# ALGORITHMS AND DATA STRUCTURES

MEMBERS:

Daniel Ramirez

Brian Stiven Romero Restrepo

Sebastian Navia Ramirez - A00369304.

UNIVERSITY ICESI

2021-2

# Content

Functional requirements ..... 3

BBT unit Test ..... 3

BST unite Test ..... 5

Functional requirements

RF1 Manage a basketball player

RF1.1 Add a basketball player with name, age, team he belongs to and 5 statistics

RF1.2 Modify any parameter of a basketball player

RF1.3 Eliminate any basketball player

RF1.4 Search ABBs for basketball players based on any of their parameters

RF2 Improve the effectiveness of player searches by parameters using ABBs

RF3 Import csv files that will contain basketball players, this will be another additional means to add players

RF4 Show the time it takes to search for players based on some parameter

RF5 Show the status of the program at all times using a graphical interface for this process

BBT unit Test

| Name | Class | Scenary |
|---|---|---|
| setupStage1 | BBT | An empty BBT type object is created |

**Test Objective:** verify the correct addition of objects to the BBT

| Class | Method | Scenary | Input | Output |
|---|---|---|---|---|
| BBT | insert | setupScenary 1 | new BBT(node: “a” ," key: 3);<br>new BBT(node: “b” ," key: 2);<br>new BBT(node: “c” ," key: 8); | The node “a”,”b” and “c” were correctly added |

**Test Objective:** verify the correct deletion of a node in BBT

| Class | Method | Scenary | Input | Output |
|---|---|---|---|---|
| BBT | delete | setupScenary 1 | new BBT(node: “a” ," key: 3);<br>new BBT(node: “b” ," key: 2);<br>new BBT(node: “c” ," key: 8);<br>delete(key : 8); | The node “c” with key 8 was deleted, now that node is equals to null and no belong to the BBT |

**Test Objective:** verify the correct deletion of a node with one child in BBT

| Class | Method | Scenary | Input | Output |
|---|---|---|---|---|
| BBT | delete | setupScenary 1 | new BBT(node: “a” ," key: 3);<br>new BBT(node: “b” ," key: 2);<br>new BBT(node: “c” ," key: 8);<br>new BBT(node: “d” ," key: 5);<br>delete(key : 8); | The node “c” with key 8 was deleted, now that node is equals to null and no belong to the BBT and now the node “a” has a new right child, the node “5” |

| | | | | |
|---|---|---|---|---|
| | | | | |

**Test Objective:** verify the correct deletion of a node with two children in BBT

| Class | Method | Scenary | Input | Output |
|---|---|---|---|---|
| BBT | delete | setupScenary 1 | new BBT(node: “a” ," key: 3);<br>new BBT(node: “b” ," key: 2);<br>new BBT(node: “c” ," key: 8);<br>new BBT(node: “d” ," key: 5);<br>new BBT(node: “d” ," key: 10);<br>delete(key : 8); | The node “c” with key 8 was deleted, now that node is equals to null and no belong to the BBT |

BST unite Test

| Name | Class | Scenary |
|---|---|---|
| setupStage1 | BST | An empty BST type object is created |

**Test Objective:** verify the correct addition of objects to the BST

| Class | Method | Scenary | Input | Output |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| BST | insert | setupScenary 1 | new BST(node: “a” ," key: 3);<br>new BST(node: “b” ," key: 2);<br>new BST(node: “c” ," key: 8); | The node “a”,”b” and “c” were correctly added |

**Test Objective:** verify the search of the BST

| Class | Method | Scenary | Input | Output |
|---|---|---|---|---|
| BST | search | setupScenary1 | new BST(node: “a” ," key: 3);<br>new BST(node: “b” ," key: 2);<br>new BST(node: “c” ," key: 8);<br>search(key:3) | The value associated with key three is the value "a" |

