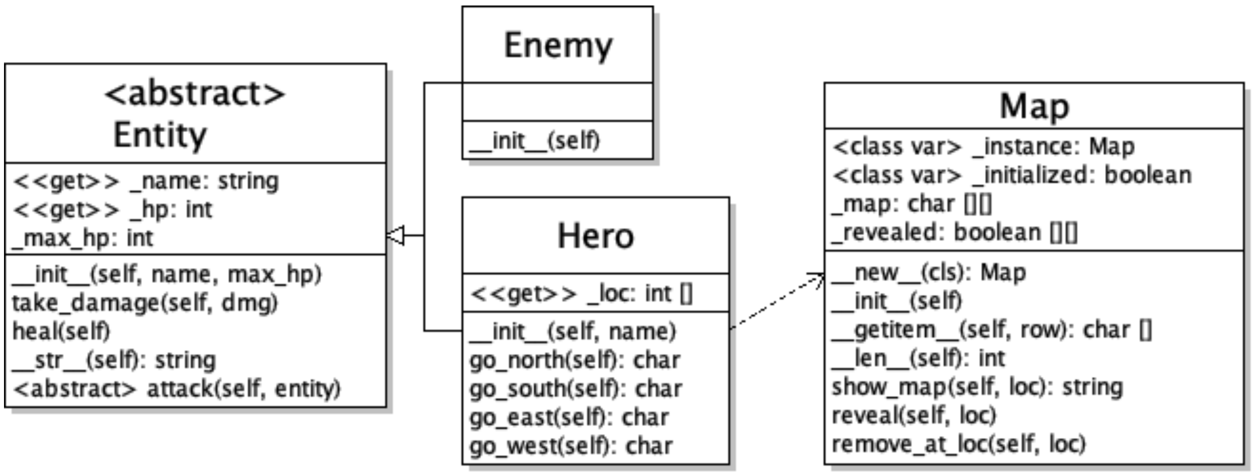
CECS 277 – Lab 10 – Singleton

Dungeons and Monsters

Create a program that allows the user to wander through a haunted dungeon maze and fight monsters that they encounter as they explore. The user wins if they reach the dungeon's exit alive. Use the following UML diagram and the class descriptions below to create your program.



Classes:

1. Entity – abstract class - describes a character in the game.
    a. __init__(self, name, max_hp) – initializes each of the instance variables
    b. take_damage(self, dmg) – subtracts the dmg from the hp, but does not allow the hp to drop below 0.
    c. heal(self) – restores the entity's hp to max_hp.
    d. __str__(self) – returns a string in the format 'Name\nHP: hp/max_hp'.
    e. attack(self, entity) – abstract method (no code) that all entity subclasses will override to attack and do damage to the opposing entity.
2. Hero – extends entity - the user's character
    a. __init__(self, name) – initializes the name and max_hp using super, sets the hero's starting location to row=0, col=0.
    b. attack(self, entity) – hero attacks the enemy – randomize damage in the range 2-5, the enemy should take the damage and the method should return a string representing the event.
    c. go_north/south/east/west(self) – update the hero's location by adding or subtracting 1 to the row or column, but only if that location is within the bounds of the map (between 0 and the len(map)-1). If it is, return the character at that location, if it isn't, return an 'o' to signify that the direction is out of bounds.
3. Enemy – extends entity - monster character that the hero encounters in the maze.
    a. __init__(self) – randomizes a name from a list of names (ex. 'Goblin', 'Vampire', 'Ghoul', 'Skeleton', 'Zombie, etc) and randomizes the monster's hp (4-8).
    b. attack(self, entity) – enemy attacks hero – randomize damage in the range 1-4. The hero should take the damage and the method should return a string representing the event.

4. Map – singleton – the map of the dungeon maze.
    a. __new__(cls) – if the map hasn't been constructed, then construct it and store it in the instance class variable and return it. If it has, then just return the instance.
    b. __init__(self) – create and fill the 2D map list from the file contents. Create and fill the 2D revealed list with all False values. The map stores the contents of the file and the revealed list is used to determine whether the contents of the map are displayed or not ('x' if not displayed).
    c. __getitem__(self, row) – overloaded [] operator – returns the specified row from the map. (Note: this operator can be used to access a row (ex. m[r]) or can be used to access a value at a row and column (ex. m[r][c]).
    d. __len__(self) – returns the number of rows in the map list. (Note: if you want to know the number of rows, use len(m), if you need the number of columns, use len(m[r])).
    e. show_map(self, loc) – returns the map as a string in the format of a 5x5 matrix of characters where revealed locations are the characters from the map, unrevealed locations are 'x's, and the hero's location is a '*'.
    f. reveal(self, loc) – sets the value in the 2D revealed list at the specified location to True.
    g. remove_at_loc(self, loc) – overwrites the character in the map list at the specified location with an 'n'.
5. Main – prompt the user to enter their name, then construct the hero and a map object. Create a loop that repeats until the hero dies, the hero finds the finish, or the user quits the game. Present a menu that allows the user to choose a direction to move in (north, south, east, west), move the hero in that direction, reveal that spot, and then present the encounter at that location as follows:
    a. 'm' – monster – construct an enemy and display its information. Create a loop that allows the user to either attack or run away. If they attack, the hero attacks the monster, and if the monster has hp left, then the monster attacks back. If the monster is dead, then display a message and remove the 'm' from the map. If the user chooses to run away, then choose a random direction to run in (reveal but don't present the encounter for the new location) ('m' should remain on the map).
    b. 'o' – invalid direction – display a message stating that they cannot move that direction.
    c. 'n' – nothing – display a message stating that this room is empty.
    d. 's' – start – display a message that they wound up back at the start of the dungeon.
    e. 'i' – item room – display a message stating that they found a health potion. Heal the hero and remove the 'i' from the map so they can't get it again (not required, but you can add a check to see if the hero has full hp, if they do, then you can leave the 'i' on the map to save it for later).
    f. 'f' – finish – display a congratulatory message stating that they found the way out of the maze and won the game.

