|  |
| --- |
| **2TAD RB TREE** |
| Atributos o descripción. |
| {inv: } //Aquí van las propiedades |
| Operaciones básicas   * createRBTree * getMin * getMax * insert * delete * modify * search * rightRotate * leftRotate |

Operaciones

|  |
| --- |
| **Operacion**  “Descripcion”  Pre:  Post: |
|  |
|  |

**Diseños de casos de pruebas unitarias**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba 1: Verifica que el método añade inserta correctamente un nodo al árbol manteniendo las propiedades. | | | | |
| Clase | Método | Escenario | Entrada | Resultado |
| RBTree | +insert (K, V): void | Hay un árbol rojinegro vacío | K=10  V=1 | La raíz es negra |
| RBTree | +insert (K, V): void | Hay un árbol rojinegro con un nodo: Key=10, value=1 | K=5  V=2 | La raíz es negra  El hijo izquierdo de la raíz es rojo. |
| RBTree | +insert (K, V): void | Hay un árbol rojinegro con los siguientes nodos:  Key=10 value=1  Key=5 value=2 | K=15  V=3 | La raíz es negra y sus dos hijos son rojos. |
| RBTree | +insert (K, V): void | El mismo que el anterior más un nodo con key=15 y value=3 | K=4  V=4 | La raíz es negra y sus dos hijos son negros. El hijo izquierdo el hijo izquierdo de la raíz es rojo. |
| RBTree | +insert (K, V): void | El mismo que el anterior más un nodo con key=4 y value=4 | K=4  V=5 | La raíz es negra. Sus dos hijos son negros y tiene key 4 y 15 respectivamente. Los dos hijos del hijo izquierdo de la raíz son rojos y tienen key 4 y 5 respectivamente. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Prueba 2: Verifica que el método delete elimina correctamente un nodo al árbol manteniendo las propiedades. | | | | |
| Clase | Método | Escenario | Entrada | Resultado |
| RBTree | +delete(K): V | Hay un árbol rojinegro vacío | K=10 | Retorna null.  El árbol sigue estando vacío |
| RBTree | +delete(K): V | Hay un árbol rojinegro con un nodo: Key=10, value=1 | K=10 | Retorna 1  El árbol está vacío. |
| RBTree | +delete(K): V | Hay un el siguiente árbol rojinegro:    Con los siguientes Values:  V(11)=1  V(2)=2  V(1)=3  V(7)=4  V(5)=5  V(8)=6  V(14)=7  V(15)=8 | K=5 | Retorna 5  No ocurre ninguna rotación, los colores permanecen estables. El hijo izquierdo de 7 es NIL. |
| RBTree | +delete(K): V | El mismo que el anterior | K=7 | Retorna 4  8 pasa a ser el hijo derecho de 2 y es negro, además su hijo derecho es 5 y sigue siendo rojo. |
| RBTree | +delete(K): V | El mismo del anterior | K=20 | Retorna null. |
| RBTree | +delete(K): V | Se tiene el siguiente árbol rojinegro    Con los siguiente values:  V(8)=1  V(7)=2 y su key ahora va a ser 6  V(12)=3  V(6)=4 y su key ahora va a ser 5  V(7.5)=5 y su key ahora va a ser 7  V(10)=6  V(14)=7  V(15)=8  V(13)=9 | K=10 | Retorna 6.  La raiz es negra y es 8  Su hijo derecho es 14 y es negro.  El hijo izquierdo de 14 es 12 y es negro y su hijo derecho es 13 y es rojo.  El hijo derecho de 14 es 15 y es negro. |
| RBTree | +delete(K): V | Existe el siguiente árbol rojinegro:    Con los siguiente Values:  V(32)=1  V(21)=2  V(64)=3  V(15)=4  V(75)=5 | K=21 | Retorna 2  La raiz es 64 y sus dos hijos son negros y son 32 y 75 respectivamente.  El hijo izquierdo de 32 es 15 y es rojo. |
| RBTree | +delete(K): V | El mismo que el anterior | K=64 | Retorna 3  La raiz es 21 y es negra, su hijo derecho es 15 y es negro, y su hijo izquierdo es 32 y es negro, y su hijo derecho es 75 y es rojo. |