

PROBLEM SOLVING ALGORITHMS

FINAL PROJECT MILESTONE - 1

TEAM

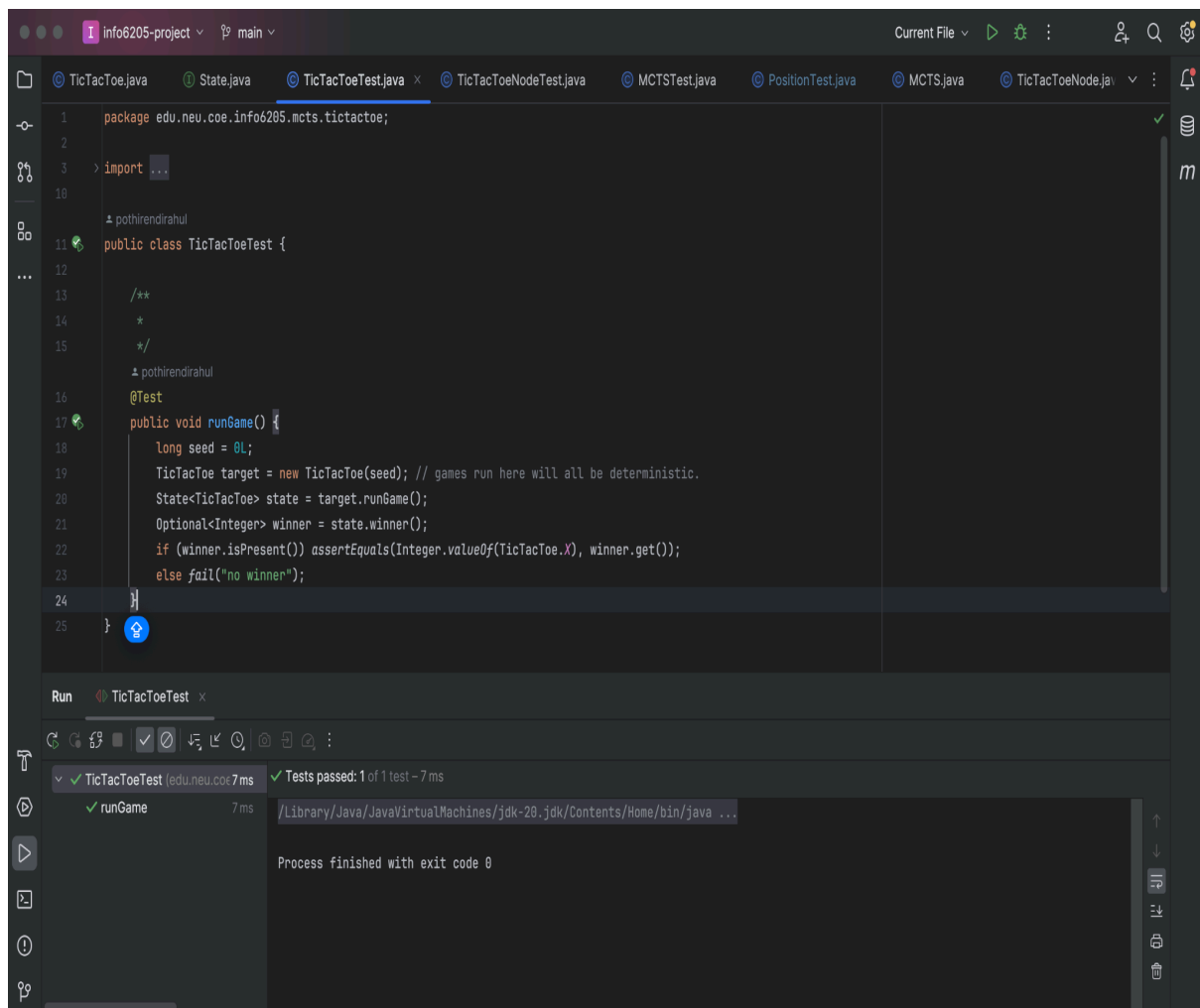
RAHUL POTHIRENDI

ROHIT VARMA

GITHUB LINK - <https://github.com/Pothirendirahul/info6205-project.git>

RUNNING UNIT TESTS

1) TICTACTOETEST.JAVA

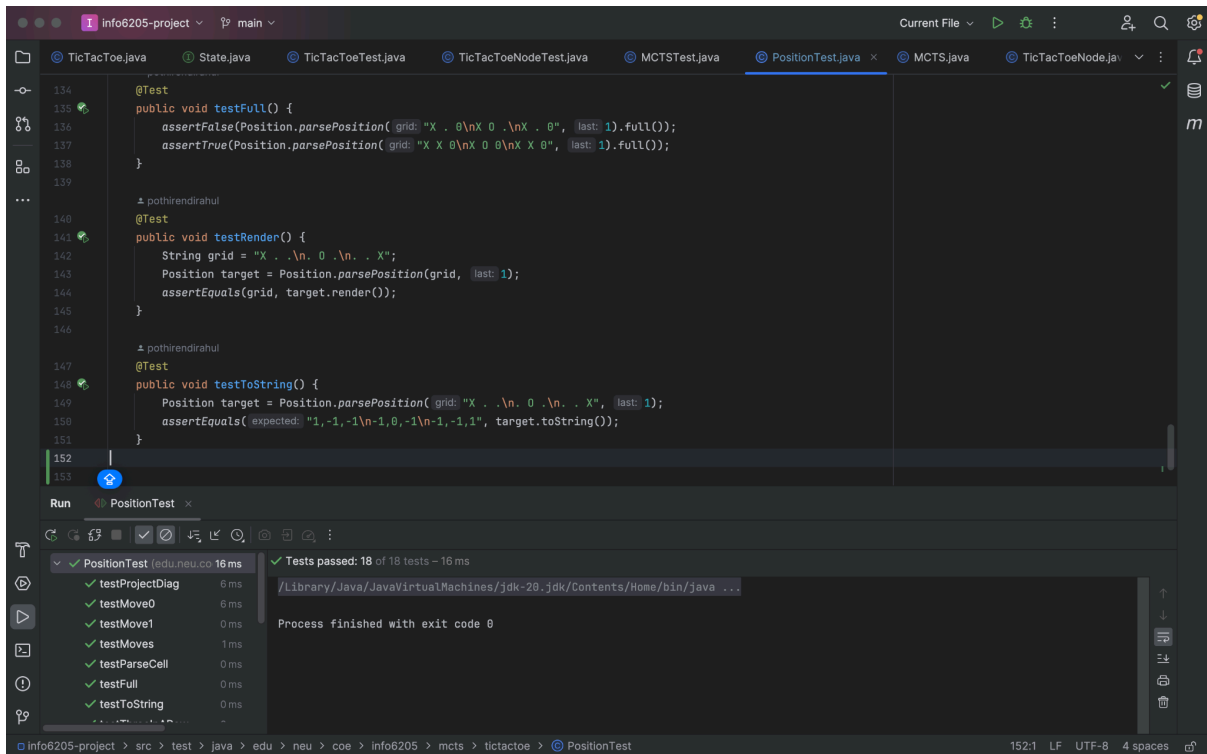


The screenshot shows an IDE window with the file `info6205-project` open. The `TicTacToeTest.java` file is selected, showing the following code:

```
1 package edu.neu.coe.info6205.mcts.tictactoe;
2
3 > import ...
10
11 public class TicTacToeTest {
12
13     /**
14     *
15     */
16     @Test
17     public void runGame() {
18         long seed = 0L;
19         TicTacToe target = new TicTacToe(seed); // games run here will all be deterministic.
20         State<TicTacToe> state = target.runGame();
21         Optional<Integer> winner = state.winner();
22         if (winner.isPresent()) assertEquals(Integer.valueOf(TicTacToe.X), winner.get());
23         else fail("no winner");
24     }
25 }
```

The `Run` button is clicked, and the test results are displayed in the bottom panel. The test `TicTacToeTest` (edu.neu.coe) passed, with 1 of 1 test passing in 7 ms. The output shows the path to the Java Virtual Machine and the message "Process finished with exit code 0".

2) POSITIONTEST.JAVA



The screenshot shows an IDE with the `PositionTest.java` file open. The code contains three test methods: `testFull()`, `testRender()`, and `testToString()`. The `testFull()` method tests the `Position.parsePosition()` method with two grid strings. The `testRender()` method tests the `Position.render()` method. The `testToString()` method tests the `Position.toString()` method. The test results pane shows that all tests passed, with a total of 18 tests passed in 16 ms.

