```java
import java.util.*;
/*********************************************
 * This class builds rooms for the Game.
 *
 * @author Ben Parsell
 * @version 1.0.0 (11/19/2015)
 *********************************************/
public class Room
{
    /** Describes the room **/
    private String roomDescription;

    /** Item in the room **/
    private Item optionalItem;

    /** ArrayList for rooms **/
    private HashMap <String, Room> neighbors;




    /*******************************************************
     * Overloaded Constuctor
     *
     * @param pDescription variable to describe room
     * @param pItem variable for item in the room
     ******************************************************/
    public Room(String pDescription) {
        // Initialize instance variables
        this.roomDescription = pDescription;

        // Creates hashmap for neighbors
        neighbors = new HashMap <String, Room> ();
    }


    /*******************************************************
     * Default Constructor for Room
     *
     * @param pDescription variable to describe room
     ******************************************************/
    public Room(String pDescription, Item pItem) {
        this.roomDescription = pDescription;
        this.optionalItem = pItem;
        neighbors = new HashMap<String, Room>();
    }


    /*******************************************************
```

```
     * Method to add an item into the game
     *
     * @param i variable for the item in the room
     *****************************************************/
    public void addItem(Item i) {
        this.optionalItem = i;
    }


    /*****************************************************
     * Method to check if player has an item
     *
     * @return returns true of false depending on null
     * value
     *****************************************************/
    public boolean hasItem() {
        // Check for if room has an item
        if(optionalItem != null) {
            return true;
        } else {
            return false;
        }
    }


    /*****************************************************
     * Method to get the description of the room
     *
     * @return roomDescription describes room
     *****************************************************/
    public String getDescription() {
        return roomDescription;
    }

    /*****************************************************
     * Method to get an item in the Room
     *
     * @return optionalItem item within the room
     *****************************************************/
    public Item getItem() {
        return optionalItem;
    }


    /*****************************************************
     * Method to view neighboring rooms
     *
     * @return value for neighboring rooms
     *****************************************************/
    public HashMap getNeighbors() {
        return neighbors;
    }


    /*****************************************************
     * Method to add a neighboring room
     *
     * @param direction value for direction to move
     * @param r value for room name
     *****************************************************/
    public void addNeighbor(String direction, Room r) {
        // Adds neighbor to room r, in that direction
        this.neighbors.put(direction, r);
    }


    /*****************************************************
     * Finds neighbor in given direction
     *
     * @param direction value for direction to move
     *****************************************************/
    public Room getNeighbor(String direction) {
        // Create new room to find neighbor
        Room neighbor = this.neighbors.get(direction);
        return neighbor;
    }


    /*****************************************************
     * Method removes an item from the room
     *
     * @return optionalItem item within the room
     *****************************************************/
    public Item removeItem() {
        // Set item in room to null, to delete
        Item temp = optionalItem;
        this.optionalItem = null;
        return temp;
    }


    /*****************************************************
     * Method to retrieve the full description of a room
     *
     * @return longDescription value for well-described
     * room information
     *****************************************************/
    public String getLongDescription() {
        // Check to see if room has an item, then return text
        if(hasItem()) {
            String longDescription = "You are currently in " + roomDescription + ".\n You see " + optionalItem.getDescription() + ".";
            return longDescription + "\n";
        }

        // If no item, gives room description without item.
        else {
            String longDescription = "You are in " + roomDescription + ".";
            return longDescription + "\n";
        }
    }
}

```