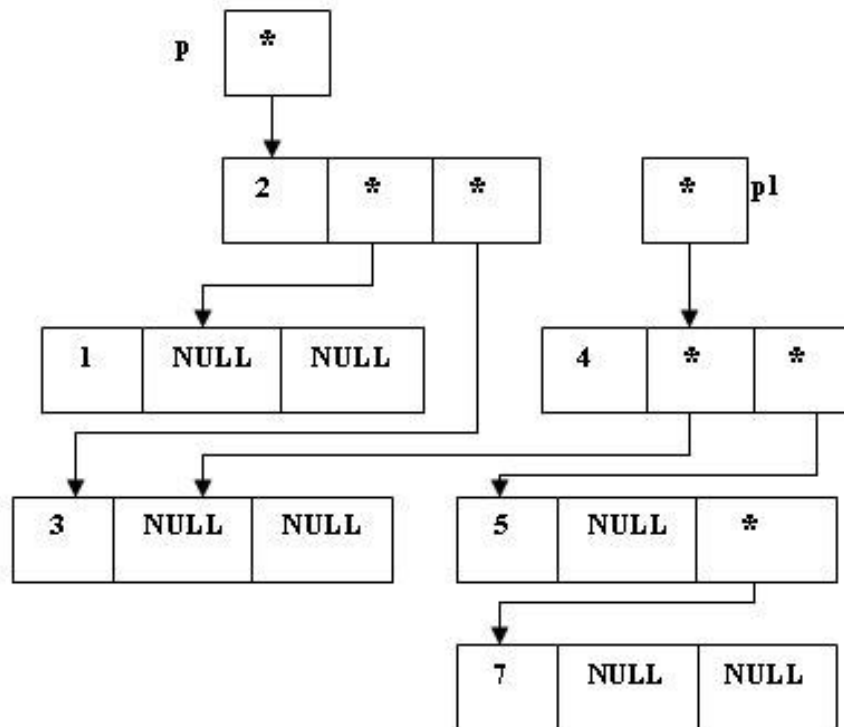
Сохранение адреса нового корня дерева

* $p1 = (*p).Right;$



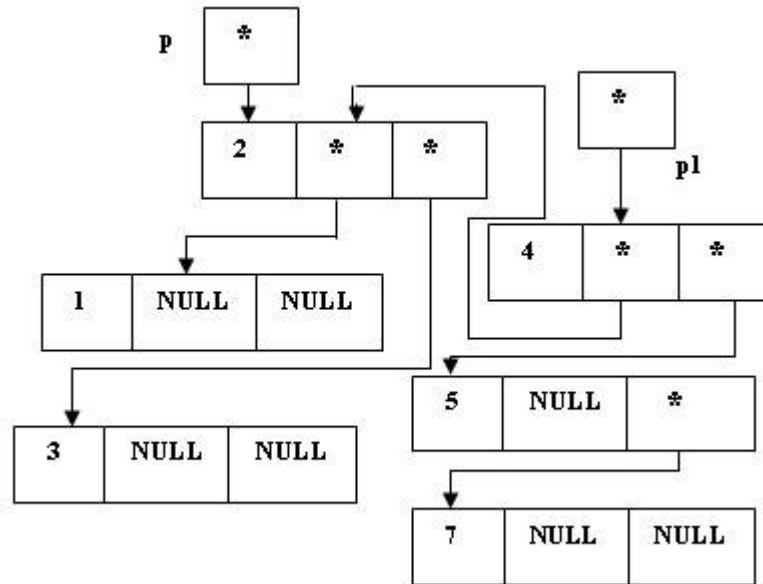
Переприкрепление

* (*p).Right = (*p1).Left;



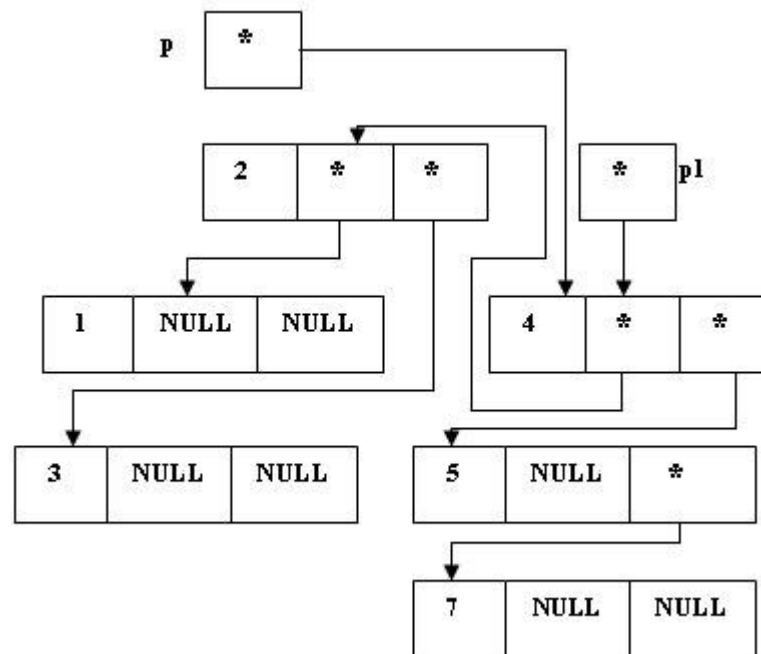
Определение левого поддерева "нового" корня

