|  |  |
| --- | --- |
| Prueba No: | 1. |
| Tipo de Prueba: | Unitaria Automática |
| Técnica: | JUnit, con casos de prueba implementados como atributos dentro de las clases de pruebas |
| Método a Probar: | insertarNodo(objetoAct: T) |
| Descripción de Entradas: | Elemento que se va a agregar al árbol |
| Descripción de Salidas: | El elemento se agrega al árbol en la posición que corresponda. En caso de ser necesario, el árbol se modifica hasta cumplir todas las propiedades de un árbol rojinegro. |

|  |  |
| --- | --- |
| Caso de Prueba No: | 1.1. |
| Objetivo: | Probar que el método agrega correctamente un nodo cuando se encuentra vacío, y que se elemento es la raíz, la cual es negra. |
| Escenario: | Un árbol vacío |
| Entrada(s): | “D” |
| Salida Esperada: | El árbol tiene un elemento el cual es la raíz, que tiene por valor “D” y por color negro. |

|  |  |
| --- | --- |
| Caso de Prueba No: | 1.2. |
| Objetivo: | Probar que el método agrega correctamente cuando el árbol tiene elementos, y que no hace modificaciones cuando cumple todas las reglas del árbol rojinegro |
| Escenario: |  |
| Entrada(s): | “C” |
| Salida Esperada: |  |

|  |  |
| --- | --- |
| Caso de Prueba No: | 1.3. |
| Objetivo: | Probar que el método agrega correctamente cuando el árbol tiene elementos, y que modifica el árbol cuando es necesario y tanto el elemento nuevo como el padre son ambos hijos izquierdos (o derechos) |
| Escenario: | C:\Users\WINDOWS 10\AppData\Local\Microsoft\Windows\INetCache\Content.Word\escenario1.png |
| Entrada(s): | “0” |
| Salida Esperada: |  |

|  |  |
| --- | --- |
| Caso de Prueba No: | 1.4. |
| Objetivo: | Probar que el método agrega correctamente cuando el árbol tiene elementos, y que modifica el árbol cuando es necesario y el elemento nuevo es hijo derecho pero el padre es izquierdo (o al contrario) |
| Escenario: |  |
| Entrada(s): | “B” |
| Salida Esperada: |  |

|  |  |
| --- | --- |
| Caso de Prueba No: | 1.5. |
| Objetivo: | Probar que el método agrega correctamente cuando el árbol tiene elementos, y que modifica el árbol cuando es necesario y tanto el padre como el tío del nuevo elemento son rojos (o al contrario) |
| Escenario: |  |
| Entrada(s): | “0” |
| Salida Esperada: |  |

|  |  |
| --- | --- |
| Prueba No: | 2. |
| Tipo de Prueba: | Unitaria Automática |
| Técnica: | JUnit, con casos de prueba implementados como atributos dentro de las clases de pruebas |
| Método a Probar: | buscarNodo(objetoAct: T) |
| Descripción de Entradas: | Elemento que se va a buscar en el árbol |
| Descripción de Salidas: | El método retorna un ArrayList con todos los elementos que coinciden con T, o null si ninguno lo hace. |

|  |  |
| --- | --- |
| Caso de Prueba No: | 2.1. |
| Objetivo: | Probar que el método retorna correctamente cuando se busca la raíz |
| Escenario: | C:\Users\WINDOWS 10\AppData\Local\Microsoft\Windows\INetCache\Content.Word\escenario1.png |
| Entrada(s): | “D” |
| Salida Esperada: | {“D} |

|  |  |
| --- | --- |
| Caso de Prueba No: | 2.2. |
| Objetivo: | Probar que el método null cuando el elemento no está en el árbol |
| Escenario: | C:\Users\WINDOWS 10\AppData\Local\Microsoft\Windows\INetCache\Content.Word\escenario1.png |
| Entrada(s): | “0” |
| Salida Esperada: | null |

|  |  |
| --- | --- |
| Caso de Prueba No: | 2.3. |
| Objetivo: | Probar que el método el ArrayList cuando el elemento está varias veces repetidos en el árbol |
| Escenario: |  |
| Entrada(s): | “A” |
| Salida Esperada: | {“A”, “A”, “A”} |

|  |  |
| --- | --- |
| Prueba No: | 3. |
| Tipo de Prueba: | Unitaria Automática |
| Técnica: | JUnit, con casos de prueba implementados como atributos dentro de las clases de pruebas |
| Método a Probar: | eliminarNodo(objetoAct: T) |
| Descripción de Entradas: | Elemento que se va a eliminar en el árbol |
| Descripción de Salidas: | El elimina el nodo con el valor correspondiente al pasado por parámetro. |

|  |  |
| --- | --- |
| Caso de Prueba No: | 3.1. |
| Objetivo: | Probar que el método el elimina correctamente un elemento y no modifica el árbol si no es necesario |
| Escenario: | C:\Users\WINDOWS 10\AppData\Local\Microsoft\Windows\INetCache\Content.Word\escenario1.png |
| Entrada(s): | “A” |
| Salida Esperada: | C:\Users\WINDOWS 10\AppData\Local\Microsoft\Windows\INetCache\Content.Word\escenarioEsperado1-3.png |

|  |  |
| --- | --- |
| Caso de Prueba No: | 3.2. |
| Objetivo: | Probar que el método elimina correctamente el nodo que se le pasa, y modifica el árbol si es necesario |
| Escenario: | C:\Users\WINDOWS 10\AppData\Local\Microsoft\Windows\INetCache\Content.Word\escenario1.png |
| Entrada(s): | “D” |
| Salida Esperada: |  |

|  |  |
| --- | --- |
| Caso de Prueba No: | 3.3. |
| Objetivo: | Probar que el método tira excepción cuando se intenta eliminar un nodo que no existe en el árbol |
| Escenario: | C:\Users\WINDOWS 10\AppData\Local\Microsoft\Windows\INetCache\Content.Word\escenario1.png |
| Entrada(s): | “K” |
| Salida Esperada: | Excepción |