src/Tests/PartA_DFSTest.java

```
packageimportimportimportstaticimportimportimportpublicclassPartA_DFSTest
privatefinalByteArrayOutputStreamoutContent=newByteArrayOutputStream@Test
publicvoidtestBasicPathfinding()newPrintStreamNodegoal=newNode1180null0null
Nodestart=newNode190null0intplanetSize=4Stringfrontier_result="[(1:90)]\n"
"[(1:45),(1:135),(2:90)]\n""[(1:0),(2:45),(1:135),(2:90)]\n""[(1:315),(2:0),
(2:45),(1:135),(2:90)]\n""[(1:270),(2:315),(2:0),(2:45),(1:135),(2:90)]\n"
"[(1:225),(2:270),(2:315),(2:0),(2:45),(1:135),(2:90)]\n""[(1:180),(2:225),
(2:270),(2:315),(2:0),(2:45),(1:135),(2:90)]\n""(1:90)(1:45)(1:0)(1:315)
(1:270)(1:225)(1:180)\n""4.712\n""7\n"@TestpublicvoidtestAdvancedPathfinding()
newPrintStreamNodegoal=newNode290null0nullNodestart=newNode70null0int
planetSize=8Stringfrontier_result="[(7:0)]\n""[(6:0),(7:45),(7:315)]\n"
"[(5:0),(6:45),(6:315),(7:45),(7:315)]\n""[(4:0),(5:45),(5:315),(6:45),
(6:315),(7:45),(7:315)]\n""[(3:0),(4:45),(4:315),(5:45),(5:315),(6:45),
(6:315),(7:45),(7:315)]\n""[(2:0),(3:45),(3:315),(4:45),(4:315),(5:45),
(5:315),(6:45),(6:315),(7:45),(7:315)]\n""[(1:0),(2:45),(2:315),(3:45),
(3:315),(4:45),(4:315),(5:45),(5:315),(6:45),(6:315),(7:45),(7:315)]\n"
"[(1:45),(1:315),(2:45),(2:315),(3:45),(3:315),(4:45),(4:315),(5:45),(5:315),
(6:45),(6:315),(7:45),(7:315)]\n""[(1:90),(1:315),(2:45),(2:315),(3:45),
(3:315),(4:45),(4:315),(5:45),(5:315),(6:45),(6:315),(7:45),(7:315)]\n"
"[(1:135),(2:90),(1:315),(2:45),(2:315),(3:45),(3:315),(4:45),(4:315),(5:45),
(5:315),(6:45),(6:315),(7:45),(7:315)]\n""[(1:180),(2:135),(2:90),(1:315),
(2:45),(2:315),(3:45),(3:315),(4:45),(4:315),(5:45),(5:315),(6:45),(6:315),
(7:45),(7:315)]\n""[(1:225),(2:180),(2:135),(2:90),(1:315),(2:45),(2:315),
(3:45),(3:315),(4:45),(4:315),(5:45),(5:315),(6:45),(6:315),(7:45),(7:315)]\n"
"[(1:270),(2:225),(2:180),(2:135),(2:90),(1:315),(2:45),(2:315),(3:45),
(3:315),(4:45),(4:315),(5:45),(5:315),(6:45),(6:315),(7:45),(7:315)]\n"
"[(2:270),(2:225),(2:180),(2:135),(2:90),(1:315),(2:45),(2:315),(3:45),
(3:315),(4:45),(4:315),(5:45),(5:315),(6:45),(6:315),(7:45),(7:315)]\n"
"[(3:270),(2:225),(2:180),(2:135),(2:90),(1:315),(2:45),(2:315),(3:45),
(3:315),(4:45),(4:315),(5:45),(5:315),(6:45),(6:315),(7:45),(7:315)]\n"
"[(3:225),(4:270),(2:225),(2:180),(2:135),(2:90),(1:315),(2:45),(2:315),
(3:45),(3:315),(4:45),(4:315),(5:45),(5:315),(6:45),(6:315),(7:45),(7:315)]\n"
"[(3:180),(4:225),(4:270),(2:225),(2:180),(2:135),(2:90),(1:315),(2:45),
(2:315),(3:45),(3:315),(4:45),(4:315),(5:45),(5:315),(6:45),(6:315),(7:45),
(7:315)]\n""[(3:135),(4:180),(4:225),(4:270),(2:225),(2:180),(2:135),(2:90),
(1:315),(2:45),(2:315),(3:45),(3:315),(4:45),(4:315),(5:45),(5:315),(6:45),
(6:315),(7:45),(7:315)]\n""[(3:90),(4:135),(4:180),(4:225),(4:270),(2:225),
(2:180),(2:135),(2:90),(1:315),(2:45),(2:315),(3:45),(3:315),(4:45),(4:315),
(5:45),(5:315),(6:45),(6:315),(7:45),(7:315)]\n""[(4:90),(4:135),(4:180),
(4:225),(4:270),(2:225),(2:180),(2:135),(2:90),(1:315),(2:45),(2:315),(3:45),
(3:315),(4:45),(4:315),(5:45),(5:315),(6:45),(6:315),(7:45),(7:315)]\n"
