Object Decomposition and Map / HashMap
How hard was third code review assignment?

A) Easy
B) Moderate
C) Challenging
D) Unreasonable
How long did third assignment take?

A) Less than 2 hours
B) 2 to 4 hours
C) 4 to 6 hours
D) 6 to 8 hours
E) More than 8 hours
Object-oriented Decomposition

- Break large problem into logical pieces (data + code)
- Implement those pieces as classes

- A (good) class can be:
  - A collection of data and routines that share a cohesive, well defined responsibility
  - A collection of routines that provide a cohesive set of services even if no common data is involved.
JSON format

- **Direction:**
  - directionName: String
  - room: String  // a Room’s name

- **Room:**
  - name: String
  - description: String
  - directions: Direction []

- **Layout:**
  - startingRoom: String  // a Room’s name
  - rooms: Room []
{ "startingRoom": "MatthewsStreet",
"rooms": [
{
   "name": "MatthewsStreet",
   "description": "You are on Matthews, outside the Siebel Center",
   "items": ["coin"],
   "directions": [
   {
      "directionName": "East",
      "room": "SiebelEntry"
   }
   ]
},
{
   "name": "SiebelEntry",
   "description": "You are in the west entry of Siebel Center. You can see the elevator, the ACM office, and hallways to the north and east.",
   "directions": [
   {
      "directionName": "West",
      "room": "MatthewsStreet"
   },
   {
      "directionName": "Northeast",
      "room": "AcmOffice"
   },
   {
      "directionName": "North",
      "room": "SiebelNorthHallway"
   }
   ]
}]}
What all does this code need to do?
Java Collection Framework
Map

- Allows lookups from one kind of object to find another object

Map<KeyType, ValueType> myMap =
    HashMap<KeyType, ValueType>();

KeyType key = ...;
ValueType value = ...;
myMap.put(key, value);
ValueType lookup = myMap.get(key);
assert lookup == value;
Map Interface

- **put**(k,v)  
  Associate v with k

- **get**(k)  
  The value associated with k

- **size()**  
  The number of pairs

- **isEmpty()**  
  Whether it is empty

- **remove**(k)  
  Remove the mapping for k

- **clear()**  
  Remove all mappings

- **containsKey**(k)  
  Whether contains a mapping for k

- **containsValue**(v)  
  Whether contains a mapping to v
Command Line Arguments

- Set via Run->Edit Configuration