```
import java.util.*;
   /************
2
    * This class builds rooms for the Game.
3
    * @author Ben Parsell
5
    * @version 1.0.0 (11/19/2015)
   public class Room
8
9
      /** Describes the room **/
10
      private String roomDescription;
11
12
      /** Item in the room **/
13
      private Item optionalItem;
14
15
      /** ArrayList for rooms **/
16
      private HashMap <String, Room> neighbors;
17
18
19
20
21
      /*****************
22
       * Overloaded Constuctor
23
24
       * @param pDescription variable to describe room
25
       * @param pItem variable for item in the room
26
       ****************
27
      public Room(String pDescription) {
          // Initialize instance variables
29
          this.roomDescription = pDescription;
30
31
32
          // Creates hashmap for neighbors
          neighbors = new HashMap <String, Room> ();
33
      }
34
35
36
      /******************
37
       * Default Constructor for Room
38
       * @param pDescription variable to describe room
40
       ******************
41
      public Room(String pDescription, Item pItem) {
42
          this.roomDescription = pDescription;
43
          this.optionalItem = pItem;
44
          neighbors = new HashMap<String, Room>();
45
      }
46
47
48
      /*****************
49
```

```
* Method to add an item into the game
50
       * @param i variable for the item in the room
52
      ***************
      public void addItem(Item i) {
54
         this.optionalItem = i;
56
57
58
      /*******************
59
       * Method to check if player has an item
60
61
       * @return returns true of false depending on null
62
63
      *****************
64
      public boolean hasItem() {
65
         // Check for if room has an item
         if(optionalItem != null) {
67
            return true;
68
         } else {
69
            return false;
70
71
      }
72
73
74
      /*****************
75
76
       * Method to get the description of the room
77
       * @return roomDescription describes room
78
79
80
      public String getDescription() {
81
         return roomDescription;
82
83
      /*********************
84
85
       * Method to get an item in the Room
86
       * @return optionalItem item within the room
87
       *************
      public Item getItem() {
89
90
         return optionalItem;
91
92
93
      /****************
94
95
       * Method to view neighboring rooms
96
       * @return value for neighboring rooms
97
       ******************
98
```

```
Class Room - Project 5 Updated (continued)
                                                                    3/4
      public HashMap getNeighbors() {
99
          return neighbors;
100
       }
101
102
103
       /****************
        * Method to add a neighboring room
105
106
       * @param direction value for direction to move
107
       * @param r value for room name
108
       *************
109
110
      public void addNeighbor(String direction, Room r) {
          // Adds neighbor to room r, in that direction
111
          this.neighbors.put(direction, r);
112
       }
113
114
       /*****************
116
       * Finds neighbor in given direction
117
118
       * @param direction value for direction to move
119
120
      public Room getNeighbor(String direction) {
121
          // Create new room to find neighbor
122
          Room neighbor = this.neighbors.get(direction);
123
          return neighbor;
124
125
       }
127
       /*****************
128
129
        * Method removes an item from the room
130
       * @return optionalItem item within the room
131
       ******************
132
      public Item removeItem() {
133
          // Set item in room to null, to delete
134
135
          Item temp = optionalItem;
          this.optionalItem = null;
136
          return temp;
137
       }
138
```

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Method to retrieve the full description of a room

\* @return longDescription value for well-described

\* room information

public String getLongDescription() {

139 140

141

142 143 144

145

146

147

```
Class Room - Project 5 Updated (continued)
```

```
4/4
```

```
// Check to see if room has an item, then return text
148
            if(hasItem()) {
149
                String longDescription = "You are currently in " + roomDescri
150
   ption + ".\n You see " + optionalItem.getDescription() + ".";
               return longDescription + "\n";
151
153
            // If no item, gives room description without item.
154
            else {
155
                String longDescription = "You are in " + roomDescription + ".
156
                return longDescription + "\n";
157
            }
       }
159
160
161
```