"[(5:90),(4:135),(4:180),(4:225),(4:270),(2:225),(2:180),(2:135),(2:90),
(1:315),(2:45),(2:315),(3:45),(3:315),(4:45),(4:315),(5:45),(5:315),(6:45),
(6:315),(7:45),(7:315)]\n""[(5:135),(6:90),(4:135),(4:180),(4:225),(4:270),
(2:225),(2:180),(2:135),(2:90),(1:315),(2:45),(2:315),(3:45),(3:315),(4:45),
(4:315),(5:45),(5:315),(6:45),(6:315),(7:45),(7:315)]\n""[(5:180),(6:135),
(6:90),(4:135),(4:180),(4:225),(4:270),(2:225),(2:180),(2:135),(2:90),(1:315),
(2:45),(2:315),(3:45),(3:315),(4:45),(4:315),(5:45),(5:315),(6:45),(6:315),
(7:45),(7:315)]\n""[(5:225),(6:180),(6:135),(6:90),(4:135),(4:180),(4:225),
(4:270),(2:225),(2:180),(2:135),(2:90),(1:315),(2:45),(2:315),(3:45),(3:315),
(4:45),(4:315),(5:45),(5:315),(6:45),(6:315),(7:45),(7:315)]\n""[(5:270),
(6:225),(6:180),(6:135),(6:90),(4:135),(4:180),(4:225),(4:270),(2:225),
(2:180),(2:135),(2:90),(1:315),(2:45),(2:315),(3:45),(3:315),(4:45),(4:315),
(5:45),(5:315),(6:45),(6:315),(7:45),(7:315)]\n""[(6:270),(6:225),(6:180),
(6:135),(6:90),(4:135),(4:180),(4:225),(4:270),(2:225),(2:180),(2:135),(2:90),
(1:315),(2:45),(2:315),(3:45),(3:315),(4:45),(4:315),(5:45),(5:315),(6:45),
(6:315),(7:45),(7:315)]\n""[(7:270),(6:225),(6:180),(6:135),(6:90),(4:135),
(4:180),(4:225),(4:270),(2:225),(2:180),(2:135),(2:90),(1:315),(2:45),(2:315),
(3:45),(3:315),(4:45),(4:315),(5:45),(5:315),(6:45),(6:315),(7:45),(7:315)]\n"
"[(7:225),(6:225),(6:180),(6:135),(6:90),(4:135),(4:180),(4:225),(4:270),
(2:225),(2:180),(2:135),(2:90),(1:315),(2:45),(2:315),(3:45),(3:315),(4:45),
(4:315),(5:45),(5:315),(6:45),(6:315),(7:45),(7:315)]\n""[(7:180),(6:225),
(6:180),(6:135),(6:90),(4:135),(4:180),(4:225),(4:270),(2:225),(2:180),
(2:135),(2:90),(1:315),(2:45),(2:315),(3:45),(3:315),(4:45),(4:315),(5:45),
(5:315),(6:45),(6:315),(7:45),(7:315)]\n""[(7:135),(6:225),(6:180),(6:135),
(6:90),(4:135),(4:180),(4:225),(4:270),(2:225),(2:180),(2:135),(2:90),(1:315),
(2:45),(2:315),(3:45),(3:315),(4:45),(4:315),(5:45),(5:315),(6:45),(6:315),
(7:45),(7:315)]\n""[(7:90),(6:225),(6:180),(6:135),(6:90),(4:135),(4:180),
(4:225),(4:270),(2:225),(2:180),(2:135),(2:90),(1:315),(2:45),(2:315),(3:45),
(3:315),(4:45),(4:315),(5:45),(5:315),(6:45),(6:315),(7:45),(7:315)]\n"
"[(6:225),(6:180),(6:135),(6:90),(4:135),(4:180),(4:225),(4:270),(2:225),
(2:180),(2:135),(2:90),(1:315),(2:45),(2:315),(3:45),(3:315),(4:45),(4:315),
(5:45),(5:315),(6:45),(6:315),(7:45),(7:315)]\n""[(6:180),(6:135),(6:90),
(4:135),(4:180),(4:225),(4:270),(2:225),(2:180),(2:135),(2:90),(1:315),(2:45),
(2:315),(3:45),(3:315),(4:45),(4:315),(5:45),(5:315),(6:45),(6:315),(7:45),
(7:315)]\n""[(6:135),(6:90),(4:135),(4:180),(4:225),(4:270),(2:225),(2:180),
(2:135),(2:90),(1:315),(2:45),(2:315),(3:45),(3:315),(4:45),(4:315),(5:45),
(5:315),(6:45),(6:315),(7:45),(7:315)]\n""[(6:90),(4:135),(4:180),(4:225),
(4:270),(2:225),(2:180),(2:135),(2:90),(1:315),(2:45),(2:315),(3:45),(3:315),
(4:45),(4:315),(5:45),(5:315),(6:45),(6:315),(7:45),(7:315)]\n""[(4:135),
(4:180),(4:225),(4:270),(2:225),(2:180),(2:135),(2:90),(1:315),(2:45),(2:315),
(3:45),(3:315),(4:45),(4:315),(5:45),(5:315),(6:45),(6:315),(7:45),(7:315)]\n"
