Our goal to reduce coupling demands that, as far as possible, changes to the Room class do not require changes to the Game class. We can still improve this.

Currently, we have still encoded in the Game class the knowledge that the information we want from a room consists of a description string and the exit string:

```
System.out.println("You are " + currentRoom.getDescription());
System.out.println(currentRoom.getExitString());
```

What if we add items to rooms in our game? Or monsters? Or other players?

When we describe what we see, the list of items, monsters, and other players should be included in the description of the room. We would need not only to make changes to the Room class to add these things, but also to change the code segment above where the description is printed out.

This is again a breach of the responsibility-driven design rule. Since the Room class holds information about a room, it should also produce a description for a room. We can improve this by adding to the Room class the following method:

```
/**
 * Return a long description of this room, of the form:
 *     You are in the kitchen.
 *     Exits: north west
 * @return A description of the room, including exits.
 */
public String getLongDescription()
{
    return "You are " + description + ".\n" + getExitString();
}
```

In the Game class we then write

```
System.out.println(currentRoom.getLongDescription());
```

The 'long description' of a room now includes the description string, information about the exits, and may in the future include anything else there is to say about a room. When we make these future extensions, we will have to make changes to only a single class: the Room class.