**Test Objective:** Verify that the value of the successor given an object is correct

| Class | Method | Scenary | Input | Output |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| BST | getSuccessor | setupScenary 1 | new BST(node: “a” ," key: 3);<br>new BST(node: “b” ," key: 2);<br>new BST(node: “c” ," key: 8);<br>getSuccessor(search(key:3)) | The value associated with the successor of key 3 is "c" |

**Test Objective:** Verify the minimum value of a BST

| Class | Method | Scenary | Input | Output |
|---|---|---|---|---|
| BST | getMinimum | setupScenary 1 | new BST(node: “a” ," key: 3);<br>new BST(node: “b” ," key: 2);<br>new BST(node: “c” ," key: 8);<br>getMinimum(key: 3) | The key associated with the minimum of the n elements in BST is 2 |

**Test Objective:** Verify the maximum value of a BST

| Class | Method | Scenary | Input | Output |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| BST | getMaximum | setupScenary 1 | new BST(node: “a” ," key: 3);<br>new BST(node: “b” ," key: 2);<br>new BST(node: “c” ," key: 8);<br>getMaximum (key: 3) | The key associated with the minimum of the n elements in BST is 8 |

**Test Objective:** Verify if an object is leaf or not

| Class | Method | Scenary | Input | Output |
|---|---|---|---|---|
| BST | isLeaf | setupScenary 1 | new BST(node: “a” ," key: 3);<br>new BST(node: “b” ," key: 4);<br>new BST(node: “c” ," key: 5);<br>isLeaf (key: 3)<br>isLeaf (key: 4)<br>isLeaf (key: 5) | With key 3 the return value is false, with key 4 the return value is false and with key 5 the return value is true |

**Test Objective:** Verify if the BST are empty or not

| Class | Method | Scenary | Input | Output |
|---|---|---|---|---|
| BST | isLeaf | setupScenary 1 | Tree.isEmpty(); | The return value is true |

**Test Objective:** Verify the height of the tree

| Class | Method | Scenary | Input | Output |
|---|---|---|---|---|
| BST | bstHeight | setupScenary 1 | new BST(node: “a” ," key: 1);<br>new BST(node: “b” ," key: 4);<br>new BST(node: “c” ," key: 5);<br>tree.bstHeight<br>new BST(node:”a:”key:6)<br>tree.bstHeight | The return value is 3<br>And after of adding a bst object, the return value is 4 |

**Test Objective:** verify that a leaf is properly removed

| Class | Method | Scenary | Input | Output |
|---|---|---|---|---|
| BST | delete | setupScenary 1 | new BST(node: “a” ," key: 1);<br>new BST(node: “b” ," key: 4);<br>new BST(node: “c” ," key: 5);<br>new BST(node: “f” ," key: 3);<br>tree.delete(key:5);<br>tree.search(key:5);<br>tree.search(key:4).getRight(); | The return value is null<br>The return value is null |

**Test Objective:** Verify if an object with only one link to another object is correctly removed

| Class | Method | Scenary | Input | Output |
|---|---|---|---|---|
| BST | delete | setupScenary 1 | new BST("a",5);<br>new BST ("b",7);<br>new BST ("c",6);<br>new BST ("d",2);<br>new BST ("e",4);<br>new BST ("f",1);<br>tree.delete(key:7);<br>tree.search(key:7);<br>tree.search(5).getRight().getValue(); | The return value is null<br>The return value is an object with key : 6 |

**Test Objective:** Verify if an object with only one link to another object is correctly removed

| Class | Method | Scenary | Input | Output |
|---|---|---|---|---|
| BST | delete | setupScenary 1 | new BST("a",5);<br>new BST ("b",7);<br>new BST ("c",6);<br>new BST ("d",2);<br>new BST ("e",4);<br>new BST ("f",1);<br>new BST(“g”,8);<br>tree.delete(key:7);<br>tree.search(key:7);<br>tree.search(5).getRight(); | The return value is null<br>The return value is an object with key : 6 |