The recursive function

I tried to design a recursive function for the Goblin to check if the player is within 15 steps to reach or not. The function is named as goblinSmell in the goblin class.

The function is implemented by using the algorithm we used to solve the maze.

And in goblin’s doSomething function, I called goblinSmell to check if the goblin can move or not, the use the getDirection function in monster class to determine the direction of the action.

Design

I divided the program into 4 components, Game class, Dungeon class, Actor class, and object class.

For the **game class**, it contains necessary information to start a new game, and it has a dungeon class member as its private member.

For the **dungeon class**, it generates the map for the game. It also takes player and object and monster as its private member so that they can be displayed on the map.

For the **actor class**, it contains two direct subclasses: player class and monster class. The actor class takes dungeon as its private member so that it can influence the map. The player class maintains a inventory, which is a list of object class.

For the **monster class**, it contains 4 different kinds of monster classes: Goblin, Bogeyman, Snakewoman, Dragon.

For the **object class**, it contains 2 different kinds of classes: weapon and scroll. Also, weapons have subclasses as mace, short sword, long sword, magic axe and magic fangs of sleep. Scrolls have scroll of teleportation, scroll of improve armor, scroll of enhance health,

scroll of enhance dexterity 5 sub classes.

The game class is used to generate the game, and also, get the number of Goblin smell distance for the game.

The dungeon class is used to create map, record all the necessary information for the map (e.g. the object list, the monster list, the current level etc.)

The player class is what the player needs to do, and also we have monster classes to implement monsters’ specific behaviors.

The object class records mainly the name and action strings for different objects. Their specific functions are implemented in the player’s use objects functions (since only player can use the different objects).

Non-trivial algorithms for some functions

Dungeon class:

**setObject**

start to search the object list

generate a pair of random number for the position of one object

check it is a weapon or a scroll

place it accordingly

place the stair if the level is not 4

place the golden idol if the level is 4

**deleteDungeon** (in each new level)

increase level

generate the new number of regions

create map

add current player to the new map

clean the old objects and monsters

add new objects and monsters.

**display**

clearscreen

clean all the corpse

set golden idol or stair

set objects

set monsters

set player

display the map

display the statistics

display the gaming strings (if any)

actor class

**attack**

get attacker point and defender point

generate random number and check if attacker can hits the defender

if the weapon is not magic fangs of sleep

hit the defender

check if the defender is dead or not

if the weapon is the magic fangs of sleep

hit the defender

check if it can make the defender sleep

if it cannot hit

return

player class

**playerMove**

check if the player is dead

check if the player is asleep

random recover one HP

get the direction

in each direction, check if it can attack monster

if so, attack; otherwise, just move

**pickobject**

check if the object is golden idol

if so, you won

check if the inventory size is bigger than 25

if so, we cannot pickup

put the object into the inventory

delete it from the map’s object list

monster class

**monsterDrop**

get the monster’s current position

if it is a snakewoman

drops fang at the current position (with probability)

if it is a bogeyman

drops magic axe at the current position (with probability)

if it is a dragon

generate a random scroll

drops that scroll at the current position

if it is a goblin

drops magic axe or magic fangs at the current position (with possibility)

Known Bugs:

The goblin cannot move as expected. Although I used recursion function, the goblin cannot move accordingly. It does move, however, with strange movements.