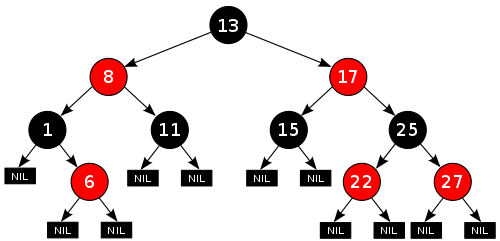
# Tarea 4 – Arboles Binarios

## Red-Black Tree

Un árbol rojo-negro es un tipo de árbol binario de búsqueda que intenta mantener su altura o el número de niveles de nodos bajo la raíz tan pequeños como sea posible. Mantener baja la altura se consigue realizando transformaciones en el árbol con acciones como la rotación y la inserción de árbol. Este tipo de árbol contiene las siguientes características:

* Un nodo es rojo o negro.
* La raíz siempre es negra.
* Todas las hojas “NULL” son negras.
* Un nodo rojo siempre tendrá hijos negros.
* El número de nodos negros es el mismo en cualquier camino que vaya de la raíz a la hoja.



Cuando los nodos son removidos o borrados, el árbol debe ser transformado para mantener estas propiedades. Esto se hace repintando o rodando los nodos.

Los nuevos nodos se insertan normalmente con el color rojo. Si el nodo padre es negro, el árbol todavía es válido. Si el nodo padre es rojo y existe un nodo tío rojo, entonces ellos deben ser repintados de negro, y el nodo abuelo debe ser repintado de rojo. Cuando queda un nodo rojo con padre rojo se efectúa una rotación. Si la raíz termina roja, esta debe ser repintada de negro.

## Splay Tree

Un árbol biselado es un árbol binario de búsqueda auto-balanceable, con la propiedad adicional de que a los elementos accedidos recientemente se accederá más rápidamente en accesos posteriores. Realiza operaciones básicas como la inserción, la búsqueda y el borrado en un tiempo del orden de O(log n). el árbol biselado se comporta mejor que otros árboles de búsqueda. La operación de biselacion que este realiza consiste en reorganizar el árbol para un cierto elemento, colocándolo en la raíz. Una manera de hacerlo es realizando primero una búsqueda binaria en el árbol para encontrar el elemento en cuestión y, a continuación, usar rotaciones de árboles de una manera específica para traer el elemento a la cima. Alternativamente, un algoritmo "de arriba a abajo" puede combinar la búsqueda y la reorganización del árbol en una sola fase.

El buen rendimiento de un [árbol](https://www.ecured.cu/%C3%81rbol) biselado depende del hecho de que es auto-balanceado, y además se optimiza automáticamente. Los [nodos](https://www.ecured.cu/Nodo) accedidos con más frecuencia se moverán cerca de la [raíz](https://www.ecured.cu/Ra%C3%ADz) donde podrán ser accedidos más rápidamente. Esto es una ventaja para casi todas las aplicaciones, y es particularmente útil para implementar cachés y algoritmos de recolección de basura; sin embargo, es importante apuntar que para un acceso uniforme, el rendimiento de un árbol biselado será considerablemente peor que un [árbol](https://www.ecured.cu/%C3%81rbol) de búsqueda binaria balanceado simple.

Los árboles biselados también tienen la ventaja de ser consideradamente más simples de implementar que otros árboles binarios de búsqueda auto-balanceados, como pueden ser los árboles Rojo-Negro o los árboles AVL, mientras que su rendimiento en el caso promedio es igual de eficiente. Además, los árboles biselados no necesitan almacenar ninguna otra información adicional a parte de los propios datos, minimizando de este modo los requerimientos de memoria. Sin embargo, estas estructuras de datos adicionales proporcionan garantías para el peor caso, y pueden ser más eficientes en la práctica para el acceso uniforme.