* (*p1).Left = p;

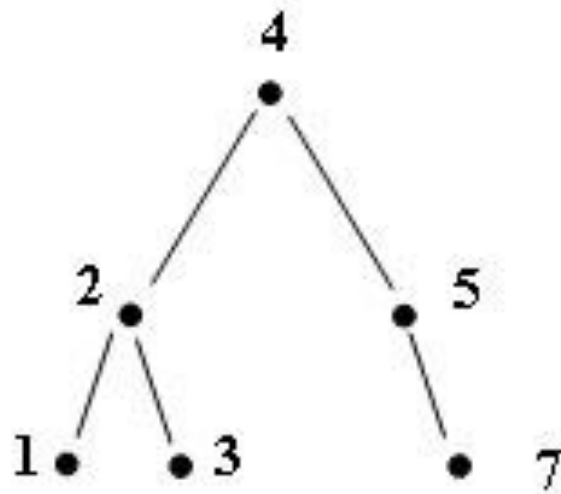



Установка начальных значений

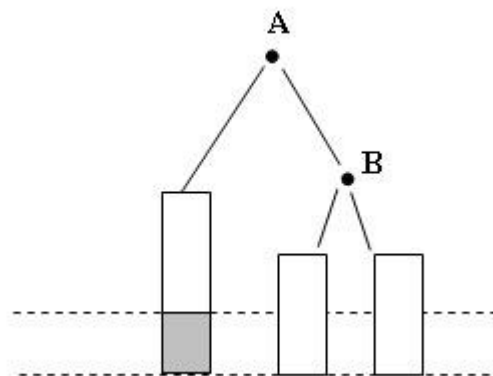
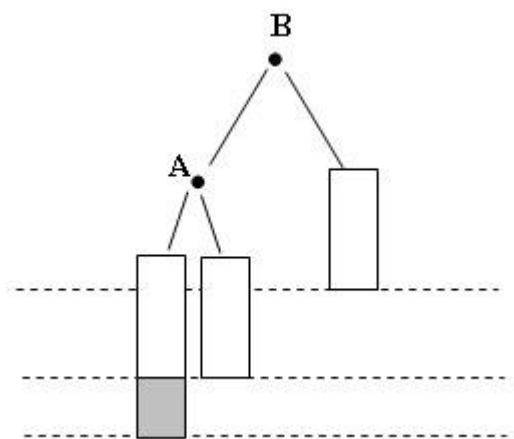
* (*p).bal = 0; p = p1;



Результирующее дерево



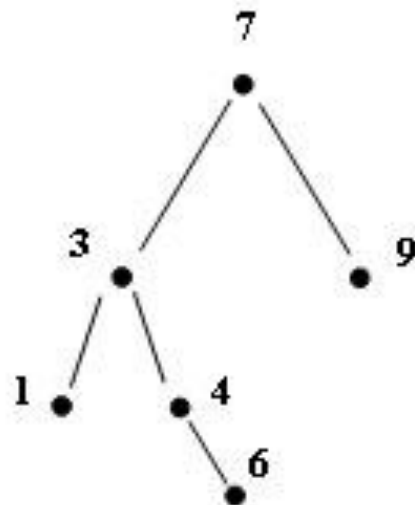
- 
- * Если после вставки показатели сбалансированности вершин имеют одинаковый знак и отличаются только на единицу, то восстановить баланс дерева можно **однократным поворотом** (включая одно "переприкрепление" поддеревя), при этом вставка не будет оказывать влияния на другие участки дерева.



Двукратный LR-поворот

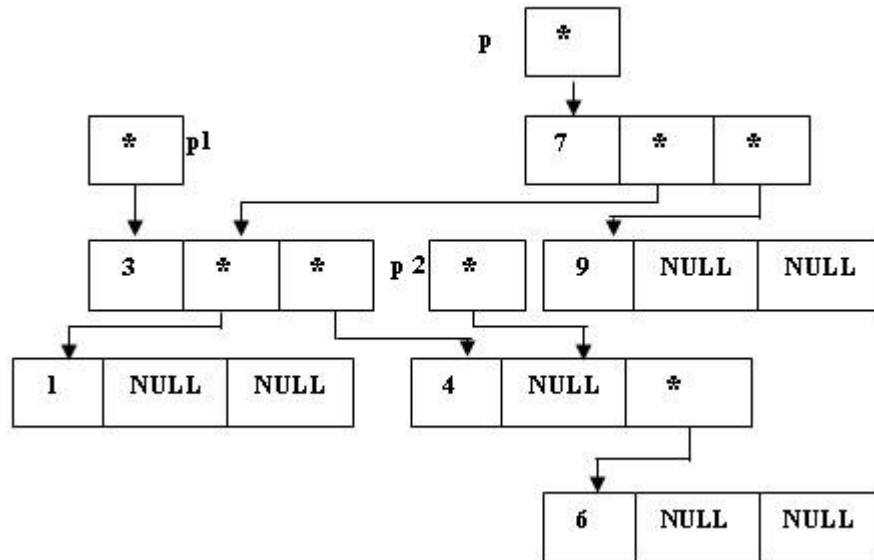
- * $p1 = (*p).Left;$
- * $p2 = (*p1).Right;$
- * $(*p1).Right = (*p2).Left;$
- * $(*p2).Left = p1;$
- * $(*p).Left = (*p2).Right;$
- * $(*p2).Right = *p;$
- * $p = p2;$

