|  |  |  |
| --- | --- | --- |
| Nodes & Edges (i) | Def (i) | Use (i) |
| 0 | {} | {} |
| 1 | {} | {} |
| (1,2), (1,3) | {} | {} |
| 2 | {data, empty} | {elem} |
| 3 | {} | {elem} |
| (3,4), (3,5) | {} | {elem} |
| 4 | {} | {} |
| 5 | {} | {data, elem} |
| (5,6), (5,7) | {} | {data, elem} |
| 6 | {tmp, data, elem} | {data, elem, tmp} |
| (6,7) | {} | {} |
| 7 | {} | {} |
| (7,8), (7,9) | {} | {} |
| 8 | {} | {elem} |
| 9 | {position} | {elem, position} |
| (9,10), (9,11) | {} | {position} |
| 10 | {previousValue, children[0].data} | {children[0].data, elem, previousValue} |
| (10,1) | {} | {previousValue} |
| 11 | {} | {nChildren, capacity, children[position]} |
| (11,12), (11,15) | {} | {nChildren, capacity, children[position]} |
| 12 | {} | {elem, children[position-1]} |
| (12,13), (12,14) | {} | {elem, children[position-1]}} |
| 13 | {} | {elem, position} |
| 14 | {} | {children[position-1], elem} |
| 15 | {} | {nChildren, capacity, elem, children[position]} |
| (15,16), (15,17) | {} | {nChildren, capacity, elem, children[position]} |
| 17 | {} | {position, capacity, elem, children[position]} |
| (17,16), (17,18) | {} | {nChildren, capacity, elem, children[position]} |
| 18 | {} | {position, capacity} |
| (18,19), (18,20) | {} | {position, capacity} |
| 19 | {} | {children[position-1], elem} |
| 20 | {} | {children[position], elem} |

2. Edge-Pair Coverage

|  |
| --- |
| Edge-Pair |
| [0,1,2],[0,1,3],[1,3,4],[1,3,5],[3,5,6],[3,5,7],[5,6,7],[5,7,8,],[5,7,9],[6,7,8],[6,7,9],[7,9,10],[7,9,11],[9,11,12],[9,11,15],[11,12,13],[11,12,14],[11,15,16],[11,15,17],[15,17,16],[15,17,18],[17,18,19],[17,18,20] |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Test Case Values  (ArrayNTree A, Capacity c, T Elem) | Expected Value | Test Path | Requirements covered |
| T1 | ([], 2, null) | [] | [0,1,2] | [0,1,2] |
| T2 | ([1], 2, 1) | [1] | [0,1,3,4] | [0,1,2],[0,1,3],[1,3,4] |
| T3 | ([2], 2, 1) | [1:[2]] | [0,1,3,5,6,7,8] | [0,1,3],[1,3,5],[5,6,7],[6,7,8] |
| T4 | ([1,2], 2, 3) | [1:[2][3]] | [0,1,3,5,7,9,11,12,13] | [0,1,3],[1,3,5],[3,5,7],[5,7,9],[7,9,11],[9,11,12],[11,12,13] |
| T5 | ([2,3], 2, 1) | [1:[2][3]] | [0,1,3,5,6,7,9,10] | [0,1,3],[1,3,5],[3,5,6],[5,6,7],[6,7,9],[7,9,10] |
| T6 | ([2:[7][11][15:[16][19]]], 5, 18) | [[2:[7][11][15:[16][18][19]]] | [0,1,3,5,7,9,11,12,14] | [0,1,3],[1,3,5],[3,5,7][5,7,9],[7,9,11],[9,11,12],[11,12,14] |
| T7 | ([2:[7][11][15:[18][19]][25]],5,20) | [2:[7][11][15:[18][19]][29][25]] | [0,1,3,5,7,9,11,15,16] | [0,1,3],[1,3,5],[3,5,7][5,7,9],[7,9,11],[9,11,15],[11,15,16] |
| T8 | ([2:[7][11][15:[16][19]]], 3, 18) | [2:[7][11][15:[16][18][19]]] | [0,1,3,5,7,9,11,15,17,18,19] | [0,1,3],[1,3,5],[3,5,7][5,7,9],[7,9,11],[11,15,17],[15,17,18],[17,18,19] |
| T9 | ([2:[7][11:[16][19]][20]], 3, 18) | [2:[7][11:[16][18][19]][20]] | [0,1,3,5,7,9,11,15,17,18,20] | [0,1,3],[1,3,5],[3,5,7][5,7,9],[7,9,11],[11,15,17],[15,17,18],[17,18,20] |