"[(4:180),(4:225),(4:270),(2:225),(2:180),(2:135),(2:90),(1:315),(2:45),
(2:315),(3:45),(3:315),(4:45),(4:315),(5:45),(5:315),(6:45),(6:315),(7:45),
(7:315)]\n""[(4:225),(4:270),(2:225),(2:180),(2:135),(2:90),(1:315),(2:45),
(2:315),(3:45),(3:315),(4:45),(4:315),(5:45),(5:315),(6:45),(6:315),(7:45),
(7:315)]\n""[(4:270),(2:225),(2:180),(2:135),(2:90),(1:315),(2:45),(2:315),
(3:45),(3:315),(4:45),(4:315),(5:45),(5:315),(6:45),(6:315),(7:45),(7:315)]\n"
"[(2:225),(2:180),(2:135),(2:90),(1:315),(2:45),(2:315),(3:45),(3:315),(4:45),
(4:315),(5:45),(5:315),(6:45),(6:315),(7:45),(7:315)]\n""[(2:180),(2:135),
(2:90),(1:315),(2:45),(2:315),(3:45),(3:315),(4:45),(4:315),(5:45),(5:315),
(6:45),(6:315),(7:45),(7:315)]\n""[(2:135),(2:90),(1:315),(2:45),(2:315),
(3:45),(3:315),(4:45),(4:315),(5:45),(5:315),(6:45),(6:315),(7:45),(7:315)]\n"
"[(2:90),(1:315),(2:45),(2:315),(3:45),(3:315),(4:45),(4:315),(5:45),(5:315),
(6:45),(6:315),(7:45),(7:315)]\n""(7:0)(6:0)(5:0)(4:0)(3:0)(2:0)(1:0)(1:45)
(1:90)(2:90)\n""8.571\n""43\n"@TestpublicvoidtestAdvancedPathfinding2()new
PrintStreamNodegoal=newNode10null0nullNodestart=newNode3180null0intplanetSize=
4Stringfrontier_result="[(3:180)]\n""[(2:180),(3:135),(3:225)]\n""[(1:180),
(2:135),(2:225),(3:135),(3:225)]\n""[(1:135),(1:225),(2:135),(2:225),(3:135),
(3:225)]\n""[(1:90),(1:225),(2:135),(2:225),(3:135),(3:225)]\n""[(1:45),
(2:90),(1:225),(2:135),(2:225),(3:135),(3:225)]\n""[(1:0),(2:45),(2:90),
(1:225),(2:135),(2:225),(3:135),(3:225)]\n""(3:180)(2:180)(1:180)(1:135)(1:90)
(1:45)(1:0)\n""5.142\n""7\n"@TestpublicvoidtestGoalOfZero()newPrintStreamNode
goal=newNode00null0nullNodestart=newNode10null0intplanetSize=2String
frontier_result="[(1:0)]\n""[(1:45),(1:315)]\n""[(1:90),(1:315)]\n""[(1:135),
(1:315)]\n""[(1:180),(1:315)]\n""[(1:225),(1:315)]\n""[(1:270),(1:315)]\n"
"[(1:315)]\n""fail\n""8\n"// expected here is weirdly long?@Testpublicvoid
testStartOfZero()newPrintStreamNodegoal=newNode20null0nullNodestart=newNode00
null0intplanetSize=2Stringfrontier_result="[(0:0)]\n""[(1:0)]\n""[(1:45),
(1:315)]\n""[(1:90),(1:315)]\n""[(1:135),(1:315)]\n""[(1:180),(1:315)]\n"
"[(1:225),(1:315)]\n""[(1:270),(1:315)]\n""[(1:315)]\n""fail\n""9\n"@Test
publicvoidtestEdgeOfTheGrid()newPrintStreamNodegoal=newNode345null0nullNode
start=newNode30null0intplanetSize=2Stringfrontier_result="[(3:0)]\n""fail\n"
"1\n"@TestpublicvoidtestGoalOffGrid()newPrintStreamNodegoal=newNode3315null0
nullNodestart=newNode1270null0intplanetSize=2Stringfrontier_result=
"[(1:270)]\n""[(1:225),(1:315)]\n""[(1:180),(1:315)]\n""[(1:135),(1:315)]\n"
"[(1:90),(1:315)]\n""[(1:45),(1:315)]\n""[(1:0),(1:315)]\n""[(1:315)]\n"
"fail\n""8\n"@TestpublicvoidtestStartOffGrid()newPrintStreamNodegoal=newNode1
315null0nullNodestart=newNode4270null0intplanetSize=2Stringfrontier_result=
"[(4:270)]\n""fail\n""1\n" Tests;

     General.Node;
     Algorithms.PartA_DFS;
      org.junit.Assert.*;

     java.io.ByteArrayOutputStream;
     java.io.PrintStream;

     org.junit.Test;

       {

              ();


           {
            System.setOut( (outContent));

                (, , , , );
                (, , , , goal);