Исходное дерево



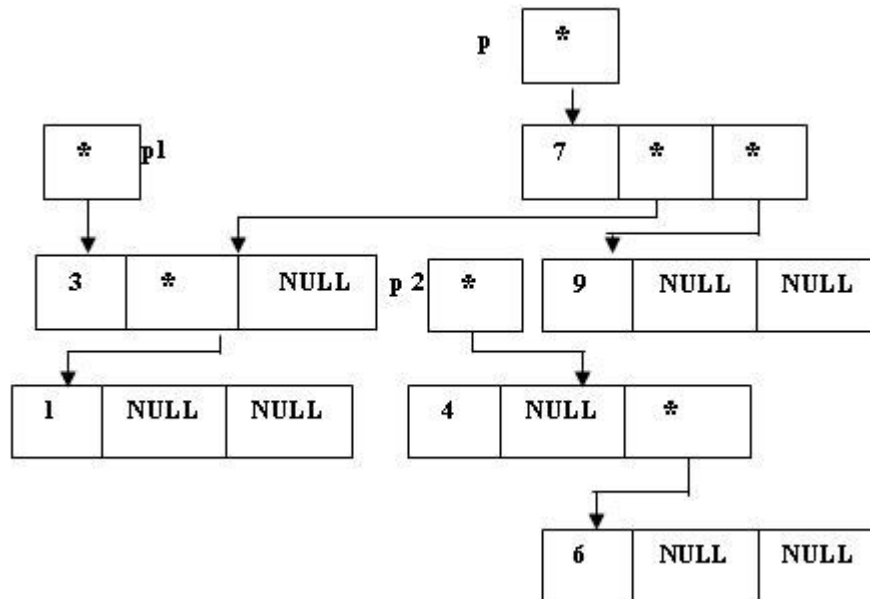
Определение p1 и p2

* $p1 = (*p).Left; p2 = (*p1).Right;$



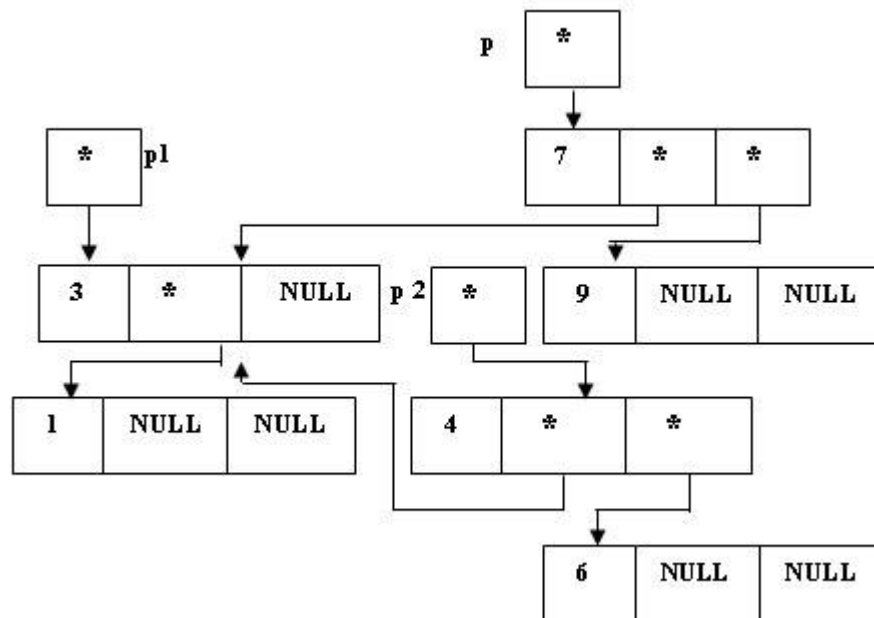
Переприкрепление

* (*p1).Right = (*p2).Left;