3. Prime Path Coverage

|  |  |
| --- | --- |
|  | Prime Paths |
| [1] | [1,2]! , [1,3,4]! , [1,3,5,7,8]! , [1,3,5,6,7,8]! , [1,3,5,7,9,10,1]\* , [1,3,5,6,7,9,10,1]\* , [1,3,5,7,9,11,12,13]! , [1,3,5,7,9,11,12,14]!, [1,3,5,7,9,11,15,16]! , [1,3,5,6,7,9,11,12,13]! , [1,3,5,6,7,9,11,12,14]!, [1,3,5,6,7,9,11,15,16]! , [1,3,5,7,9,11,15,17,16]! , [1,3,5,7,9,11,15,17,18,19,1]\* , [1,3,5,7,9,11,15,17,18,20,1]\* , [1,3,5,6,7,9,11,15,17,18,19,1]\* , [1,3,5,6,7,9,11,15,17,18,20,1]\* |
| [2] |  |
| [3] | [3,5,7,9,10,1,2]! , [3,5,7,9,10,1,3]\* , [3,5,6,7,9,10,1,2]! , [3,5,6,7,9,10,1,3]\* , [3,5,7,9,11,15,17,18,19,1,2]! , [3,5,7,9,11,15,17,18,19,1,3]\* , [3,5,7,9,11,15,17,18,20,1,2]! , [3,5,7,9,11,15,17,18,20,1,3]\* , [3,5,6,7,9,11,15,17,18,19,1,2]! , [3,5,6,7,9,11,15,17,18,19,1,3]\* , [3,5,6,7,9,11,15,17,18,20,1,2]! , [3,5,6,7,9,11,15,17,18,20,1,3]\* |
| [4] |  |
| [5] | [5,7,9,10,1,3,4]! , [5,7,9,10,1,3,5]\* , [5,6,7,9,10,1,3,4]! , [5,6,7,9,10,1,3,5]\* , [5,6,7,9,11,15,17,18,19,1,3,4]! , [5,6,7,9,11,15,17,18,19,1,3,5]\* , [5,6,7,9,11,15,17,18,20,1,3,4]! , [5,6,7,9,11,15,17,18,20,1,3,5]\* |
| [6] | [6,7,9,10,1,3,5,6]\*, [6,7,9,11,15,17,18,19,1,3,5,6]\* , [6,7,9,11,15,17,18,20,1,3,5,6]\* |
| [7] | [7,9,10,1,3,5,6,7]\* , [7,9,10,1,3,5,7]\* , [7,9,11,15,17,18,19,1,3,5,6,7]\* , [7,9,11,15,17,18,19,1,3,5,7]\* [7,9,11,15,17,18,20,1,3,5,6,7]\* , [7,9,11,15,17,18,20,1,3,5,7]\* |
| [8] |  |
| [9] | [9,10,1,3,5,6,7,8]! , [9,10,1,3,5,6,7,9]\* , [9,10,1,3,5,7,8]! , [9,10,1,3,5,7,9]\* , [9,11,15,17,18,19,1,3,5,6,7,8]!, [9,11,15,17,18,19,1,3,5,6,7,9]\* , [9,11,15,17,18,19,1,3,5,7,8]!, [9,11,15,17,18,19,1,3,5,7,9]\* , [9,11,15,17,18,20,1,3,5,6,7,8]!, [9,11,15,17,18,20,1,3,5,6,7,9]\* , [9,11,15,17,18,20,1,3,5,7,8]!, [9,11,15,17,18,20,1,3,5,7,9]\* |
| [10] | [10,1,3,5,6,7,9,10]\* , [10,1,3,5,7,9,10]\* , [10,1,3,5,6,7,9,11,12,13]! , [10,1,3,5,6,7,9,11,12,14]! , [10,1,3,5,6,7,9,11,15,16]! , [10,1,3,5,7,9,11,12,13]! , [10,1,3,5,7,9,11,12,14]! , [10,1,3,5,7,9,11,15,16]! , [10,1,3,5,6,7,9,11,15,17,16]! , [10,1,3,5,7,9,11,15,17,16]! |
| [11] | [11,15,17,18,19,1,3,5,6,7,9,11]\* , [11,15,17,18,19,1,3,5,7,9,11]\* , [11,15,17,18,20,1,3,5,6,7,9,11]\* , [11,15,17,18,20,1,3,5,7,9,11]\* |
| [12],  [13] , [14] |  |
| [15] | [15,17,18,19,1,3,5,6,7,9,11,12,13]! , [15,17,18,19,1,3,5,6,7,9,11,12,14]! , [15,17,18,19,1,3,5,6,7,9,11,15]\* , [15,17,18,19,1,3,5,7,9,11,12,13]! , [15,17,18,19,1,3,5,7,9,11,12,14]! , [15,17,18,19,1,3,5,7,9,11,15]\* , [15,17,18,20,1,3,5,6,7,9,11,12,13]! , [15,17,18,20,1,3,5,6,7,9,11,12,14]! , [15,17,18,20,1,3,5,6,7,9,11,15]\* , [15,17,18,20,1,3,5,7,9,11,12,13]! , [15,17,18,20,1,3,5,7,9,11,12,14]! , [15,17,18,20,1,3,5,7,9,11,15]\* |
| [16] |  |
| [17] | [17,18,19,1,3,5,6,7,9,11,15,16]! , [17,18,19,1,3,5,6,7,9,11,15,17]\* , [17,18,19,1,3,5,7,9,11,15,16]! , [17,18,19,1,3,5,7,9,11,15,17]\* , [17,18,20,1,3,5,6,7,9,11,15,16]! , [17,18,20,1,3,5,6,7,9,11,15,17]\* , [17,18,20,1,3,5,7,9,11,15,16]! , [17,18,20,1,3,5,7,9,11,15,17]\* |
| [18] | [18,19,1,3,5,6,7,9,11,15,16]! , [18,19,1,3,5,6,7,9,11,15,17,18]\* , [18,19,1,3,5,6,7,9,11,15,17,16]! , [18,19,1,3,5,7,9,11,15,16]! , [18,19,1,3,5,7,9,11,15,17,18]\* , [18,19,1,3,5,7,9,11,15,17,16]! , [18,20,1,3,5,6,7,9,11,15,16]! , [18,2,1,3,5,6,7,9,11,15,17,18]\* , [18,20,1,3,5,6,7,9,11,15,17,16]! , [18,20,1,3,5,7,9,11,15,16]! , [18,2,1,3,5,7,9,11,15,17,18]\* , [18,20,1,3,5,7,9,11,15,17,16]! |
| [19] | [19,1,3,5,6,7,9,11,15,16]! , [19,1,3,5,6,7,9,11,15,17,16]! , [19,1,3,5,6,7,9,11,15,17,18,19]\* , [19,1,3,5,7,9,11,15,16]! , [19,1,3,5,7,9,11,15,17,16]! , [19,1,3,5,7,9,11,15,17,18,19]\* |
| [20] | [20,1,3,5,6,7,9,11,15,16]! , [20,1,3,5,6,7,9,11,15,17,16]! , [20,1,3,5,6,7,9,11,15,17,18,20]\* , [20,1,3,5,7,9,11,15,16]! , [20,1,3,5,7,9,11,15,17,16]! , [20,1,3,5,7,9,11,15,17,18,20]\* |

