Recursive Goblin function:

In Goblin.cpp:

bool isPath(int row, int col, int steps);

Non-trivial Algorithms:

void Dungeon::createRooms();

1. Randomly generate the width, height, and location of the top left corner of the room
2. If anything is out of bounds or if any rooms intersect, keep generating values
3. If not, add the coordinates of the top left corner to a vector of pairs
4. Create the rooms by changing # to spaces
5. Make another room until m\_numRooms is done (will generate between 4 and 6 rooms)
6. Sort the vector of pair by column
7. Make corridors by creating spaces from one top left corner to another, until there are 4 corridors. Connect by order of increasing column index.
8. First move up/down, then move left/right until rooms are connected.

bool isPath(inte row, int col, int steps);

1. Base cases:
   1. If the row or col are out of bounds, stop recursing down that path
   2. If the goblin found the player, return that there does exist a path
   3. If the current position is blocked by a wall or monster, stop recursing down that path
   4. If the goblin has reached the maximum number of steps, stop recursing down that path
   5. Otherwise, recursively call the path function to the up, down, left, and right directions