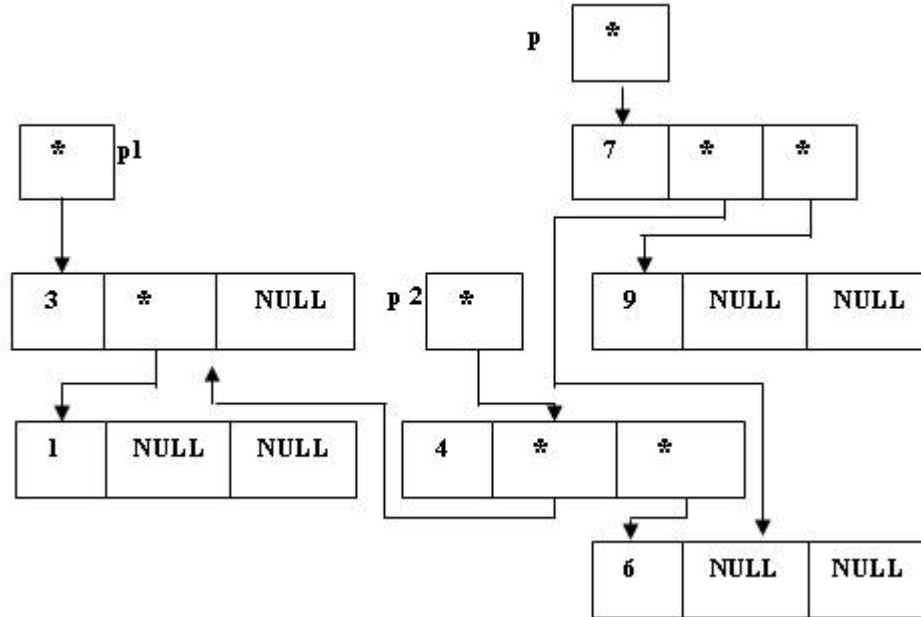
Определение левого поддерева "нового" корня

* (*p2).Left = p1;



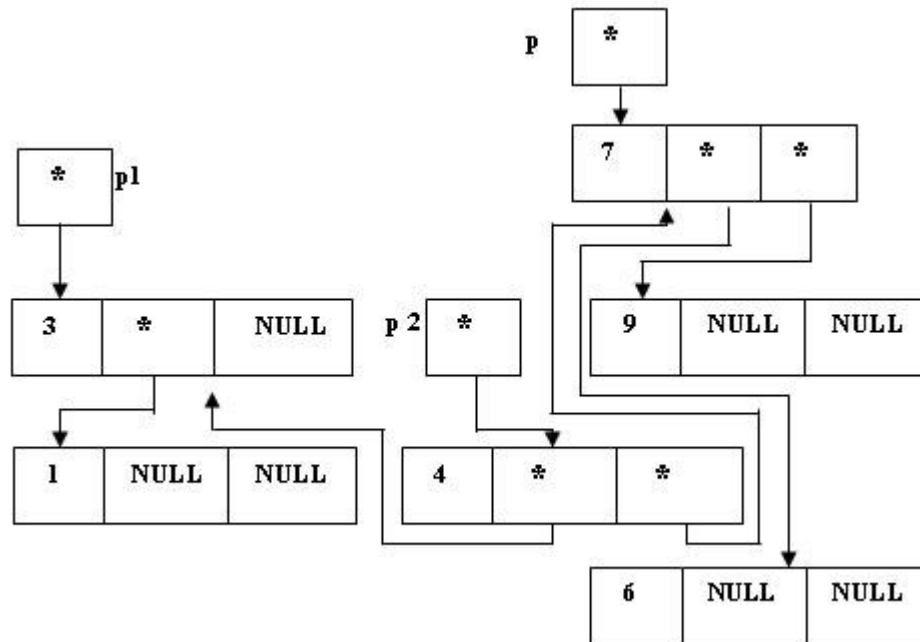
Переприкрепление

* (*p).Left = (*p2).Right;



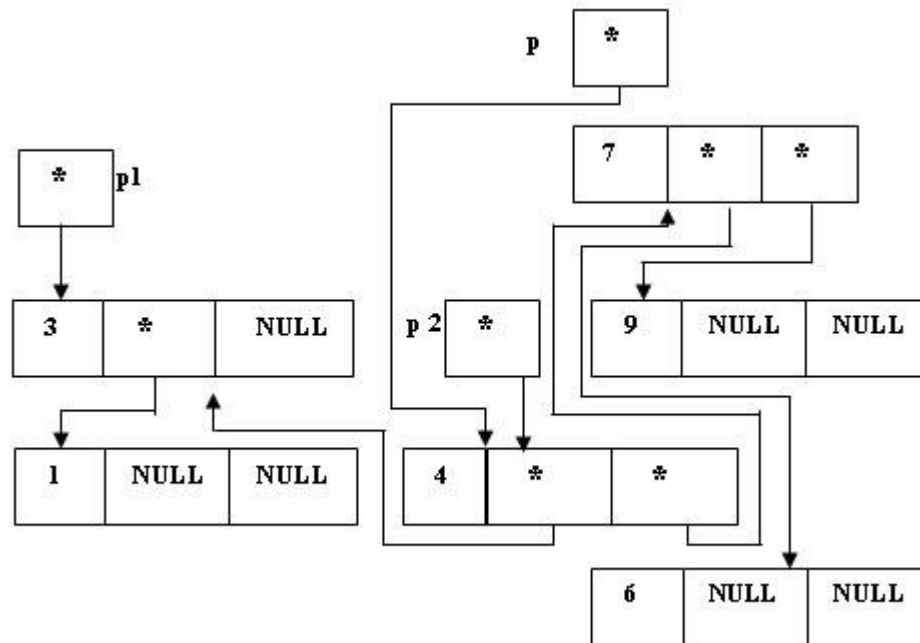
Определение правого поддерева "нового" корня

