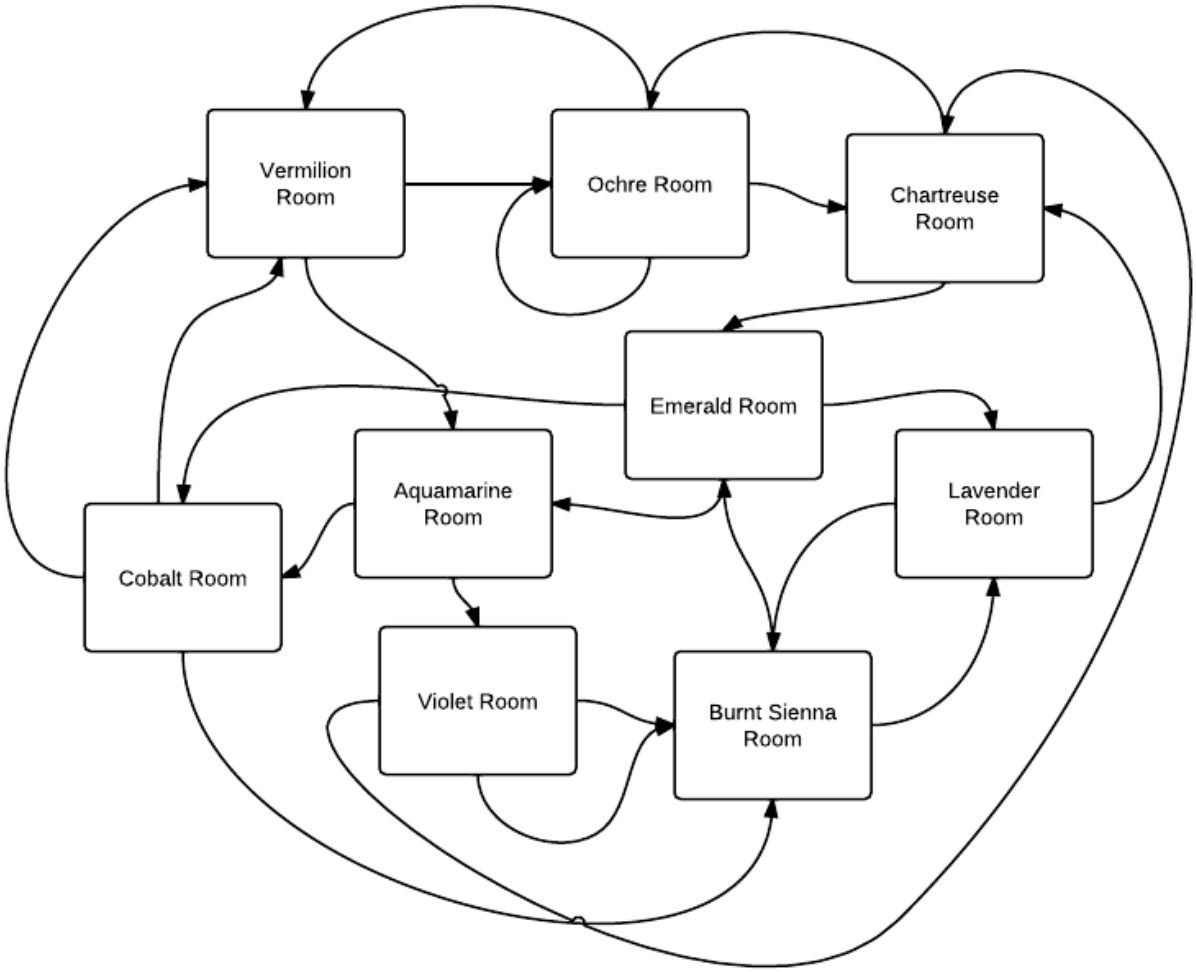
Rules of Simple Quest

Consider this sample map:



1) The world is a bunch of rooms connected by one-way corridors. Being one-way, corridors are either "in-bound" or "out-bound", depending on the perspective of which room you're in.

2) Each room has up to 4 doors, in the 4 cardinal directions.

3) Some doors only open one way, other doors open both ways, but being imbued with ornery magic, no door will open onto the same corridor from both directions. I.e., if you go from Room A to Room B via a corridor, you should never be able to turn around and go back into Room A via that same door.

4) From any given room, you cannot tell which doors are connected to "in-bound" corridors. You can tell which doors are "out-bound", by trying them in turn and seeing which of them open. Note:

No rooms are dead-ends (featuring only "in-bound" routes).

5) The game is turn-based. During each turn, you must try to go north, south, east, or west, or pick up a gem (see below). If a given direction does not work, you can try again. A failed move attempt / retry does *not* consume a turn. If the door leads to a corridor, opening the door and traveling to the other room together consume a single turn.

6) Every 4 turns, you must rest for 1 turn. (I.e., every 5th turn is a resting turn).

7) There is a Grue in the dungeon with you, who spawns upon your arrival, far away from your start point. Any time you rest, you stop making noise, which frees the Grue to move, one corridor per turn. The Grue always knows the shortest path to reach your resting spot. If the Grue attacks you, you're dead. If you die, you lose your gems and respawn randomly.

8) If you actively enter a room containing the Grue, he will drop a gem and flee randomly one room away. (To attack you he must be the one entering the room.) Thus, when you see a gem, you can surmise that a Grue is very nearby. You will automatically pick up the dropped gem (and this does not consume another turn).

9) Your goal is to collect 5 gems, then make your way to a target room (e.g., the Cobalt Room), at which point you are transported home. (There should be some clear indication of which room is the teleport room, perhaps it has a glowing dais in the center).

==========

Further Implementation Notes

* We prefer that you use Ruby or a similar dynamic language for the implementation. (e.g., Python, JavaScript within Node.js, etc)
* Please use git to track your progress. Please make regular small commits so we can see your thought process unfold.
* Please DO NOT upload to Github, however. We'd like to avoid having to write new exercises every week :)
* Don't worry about graphics. Zork-style text-based is fine. However if you want to make a graphical version, go to town!
* Please express the map and the teleport room via a configuration file (make the map editable, do not hardcode it).
* If you want to make the game a bit easier, you can scatter a few gems around randomly when you initialize the map.
* Also, if you want to make the game a bit easier, you can warn the player by making the Grue detectable. E.g., if the Grue is 1 room away, the system can announce "You can smell the Grue! It is nearby!"
* Please implement the sample map.

* Bonus: unit-test the world with what you deem to be a reasonable degree of thoroughness.