|  |  |  |  |
| --- | --- | --- | --- |
|  | Test Case Values  (ArrayNTree A, Capacity c, T Elem) | Expected Value | Test Path |
| T1 | ([], 2, null) | [] | [1,2] |
| T2 | ([1], 2, 1) | [1] | [1,3,4] |
| T3 | ([2], 2, 1) | [1:[2]] | [1,3,5,6,7,8] |
| T4 | ([1:[2]], 2, 3) | [1:[2][3]] | [1,3,5,7,9,11,12,13] |
| T5 | ([2:[3]], 2, 1) | [1:[2][3]] | [1,3,5,6,7,9,10] |
| T6 | ([2:[7][11][15:[16][19]]], 5, 18) | [[2:[7][11][15:[16][18][19]]] | [1,3,5,7,9,11,12,14] |
| T7 | ([2:[7][11][15:[18][19]][25]],5,20) | [2:[7][11][15:[18][19]][29][25]] | [1,3,5,7,9,11,15,16] |
| T8 | ([2:[7][11][15:[16][19]]], 3, 18) | [2:[7][11][15:[16][18][19]]] | [1,3,5,7,9,11,15,17,18,19] |
| T9 | ([2:[7][11:[16][19]][20]], 3, 18) | [2:[7][11:[16][18][19]][20]] | [1,3,5,7,9,11,15,17,18,20] |
| T10 | ([1], 2, 2) | [1:[2]] | [0,1,3,5,7,8] |
| T11 | ([1:[3]], 2, 2) | [1:[2][3]] | [0,1,3,5,7,9,10] |
| T12 |  |  |  |

