```java
  1    package edu.seaaddicts.brockbutler.maps;
  2
  3    /**
  4     * Position.java
  5     * Brock Butler
  6     * Type for holding Position node
  7     * portion of Brock Butler.
  8     * Created by Thomas Nelson 2013-03-05
  9     * Copyright (c) 2013 Sea Addicts. All rights reserved.
 10     */
 11
 12    import android.util.Log;
 13
 14    public class Position implements Comparable<Object> {
 15
 16      /**
 17       * Class variable for the POSITION class. All are public
 18       * to avoid using get/set variables to increase performance
 19       */
 20      public int      xPosition;
 21      public int      yPosition;
 22      public double   fScore;
 23      public double   gScore;
 24      public double   hScore;
 25      public String   nodeNumber;
 26      public String   nodeName;
 27      public boolean  visited;
 28        public Position from;
 29        public Position accesible[];
 30        public Position nonaccesible[];
 31
 32      /**
 33       * Constructor methods for no arguments
 34       */
 35      public Position ( ) {
 36        xPosition = 0;
 37        yPosition = 0;
 38
 39        nodeNumber = "";
 40        nodeName   = "";
 41
 42        fScore = Double.MAX_VALUE;
 43            gScore = Double.MAX_VALUE;
 44            hScore = -1;
 45
 46            visited = false;
 47            from    = null;
 48      }
 49
 50      /**
 51       * Constructor with coordinates set
 52       * @param inputX
 53       * @param inputY
 54       */
 55      public Position (int inputX, int inputY) {
 56        xPosition = inputX;
 57        yPosition = inputY;
 58
 59        nodeNumber = "";
 60        nodeName   = "";
 61
 62        fScore = Double.MAX_VALUE;
 63        gScore = Double.MAX_VALUE;
 64        hScore = Double.MAX_VALUE;
 65
 66        visited = false;
 67            from    = null;
 68      }
 69
 70      /**
 71       * Constructor with all position information set
```

```java
 72          * @param inputX
 73          * @param inputY
 74          * @param inputName
 75          * @param inputNumber
 76          */
 77         public Position (int inputX, int inputY, String inputName, String inputNumber) {
 78           xPosition = inputX;
 79           yPosition = inputY;
 80
 81           nodeNumber = inputNumber;
 82           nodeName   = inputName;
 83
 84           fScore = Double.MAX_VALUE;
 85           gScore = Double.MAX_VALUE;
 86           hScore = Double.MAX_VALUE;
 87
 88           visited = false;
 89               from    = null;
 90         }
 91
 92         /**
 93          * Set coordinates
 94          * @param inputX
 95          * @param inputY
 96          */
 97         public void setCoordinates (int inputX, int inputY) {
 98           xPosition  = inputX;
 99           yPosition  = inputY;
100         }
101
102         /**
103          * Set position number
104          * @param inputNumber
105          */
106         public void setNumber (String inputNumber) {
107           nodeNumber  = inputNumber;
108         }
109
110         /**
111          * Set position description
112          * @param inputName
113          */
114         public void setName (String inputName) {
115           nodeName  = inputName;
116         }
117
118         /**
119          * get x coordinate
120          * @return
121          */
122         public int getX ( ) {
123           return xPosition;
124         }
125
126         /**
127          * get y coordinate
128          * @return
129          */
130         public int getY ( ) {
131           return yPosition;
132         }
133
134         /**
135          * get node numner
136          * @return
137          */
138         public String getNumber ( ) {
139           return nodeNumber;
140         }
141
142         /**
```

```
143         * Get node name
144         * @return
145         */
146        public String getName ( ) {
147           return nodeName;
148        }
149
150        /**
151         * Compares this node to another
152         * @param node
153         * @return
154         */
155        public boolean compare (Position node) {
156           if(this.xPosition == node.xPosition && this.yPosition == node.yPosition && this.
              nodeNumber.equals(node.nodeNumber) && this.nodeName.equals(node.nodeName))
157             return true;
158           return false;
159        }
160
161        /**
162         * Not Used but required???
163         */
164        public int compareTo (Object node) {
165           Position temp = (Position)node;
166               return (int)(fScore - temp.fScore);
167        }
168
169
170        /**
171         * Testing methods for the POSITION class. These methods are provided
172         * for testing and debugging purposes capable of printing variables to the log
173         */
174        public void printCoordinates ( ) {
175           Log.d("POSITION CLASS", "Coordinates: (" + xPosition + "," + yPosition + ")");
176        }
177
178        public void printNumber ( ) {
179           Log.d("POSITION CLASS", "Node Number: " + nodeNumber);
180        }
181
182        public void printName ( ) {
183           Log.d("POSITION CLASS", "Node Name: " + nodeName);
184        }
185     }
```