**Presenting:**

**Lior Buzaglo: 314774787**

**Elad Fisher: 318882800**

Unit test for GraphAnalyser:

* For the following tests, we assume that **calculateAlphaCharacteristic** & **calculateDiameter** are working properly.

|  |  |  |
| --- | --- | --- |
| Test case name | Description | Result (pass/failed) |
| testBuildGraphForG1() | Compering a manually made G1 graph with G1 that was built with buildGraph function.  **Expecting to pass => return (assertEquals = true)** | **failed** |
| testFirstGreaterG1andG2WithbuildGraph() | Building G1 and G2 with buildGraph and checking if G1 is greater than G2.  **Expecting to pass** **=> return (assertTrue = true)** | **pass** |
| testFirstGreaterWithManualGraphsWithG1hasTwoAdhacnetNodesToNodeA () | Building G1 and G2 manually and checking if G1 is greater than G2.  **Expecting to pass** **=> return** **(assertEquals = true)** | **pass** |
| testFirstGreaterG2andG­­­­1WithbuildGraph() | Building G2 and G2 with buildGraph and checking if G2 isn't greater than G1.  **Expecting to pass => return (assertFalse = true)** | **pass** |

Unit test for Graph:

* For the following tests, we assume that isClose& calculateAlphaCharacteristicare working properly.

|  |  |  |
| --- | --- | --- |
| Test case name | Description | Result (pass/failed) |
| testisCloseThresholdIntInTheRange() | checking that two nodes are in the range Threshold1  **Expecting to pass** **=> return (assertTrue = true)** | **pass** |
| testisCloseThresholdInIsNotInTheRange () | checking that two nodes not are in the range Threshold.  **Expecting to pass** **=> return (assertFalse = false)** | **failed** |
| testisCloseThresholdThereIsNoConnectionBetweenNode () | checking that two nodes are not connected (there is no way to get from one note to another)  **Expecting to pass => return (assertFalse = false)** | **failed** |
| testisCloseSourceNodeIsNullAndevaluationNodeIsNode () | checking that when we don't send an actual node, we will catch an exception.  **Expecting to pass => return (assertThrows = true)** | **pass** |
| testcalculateAlphaCharacteristicWithLoops() | checking that the method returns the right calculation when there a loop and D-N+L >0  **Expecting to pass => return (result)** | **failed** |
| testcalculateAlphaCharacteristicWithNoLoops() | checking that the method returns the right calculation There is no loops and D-N +L > 0  **Expecting to pass => return (result)** | **failed** |
| testcalculateAlphaCharacteristic WhenThereIsLoops() | checking that the method returns the right calculation when there is loop and D-N+L<0  **Expecting to pass => return (result)** | **failed** |
| testcalculateAlphaCharacteristicWhenThereIs LoopsAndTheEquasionNotHappends() | checking that the method returns the right calculation when there is No loop and D-N+L<0  **Expecting to pass => return (result)** | **failed** |
| testcalculateAlphaCharacteristicWhenThereIs NoLoopsAndTheEquasionNotHappends() | checking that the method returns the right calculation when there is No loop and D-N+L<0  **Expecting to pass => return (result)** | **failed** |