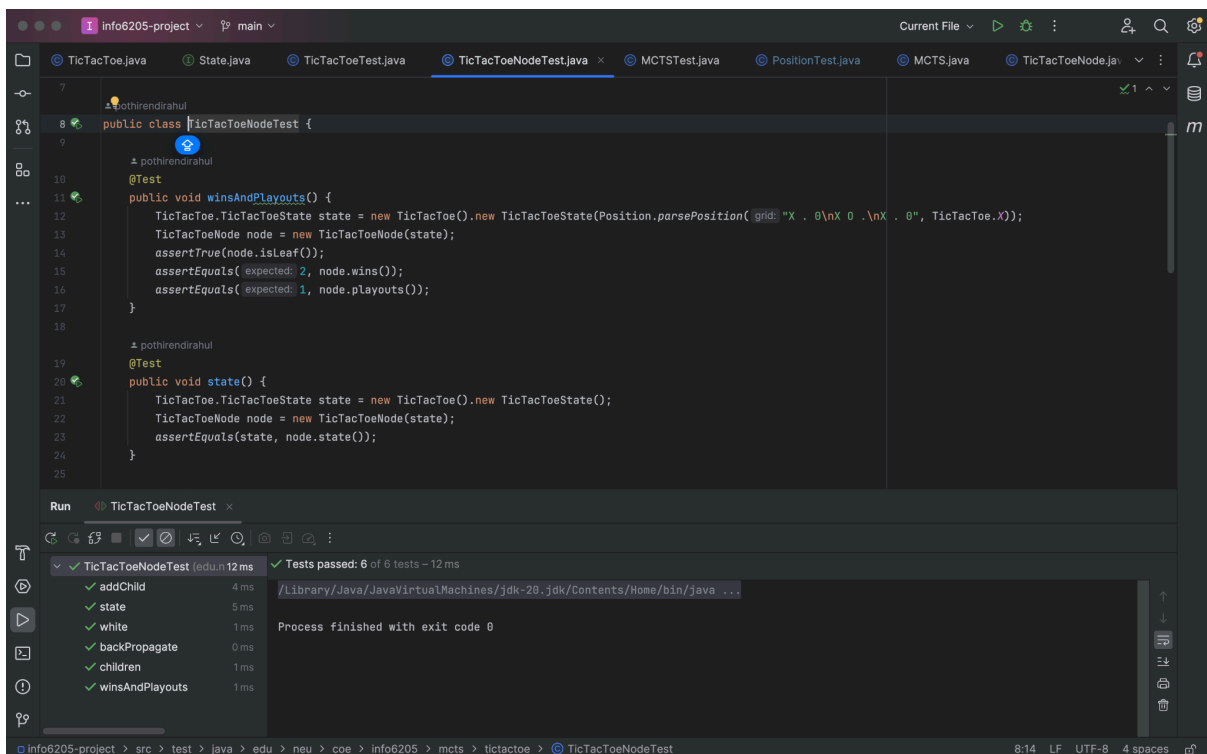
```
134  @Test
135  public void testFull() {
136      assertFalse(Position.parsePosition(grid: "X . 0\nX 0 .\nX . 0", last: 1).full());
137      assertTrue(Position.parsePosition(grid: "X X 0\nX 0 0\nX X 0", last: 1).full());
138  }
139
140  // pothirendirahul
141  @Test
142  public void testRender() {
143      String grid = "X . .\n. 0 .\n. . X";
144      Position target = Position.parsePosition(grid, last: 1);
145      assertEquals(grid, target.render());
146  }
147
148  // pothirendirahul
149  @Test
150  public void testToString() {
151      Position target = Position.parsePosition(grid: "X . .\n. 0 .\n. . X", last: 1);
152      assertEquals(expected: "1,-1,-1\n-1,0,-1\n-1,-1,1", target.toString());
153  }
```

Run PositionTest x

Tests passed: 18 of 18 tests - 16 ms

Process finished with exit code 0

3) TICTACTOENODE.JAVA



The screenshot shows an IDE with the `TicTacToeNodeTest.java` file open. The code contains two test methods: `winsAndPlayouts()` and `state()`. The `winsAndPlayouts()` method tests the `TicTacToeNode` class's `winsAndPlayouts()` method. The `state()` method tests the `TicTacToeNode` class's `state()` method. The test results pane shows that all tests passed, with a total of 6 tests passed in 12 ms.