|  |  |  |
| --- | --- | --- |
| Prime Path | Covered by | |
| [1,2] | T1 |
| [1,3,4] | T2 |
| [1,3,5,7,8] | T10 |
| [1,3,5,6,7,8] | T3 |
| [1,3,5,7,9,10] | T11 |
| [1,3,5,6,7,9,10] | T5 |
| [1,3,5,7,9,11,12,13] | T4 |
| [1,3,5,7,9,11,12,14] | T6 |
| [1,3,5,6,7,9,11,12,13] |  |
| [1,3,5,6,7,9,11,12,14] |  |
| [1,3,5,7,9,11,15,16] | T7 |
| [1,3,5,6,7,9,11,15,16] |  |
| [1,3,5,7,9,11,15,17,18,19] | T8 |
| [1,3,5,7,9,11,15,17,18,20] | T9 |
| [1,3,5,6,7,9,11,15,17,18,19] |  |
| [1,3,5,6,7,9,11,15,17,18,20] |  |

4. Logic-based test coverage for method insert

|  |  |
| --- | --- |
| P | R(P) |
| P1 | True |
| P2 | R(P1) ∧ ¬P1 ⇔ !isEmpty() |
| P3 | R(P2) ∧ ¬P2 ⇔ !contains(elem) |
| P4 | R(P3) ∧ ¬P3 ⇔ data.compareTo(elem) ≤ 0 |
| P5 | R(P4) ∧ ¬P4 ⇔ !isLeaf() |
| P6 | R(P5) ∧ ¬P5 ⇔ position != -1 |
| P7 | R(P6) ∧ P6 ⇔ nChildren < capacity && children[position] == null |
| P8 | R(P6) ∧ ¬P6 ⇔ nChildren ≥ capacity || children[position] != null |
| P9 | R(P8) ∧ ¬P8 ⇔ nChildren ≥ capacity || elem.compareTo(children[position].max()) ≤ 0 |
| P10 | R(P9) ∧ P9 ⇔ nChildren == capacity || elem.compareTo(children[position].max()) < 0 |

|  |  |  |
| --- | --- | --- |
| Test | P | R(P) |
| List<Integer> list = Arrays.asList(2,7,11,15,16,19);  ArrayNTree<Integer> tree = new ArrayNTree<>(list, 3);  tree.insert(20); | P1  P2  P3  P4  P5  P6  P8  P9  P10  P1  P2  P3  P4  P5  P6  P7 | True  R(P1) ∧ ¬P1  R(P2) ∧ ¬P2  R(P3) ∧ ¬P3  R(P4) ∧ ¬P4  R(P5) ∧ ¬P5  R(P6) ∧ ¬P6  R(P8) ∧ ¬P8  R(P9) ∧ P9  R(P1) ∧ ¬P1  R(P2) ∧ ¬P2  R(P3) ∧ ¬P3  R(P4) ∧ ¬P4  R(P5) ∧ ¬P5  R(P6) ∧ P6 |

Apenas registámos os predicados que foram cobertos após a execução da terceira instrução, ou seja, após a chamada ao método Insert. Decidimos escolher este teste porque conseguia cobrir todos os predicados e cláusulas que identificámos na Tabela (P | R(P)).