* (*p2).Right = *p;

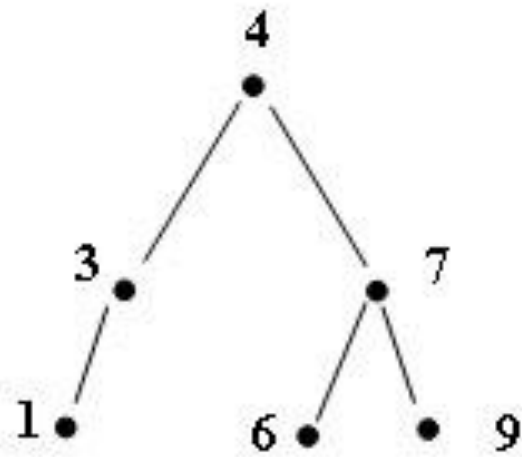


Установка начальных значений

* $p = p2;$



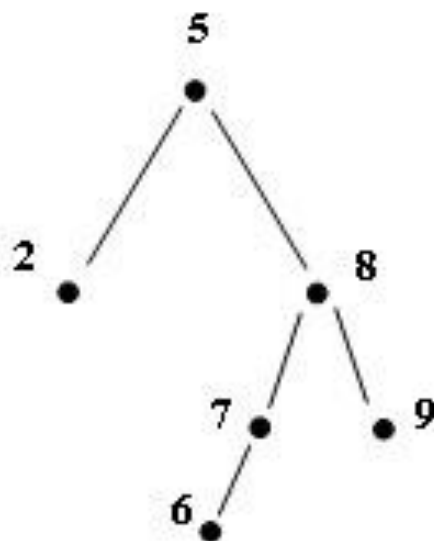
Результирующее дерево



Двукратный RL-поворот

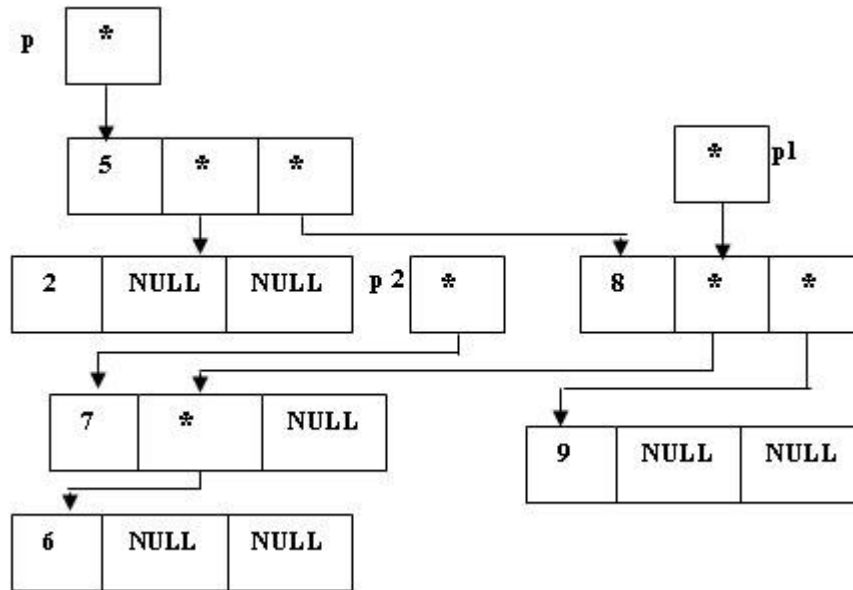
- * $p1 = (*p).Right; p2 = (*p1).Left;$
- * $(*p1).Left = (*p2).Right; (*p2).Right = p1;$
- * $(*p).Right = (*p2).Left; (*p2).Left = *p;$
- * $p = p2;$