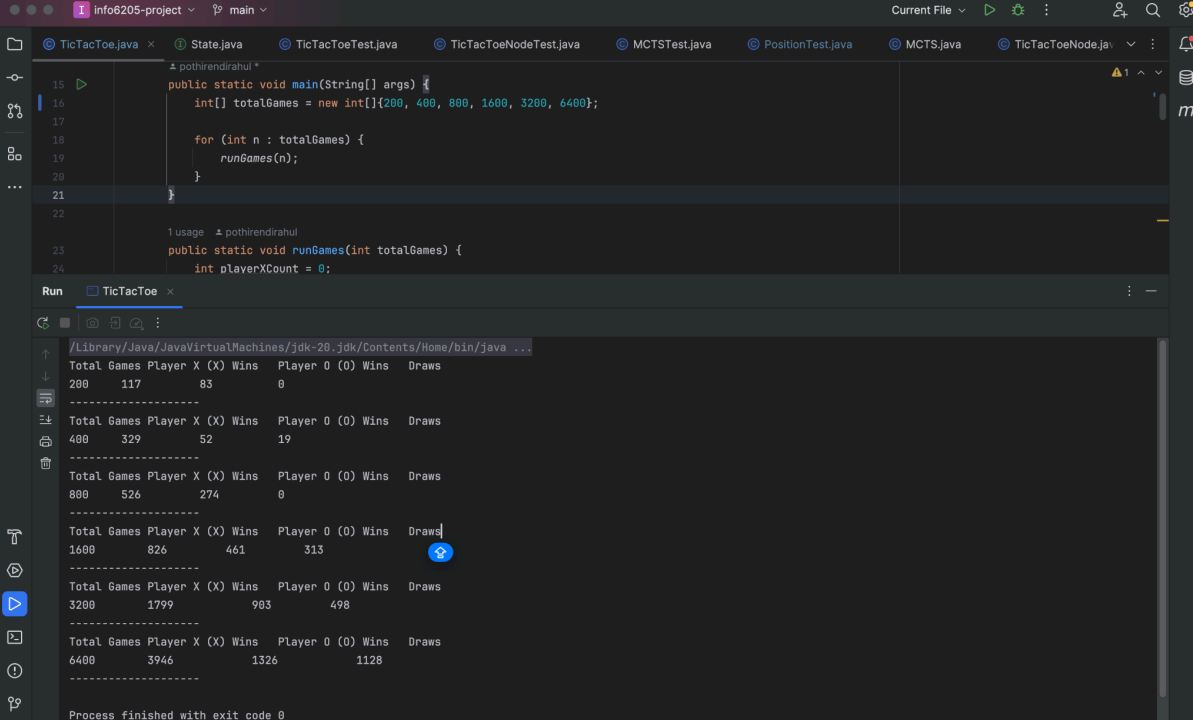
```
7
8  // pothirendirahul
9  public class TicTacToeNodeTest {
10
11      // pothirendirahul
12      @Test
13      public void winsAndPlayouts() {
14          TicTacToe.TicTacToeState state = new TicTacToe().new TicTacToeState(Position.parsePosition(grid: "X . 0\nX 0 .\nX . 0", TicTacToe.X));
15          TicTacToeNode node = new TicTacToeNode(state);
16          assertTrue(node.isLeaf());
17          assertEquals(expected: 2, node.wins());
18          assertEquals(expected: 1, node.playouts());
19      }
20
21      // pothirendirahul
22      @Test
23      public void state() {
24          TicTacToe.TicTacToeState state = new TicTacToe().new TicTacToeState();
25          TicTacToeNode node = new TicTacToeNode(state);
26          assertEquals(state, node.state());
27      }
28  }
```

