src/Tests/PartB_BestFTests.java

```
packageimportstaticimportimportimportimportimportpublicclass
PartB_BestFTestsprivatefinalByteArrayOutputStreamoutContent=new
ByteArrayOutputStream@TestpublicvoidtestBasicPathfinding()newPrintStreamNode
goal=newNode1180null0nullNodestart=newNode190null0intplanetSize=4// Setting
the planet size for successor generationStringfrontier_result=
"[(1:90)1.414]\n""[(1:135)0.765,(1:45)1.848,(2:90)2.236]\n""[(1:180)0.000,
(2:135)1.474,(1:45)1.848,(2:90)2.236]\n""(1:90)(1:135)(1:180)\n""1.571\n""3\n"
@TestpublicvoidtestAdvancedPathfinding()newPrintStreamNodegoal=newNode290null0
nullNodestart=newNode70null0intplanetSize=8Stringfrontier_result=
"[(7:0)7.280]\n""[(7:45)5.762,(6:0)6.325,(7:315)8.532]\n""[(6:45)4.799,
(7:90)5.000,(6:0)6.325,(7:315)8.532]\n""[(5:45)3.855,(6:90)4.000,(7:90)5.000,
(6:0)6.325,(7:315)8.532]\n""[(4:45)2.947,(5:90)3.000,(6:90)4.000,(7:90)5.000,
(5:0)5.385,(6:0)6.325,(7:315)8.532]\n""[(4:90)2.000,(3:45)2.125,(5:90)3.000,
(6:90)4.000,(4:0)4.472,(7:90)5.000,(5:0)5.385,(6:0)6.325,(7:315)8.532]\n"
"[(3:90)1.000,(3:45)2.125,(4:135)2.947,(5:90)3.000,(6:90)4.000,(4:0)4.472,
(7:90)5.000,(5:0)5.385,(6:0)6.325,(7:315)8.532]\n""[(2:90)0.000,(3:45)2.125,
(3:135)2.125,(4:135)2.947,(5:90)3.000,(6:90)4.000,(4:0)4.472,(7:90)5.000,
(5:0)5.385,(6:0)6.325,(7:315)8.532]\n""(7:0)(7:45)(6:45)(5:45)(4:45)(4:90)
(3:90)(2:90)\n""13.639\n""8\n"@TestpublicvoidtestAdvancedPathfinding2()new
PrintStreamNodegoal=newNode10null0nullNodestart=newNode3180null0intplanetSize=
4Stringfrontier_result="[(3:180)4.000]\n""[(2:180)3.000,(3:135)3.774,
(3:225)3.774]\n""[(1:180)2.000,(2:135)2.798,(2:225)2.798,(3:135)3.774,
(3:225)3.774]\n""[(1:135)1.848,(1:225)1.848,(2:135)2.798,(2:225)2.798,
(3:135)3.774,(3:225)3.774]\n""[(1:90)1.414,(1:225)1.848,(2:135)2.798,
(2:225)2.798,(3:135)3.774,(3:225)3.774]\n""[(1:45)0.765,(1:225)1.848,
(2:90)2.236,(2:135)2.798,(2:225)2.798,(3:135)3.774,(3:225)3.774]\n"
"[(1:0)0.000,(2:45)1.474,(1:225)1.848,(2:90)2.236,(2:135)2.798,(2:225)2.798,
(3:135)3.774,(3:225)3.774]\n""(3:180)(2:180)(1:180)(1:135)(1:90)(1:45)(1:0)\n"
"5.142\n""7\n"@TestpublicvoidtestGoalOfZero()newPrintStreamNodegoal=newNode00
null0nullNodestart=newNode190null0intplanetSize=2Stringfrontier_result=
"[(1:90)1.000]\n""[(1:45)1.000,(1:135)1.000]\n""[(1:0)1.000,(1:135)1.000]\n"
"[(1:135)1.000,(1:315)1.000]\n""[(1:180)1.000,(1:315)1.000]\n""[(1:225)1.000,
(1:315)1.000]\n""[(1:270)1.000,(1:315)1.000]\n""[(1:315)1.000]\n""fail\n""8\n"
@TestpublicvoidtestGoalOutOfBounds()newPrintStreamNodegoal=newNode490null0null
Nodestart=newNode190null0intplanetSize=2Stringfrontier_result=
"[(1:90)3.000]\n""[(1:45)3.368,(1:135)3.368]\n""[(1:135)3.368,(1:0)4.123]\n"
"[(1:0)4.123,(1:180)4.123]\n""[(1:180)4.123,(1:315)4.760]\n""[(1:225)4.760,
(1:315)4.760]\n""[(1:315)4.760,(1:270)5.000]\n""[(1:270)5.000]\n""fail\n""8\n"
@TestpublicvoidtestStartOutOfBounds()newPrintStreamNodegoal=newNode190null0
nullNodestart=newNode490null0intplanetSize=2Stringfrontier_result=
"[(4:90)3.000]\n""fail\n""1\n" Tests;

  org.junit.Assert.assertEquals;

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

 org.junit.Test;

 Algorithms.PartB_BestF;
 General.Node;

   {

          ();


       {
        System.setOut( (outContent));

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

                +
                +
                +
                +
                + ;

        PartB_BestF.BestF(start, goal, planetSize);
        assertEquals(frontier_result, outContent.toString());
    }


       {
        System.setOut( (outContent));

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

                +
                +
                +
                +
                +
                +
                +
                +
                +
                + ;

        PartB_BestF.BestF(start, goal, planetSize);
        assertEquals(frontier_result, outContent.toString());
    }


       {
        System.setOut( (outContent));

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

                +
                +
                +
                +
                +
                +
                +
                +
                + ;

        PartB_BestF.BestF(start, goal, planetSize);
        assertEquals(frontier_result, outContent.toString());
    }

```

```

       {
        System.setOut( (outContent));

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

                +
                +
                +
                +
                +
                +
                +
                +
                + ;
        PartB_BestF.BestF(start, goal, planetSize);
        assertEquals(frontier_result, outContent.toString());
    }


       {
        System.setOut( (outContent));

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

                +
                +
                +
                +
                +
                +
                +
                +
                + ;

        PartB_BestF.BestF(start, goal, planetSize);
        assertEquals(frontier_result, outContent.toString());
    }


       {
        System.setOut( (outContent));

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

                +
                + ;

        PartB_BestF.BestF(start, goal, planetSize);
        assertEquals(frontier_result, outContent.toString());
    }
}

```