Исходное дерево



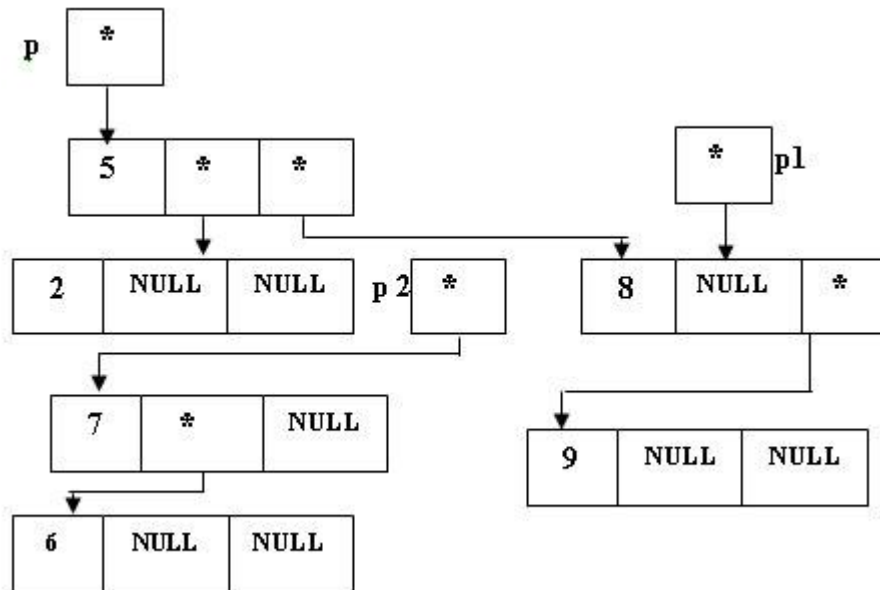
Определение p1 и p2

* $p1 = (*p).Right; p2 = (*p1).Left;$



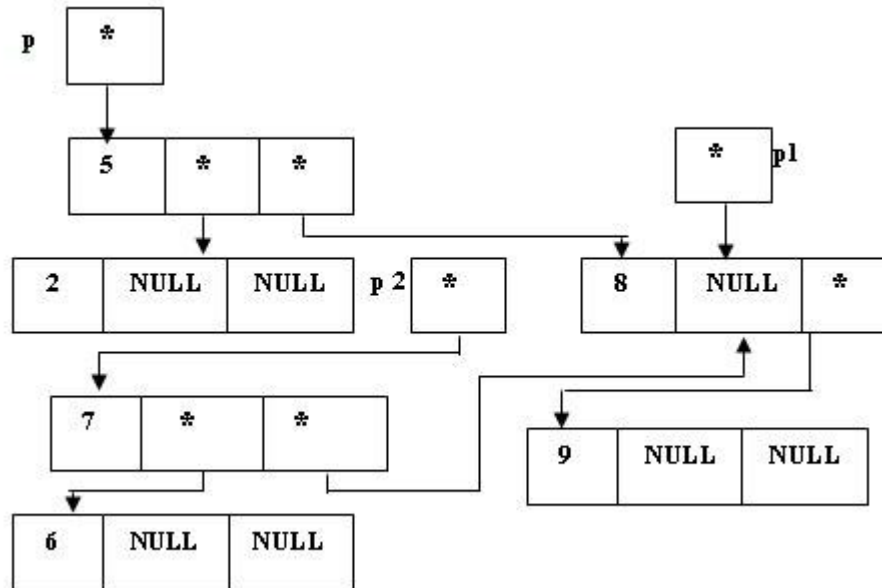
Переприкрепление

* (*p1).Left = (*p2).Right;



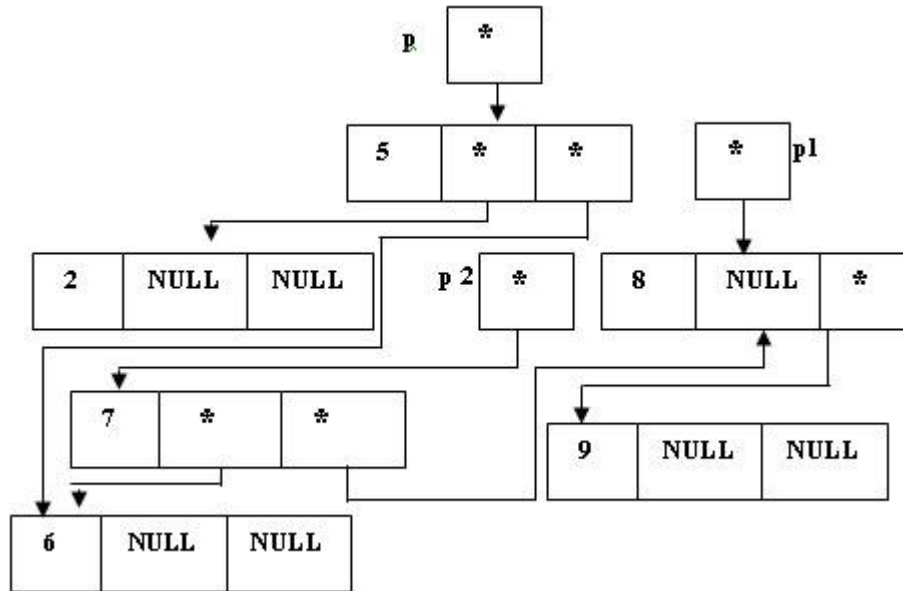
Определение правого поддерева "нового" корня