Run TicTacToeNodeTest x

Tests passed: 6 of 6 tests - 12 ms

Process finished with exit code 0

IMPLEMENTATION OF TICTACTOE.JAVA



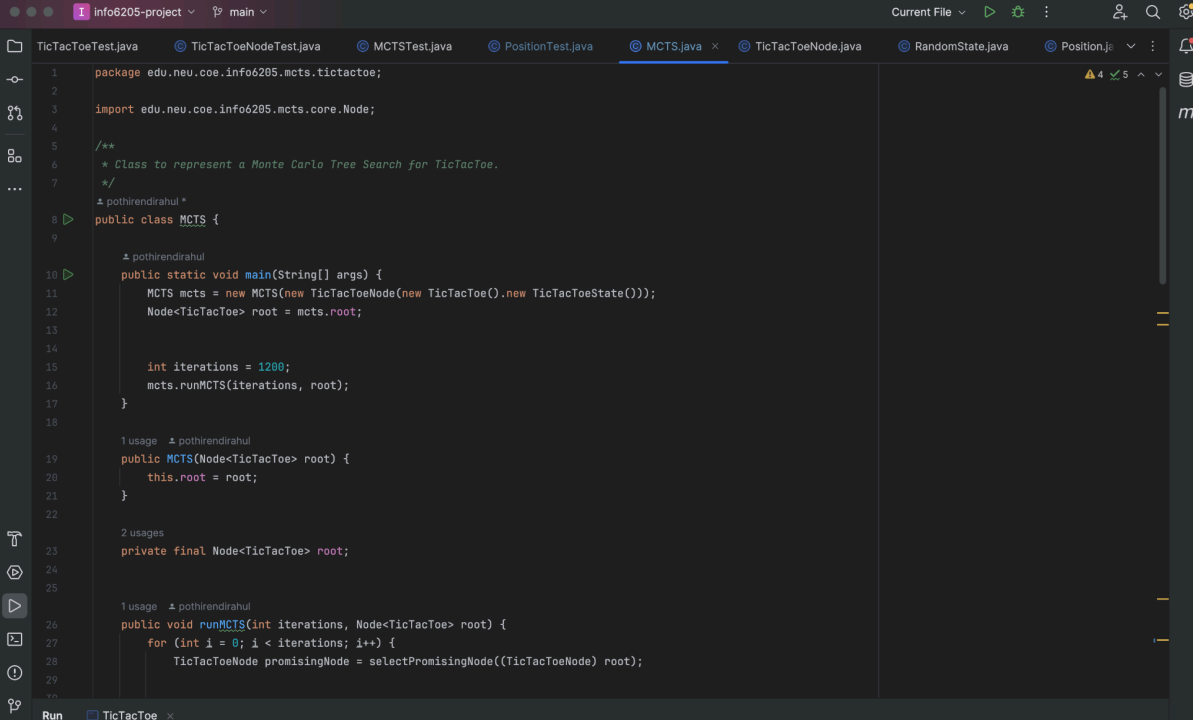
```
15 public static void main(String[] args) {
16     int[] totalGames = new int[]{200, 400, 800, 1600, 3200, 6400};
17
18     for (int n : totalGames) {
19         runGames(n);
20     }
21 }
22
23 1 usage: edu.neu.coe.info6205.mcts.tictactoe
24 public static void runGames(int totalGames) {
25     int playerXCount = 0;
```

Run TicTacToe

Total Games	Player X (X) Wins	Player 0 (0) Wins	Draws
200	117	83	0
400	329	52	19
800	526	274	0
1600	826	461	313
3200	1799	983	498
6400	3946	1326	1128

Process finished with exit code 0

MCTS



```
1 package edu.neu.coe.info6205.mcts.tictactoe;
2
3 import edu.neu.coe.info6205.mcts.core.Node;
4
5 /**
6  * Class to represent a Monte Carlo Tree Search for TicTacToe.
7  */
8 public class MCTS {
9
10     public static void main(String[] args) {
11         MCTS mcts = new MCTS(new TicTacToeNode(new TicTacToe().new TicTacToeState()));
12         Node<TicTacToe> root = mcts.root;
13
14         int iterations = 1200;
15         mcts.runMCTS(iterations, root);
16     }
17
18     1 usage: edu.neu.coe.info6205.mcts.tictactoe
19     public MCTS(Node<TicTacToe> root) {
20         this.root = root;
21     }
22
23     2 usages
24     private final Node<TicTacToe> root;
25
26     1 usage: edu.neu.coe.info6205.mcts.tictactoe
27     public void runMCTS(int iterations, Node<TicTacToe> root) {
28         for (int i = 0; i < iterations; i++) {
29             TicTacToeNode promisingNode = selectPromisingNode((TicTacToeNode) root);
30         }
31     }
32 }
```

Run TicTacToe