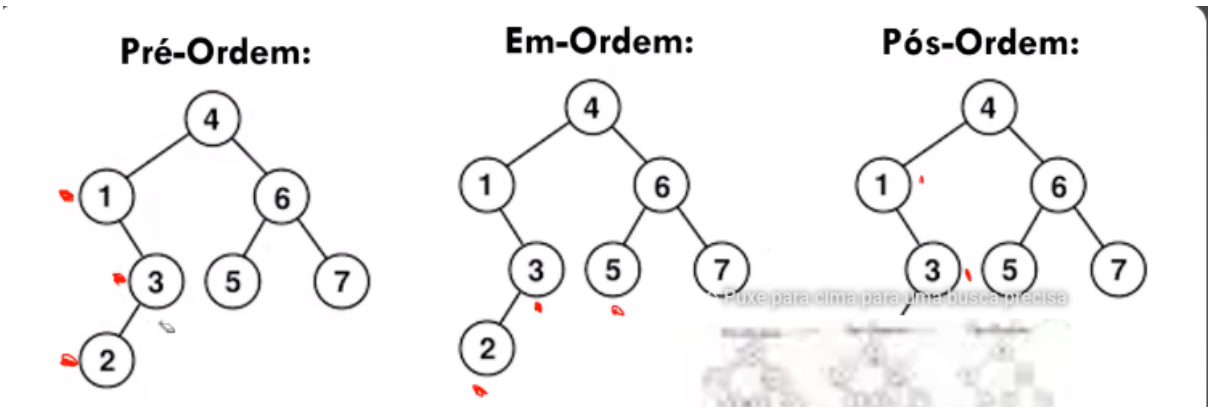
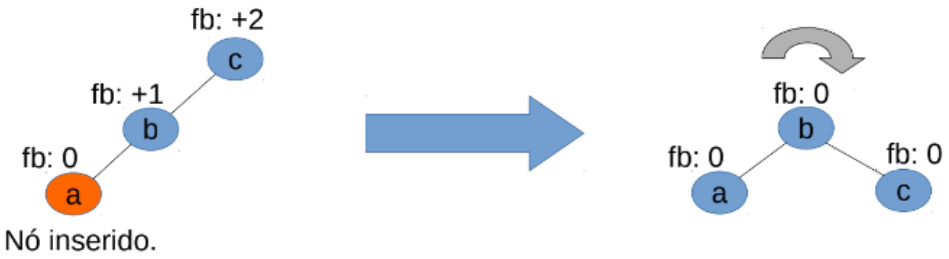


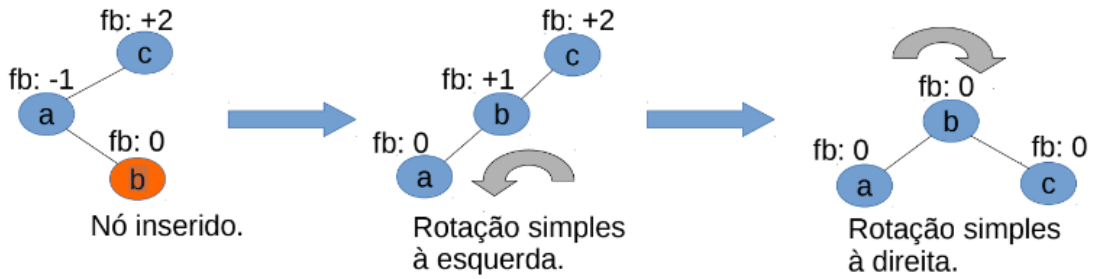
Rotações – simples à direita

- Se faz necessária quando o desbalanceamento ocorreu em função de inserções feitas à esquerda:



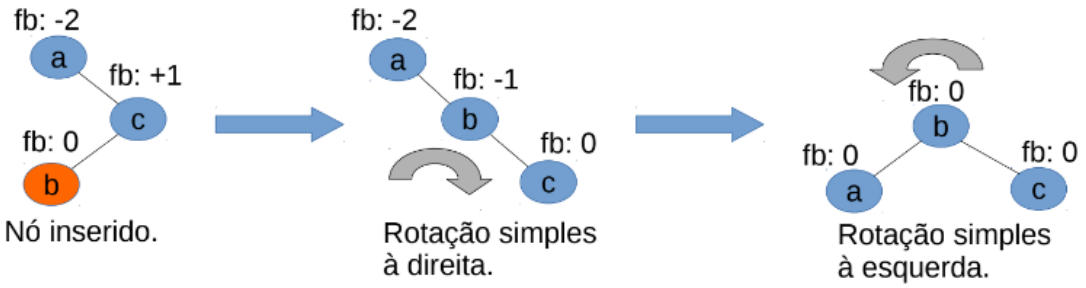
Rotaçoes – dupla a direita

- Se faz necessária quando o desbalanceamento ocorreu em função de inserção feita à direita na subárvore esquerda:



Rotações – dupla à esquerda

- Se faz necessária quando o desbalanceamento ocorreu em função de inserção feita à esquerda na subárvore direita:



Regras

- Todo nó é rubro ou negro;
- Raiz é sempre negra;
- Novo nó é sempre rubro;
- Todo caminho da raiz até algum nó folha terá o mesmo número de nós negros;
- Nenhum caminho pode ter dois nós rubros consecutivos;
- NULL é considerado negro;

- Operações:
  - Tio negro: rotaciona;
  - Tio rubro: troca cores;

- Após rotacionar:

- Após trocar cores:

```
// Inserts a new key 'k'
void MinHeap::insertKey(int k)
{
    if (heap_size == capacity)
    {
        cout << "\nOverflow: Could not insertKey\n";
        return;
    }

    // First insert the new key at the end
    heap_size++;
    int i = heap_size - 1;
    harr[i] = k;

    // Fix the min heap property if it is violated
    while (i != 0 && harr[parent(i)] > harr[i])
    {
       swap(&harr[i], &harr[parent(i)]);
       i = parent(i);
    }
}
```

	0	1	2	3	4	5	6
vet	10	15	30	40	50	100	40

Para o i-ésimo nó:

- vet[(i-1)/2] – retornará o nó pai

- vet[(2*i)+1] – retornará o filho a esquerda

- vet[(2*i)+2] – retornará o filho a direita

https://www.inf.ufsc.br/~aldo.vw/estruturas/simulador/RB.html

https://www.cs.usfca.edu/~galles/visualization/Heap.html