```java
/*
 * Programmer:  Chad Philip Johnson
 * Date Created:  Thursday, November 06th, 2012
 * Date of Last Modification:  Tuesday, December 11th, 2012
 *
 * Description:
 * Room.class contains details about each of the room objects held within
 * the dungeon.  Room.class depends on Item.class and its subclasses
 * Armor.class and Weapon.class for placing items in rooms, and on
 * Character.class for placing characters within rooms.
 */

import java.util.*;
import java.io.Serializable;

/**
 * Room.class contains details about each of the room objects held within
 * the dungeon.  Room.class depends on Item.class and its subclasses
 * Armor.class and Weapon.class for placing items in rooms, and on
 * Character.class for placing characters within rooms.
 *
 * @author Chad Philip Johnson
 * @version 1.0
 */


public class Room implements Serializable   {

    String strRoomName, strRoomDescription;
    int[] intConnectedRooms;
    int intGoldPieces;
    ArrayList<Item> itemsInRoom;
    ArrayList<Character> charactersInRoom;

    // NOTE:  The project requirement for there to be "a pointer/reference to the player object so that when the game is saved the player's
    //          location will be saved as well" is handled by the currentRoom variable in Game.class.  Writing this functionality into this class
    //          specifically would have made the implementation of this program/game unnecessarily complicated.

    /**
      * Overloaded constructor:
      * Sets all details about a room:  name, description, numerical values  for connected rooms, items in the room, characters in the room,
      * and gold pieces in the room.
      */

    public Room(    String strRoomName,
                    String strRoomDescription,
                    int intConnectedRooms[],
```

```
48                      Item defaultItems[],
49                      Character defaultCharacters[],
50                      int intGoldPieces ) {
51
52          this.strRoomName        = strRoomName;
53          this.strRoomDescription = strRoomDescription;
54          this.intConnectedRooms  = intConnectedRooms;
55
56          this.itemsInRoom    = new ArrayList<Item>();
57          for( Item item : defaultItems ) {   this.itemsInRoom.add( item );   }
58
59          this.charactersInRoom   = new ArrayList<Character>();
60          for( Character character : defaultCharacters )  {   this.charactersInRoom.add( character ); }
61
62          this.intGoldPieces      = intGoldPieces;
63
64      }
65
66      /**
67       * Remove a currently in the room.
68       *
69       * @param intIndex Numerical value representing the item to be removed.
70       * @return Item Item that was removed.
71       */
72
73      public Item removeItem( int intIndex )  {   return itemsInRoom.remove( intIndex );  }
74
75      /**
76       * Add an item to the room.
77       *
78       * @param itemToAdd The item object that is too be added to the room.
79       */
80
81      public void addItem( Item itemToAdd )   {   itemsInRoom.add( itemToAdd );   }
82
83      /**
84       * Remove a character in the room.
85       *
86       * @param intIndex Numerical value representing the character to be removed.
87       * @return Character Character object that has been removed from the room.
88       */
89
90      public Character removeCharacter( int intIndex )    {   return charactersInRoom.remove( intIndex ); }
91
92      /**
93       * Removes the gold pieces in the room.
94       *
95       * @return int Returns the number of gold pieces taken from the room.
96       */
97
```

```
98      public int retrieveGoldPieces() {
99
100         int intGoldPiecesRetrieved  = getGoldPieces();
101         setGoldPieces( 0 );
102
103         return intGoldPiecesRetrieved;
104
105     }
106
107     /**
108      * Display room details on the console.
109      *
110      * @return String Details and descriptions about a room object.
111      */
112
113     public String toString()    {
114
115         String strRoomDetails = "\nCurrent Location:  " + strRoomName + "\n" + strRoomDescription + "\n";
116
117         if( charactersInRoom.isEmpty() == false )   {
118
119             // Lists characters in the room
120             strRoomDetails += "\nThe following characters are in this room:\n";
121             for( Character character : charactersInRoom )   {   strRoomDetails += "\t" + character + "\n";  }
122
123         }
124
125         if( itemsInRoom.isEmpty() == false )    {
126
127             // Lists items in the room
128             strRoomDetails += "\nThe following items are in this room:\n";
129             for( Item item : itemsInRoom )  {   strRoomDetails += "\t" + item + "\n";   }
130
131         }
132
133         // Lists gold pieces in the room (if any exist/remain)
134         if( intGoldPieces > 0 ) {   strRoomDetails += "\nThere are " + Integer.toString( intGoldPieces ) + " gold pieces lying
            on the ground.\n";   }
135
136         return strRoomDetails;
137
138     }
139
140     // Accessor/Mutator methods
141
142     public void setRoomName( String strRoomName )   {   this.strRoomName    = strRoomName;  }
143
144     public String getRoomName() {   return this.strRoomName;    }
145
146     public void setRoomDescription( String strRoomDescription ) {   this.strRoomDescription = strRoomDescription;   }
```

```
147
148     public String getRoomDescription()  {   return this.strRoomDescription; }
149
150     public void setDoor( int intDoorLocation, int intDoorValue )    {   this.intConnectedRooms[intDoorLocation] = intDoorValue; }
151
152     public int[] getDoor()  {   return this.intConnectedRooms;  }
153
154     public int getGoldPieces()  {   return intGoldPieces;   }
155
156     public void setGoldPieces( int intGoldPieces )  {   this.intGoldPieces  = intGoldPieces;    }
157
158     public ArrayList<Item> getItem()    {   return itemsInRoom; }
159
160     public ArrayList<Character> getCharacter()  {   return charactersInRoom;    }
161
162 }
```