* (*p2). Right = p1;



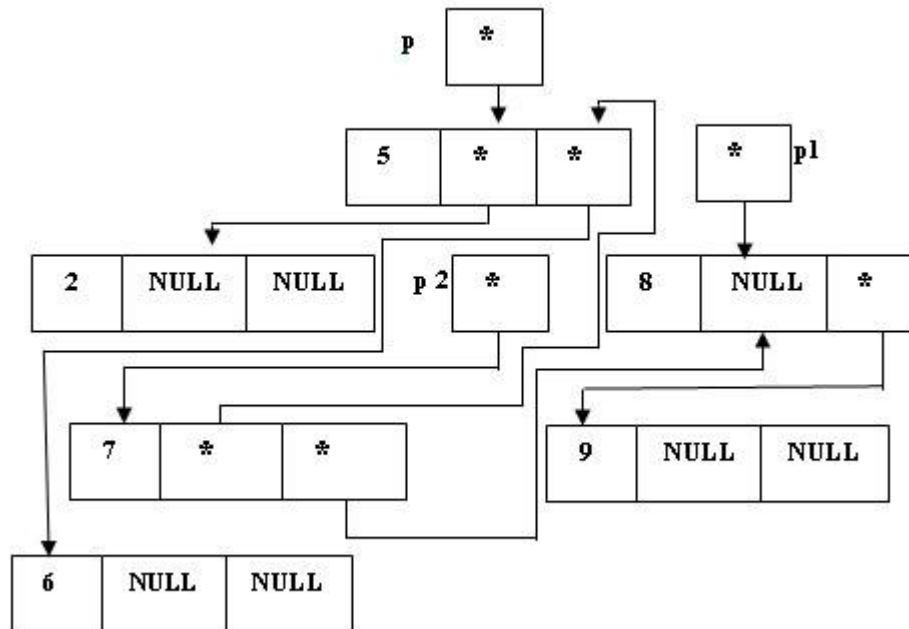
Переприкрепление

* (*p).Right = (*p2). Left;



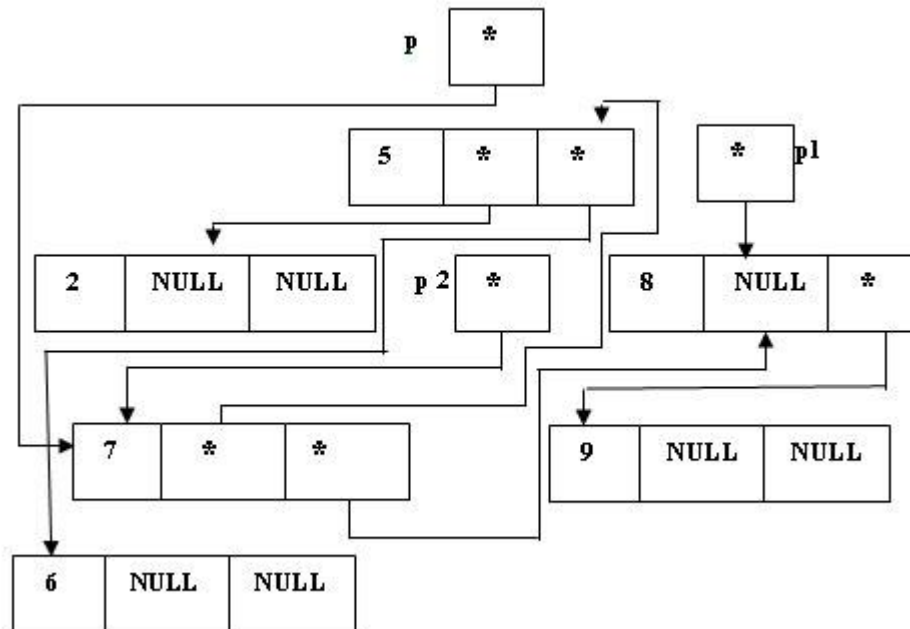
Определение правого поддерева "нового" корня

* (*p2). Left = *p;