               ;

                    +
                    +
                    +
                    +
                    +
                    +
                    +
                    +
                    + ;

            PartA_DFS.dfs(start, goal, planetSize);
            assertEquals(frontier_result, outContent.toString());
        }


           {
            System.setOut( (outContent));

                (, , , , );
                (, , , , goal);
               ;

                    +
                    +
                    +
                    +
                    +
                    +
                    +
                    +
                    +
                    +
                    +
                    +
                    +
                    +
                    +
                    +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     + ;

             PartA_DFS.dfs(start, goal, planetSize);
             assertEquals(frontier_result, outContent.toString());
         }


            {
             System.setOut( (outContent));

                 (, , , , );
                 (, , , , goal);
                ;

                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     + ;

             PartA_DFS.dfs(start, goal, planetSize);
             assertEquals(frontier_result, outContent.toString());
         }


            {
             System.setOut( (outContent));

                 (, , , , );
                 (, , , , goal);
                ;

                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     + ;

             PartA_DFS.dfs(start, goal, planetSize);
             assertEquals(frontier_result, outContent.toString());
         }


            {
             System.setOut( (outContent));

                 (, , , , );
                 (, , , , goal);
                ;

                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     + ;

             PartA_DFS.dfs(start, goal, planetSize);
             assertEquals(frontier_result, outContent.toString());
         }


            {
             System.setOut( (outContent));

                 (, , , , );
                 (, , , , goal);
                ;

                     +
                     + ;

             PartA_DFS.dfs(start, goal, planetSize);
             assertEquals(frontier_result, outContent.toString());
         }


            {
             System.setOut( (outContent));

                 (, , , , );
                 (, , , , goal);
                ;

                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     +
                     + ;

             PartA_DFS.dfs(start, goal, planetSize);
             assertEquals(frontier_result, outContent.toString());
         }


            {
             System.setOut( (outContent));

                 (, , , , );
                 (, , , , goal);
                ;

                     +
                     + ;

             PartA_DFS.dfs(start, goal, planetSize);
             assertEquals(frontier_result, outContent.toString());
         }
     }

```