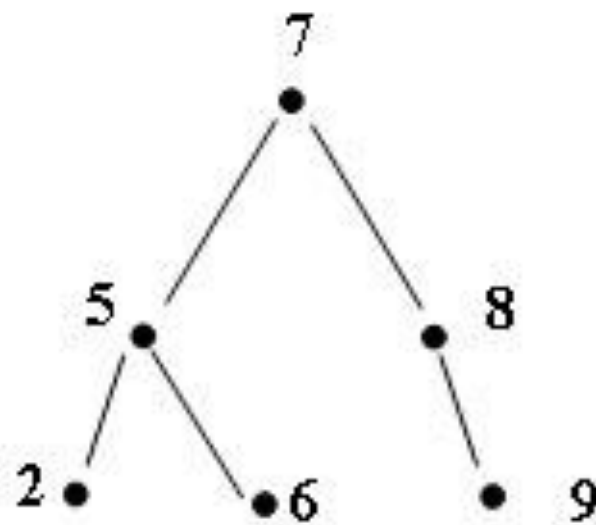



Установка начальных значений

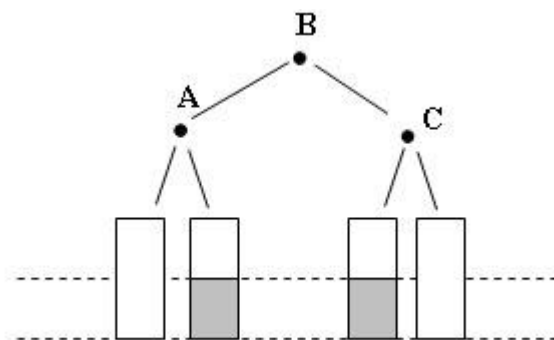
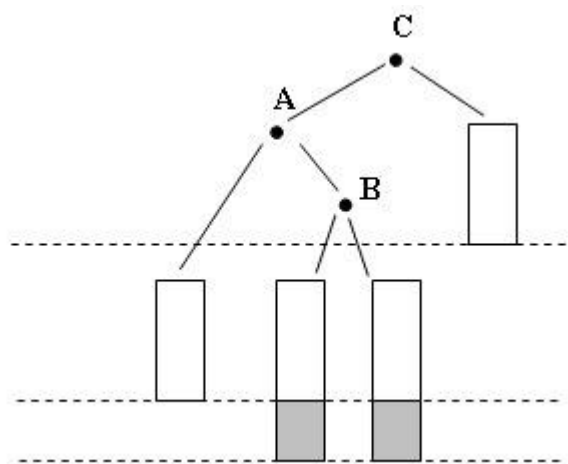
* $p = p2;$



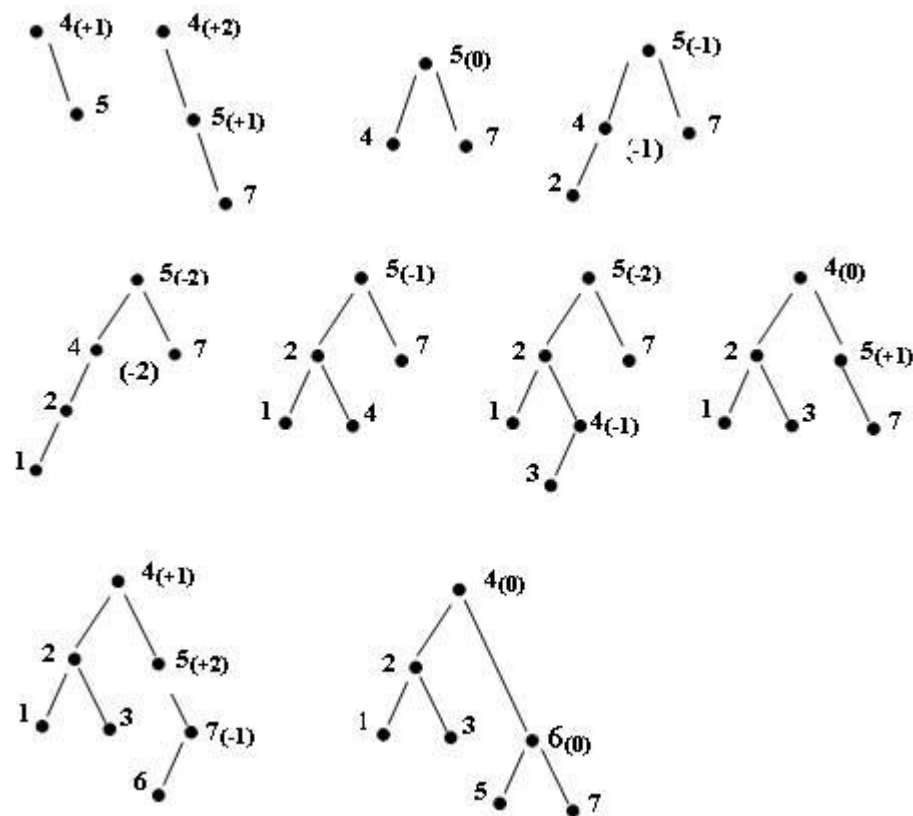
Результирующее дерево




- 
- * Если после вставки показатели сбалансированности имеют разный знак, то можно восстановить баланс дерева **двухкратными поворотами трех вершин**. В этом случае вставка также не оказывает влияния на другие участки дерева



Построение AVL дерева



- 
- * <http://neerc.ifmo.ru/wiki/index.php?title=%D0%90%D0%92%D0%9B-%D0%B4%D0%B5%D1%80%D0%B5%D0%B2%D0%BE>
 - * <http://www.proteus2001.narod.ru/gen/txt/6/avl.html>
 - * <http://habrahabr.ru/post/150732/>