Example Output:

```
What is your name, traveler? Jack
Jack
HP: 25/25
* x x x x
x x x x x
x x x x x
x x x x x
x x x x x

1. Go North
2. Go South
3. Go East
4. Go West
5. Quit
Enter choice: 1
You cannot go that way...

Jack
HP: 25/25
* x x x x
x x x x x
x x x x x
x x x x x
x x x x x

1. Go North
2. Go South
3. Go East
4. Go West
5. Quit
Enter choice: 2
There is nothing here...

Jack
HP: 25/25
s x x x x
* x x x x
x x x x x
x x x x x
x x x x x

1. Go North
2. Go South
3. Go East
4. Go West
5. Quit
Enter choice: 2
You encounter a Vampire
HP: 8/8
1. Attack Vampire
2. Run Away
Enter choice: 2
You ran away!

Jack
HP: 25/25
s x x x x
* x x x x
m x x x x
x x x x x
x x x x x

1. Go North
2. Go South
3. Go East
4. Go West
5. Quit
Enter choice: 3
There is nothing here...

Jack
HP: 25/25
s x x x x
n * x x x
m x x x x
x x x x x
x x x x x

1. Go North
2. Go South
3. Go East
4. Go West
5. Quit
Enter choice: 2
There is nothing here...

Jack
HP: 25/25
s x x x x
n n x x x
m * x x x
x x x x x
x x x x x

1. Go North
2. Go South
3. Go East
4. Go West
5. Quit
Enter choice: 2
You found a Health Potion!  You
drink it to restore your health.

Jack
HP: 25/25
s x x x x
n n x x x
m n x x x
x * x x x
x x x x x
```

```
1. Go North
2. Go South
3. Go East
4. Go West
5. Quit
Enter choice: 3
There is nothing here...

Jack
HP: 25/25
s x x x x
n n x x x
m n x x x
x n * x x
x x x x x

1. Go North
2. Go South
3. Go East
4. Go West
5. Quit
Enter choice: 3
You encounter a Goblin
HP: 4/4
1. Attack Goblin
2. Run Away
Enter choice: 1
Jack attacks a Goblin for 4 damage.
You have slain a Goblin

Jack
HP: 25/25
s x x x x
n n x x x
m n x x x
x n n * x
x x x x x

1. Go North
2. Go South
3. Go East
4. Go West
5. Quit
Enter choice: 2
You encounter a Skeleton
HP: 8/8
1. Attack Skeleton
2. Run Away
Enter choice: 1
Jack attacks a Skeleton for 4
damage.
Skeleton attacks a Jack for 2
damage.
1. Attack Skeleton
2. Run Away
Enter choice: 1
Jack attacks a Skeleton for 3
damage.
Skeleton attacks a Jack for 1
damage.
1. Attack Skeleton
2. Run Away
Enter choice: 1
Jack attacks a Skeleton for 2
damage.
You have slain a Skeleton

Jack
HP: 22/25
s x x x x
n n x x x
m n x x x
x n n n x
x x x * x

1. Go North
2. Go South
3. Go East
4. Go West
5. Quit
Enter choice: 3
Congratulations! You found the
exit.
Game Over
```

Notes:

1. You should have 6 different files: main.py, entity.py, enemy.py, hero.py, map.py, check_input.py.
2. Check all user input using the get_int_range function in the check_input module.
3. Do not create any extra methods, attributes, functions, parameters, etc.
4. Please do not create any global variables (besides the singleton map), or use attributes globally (ie. do not access any of the attributes using the underscores).
5. Use docstrings to document each of the classes, their attributes, and their methods.

6. Place your names, date, and a brief description of the program in a comment block at the top of your main file. Place brief comments throughout your code.
7. When you run away from a monster the 'm' stays on the map. If you return to that same location, it will randomize a new monster (ie. it may not be the exact same monster).
8. Thoroughly test your program before submitting:
    a. Make sure that the map is read in correctly (start is at 0,0).
    b. Make sure that the hero can move throughout the maze and cannot move out of bounds in any direction.
    c. Make sure that the hero can fight or run away from an enemy.
    d. Make sure that the hero runs away in a random direction and the new room is revealed but not activated (hero just moves there).
    e. Make sure that the hero does correct damage to the monster and the monster does correct damage to the hero.
    f. Make sure that a defeated enemy does not attack the hero back.
    g. Make sure that the hero is fully healed when they enter an item room.
    h. Make sure that the game ends when the finish is reached.

Dungeons and Monsters Rubric – Time estimate: 5 hours

Dungeons and Monsters **10 points**	Correct. 2 points	A minor mistake. 1.5 points	A few mistakes. 1 point	Several mistakes. 0.5 points	No attempt. 0 points
Entity & Enemy classes (sep. files)**:** 1. Entity is abstract with abstract attack method. 2. Entity has properties and methods. 3. Enemy class inherits from Entity and overrides attack method. 4. Enemy randomizes name and hp.					
Hero class (separate file)**:** 1. Inherits from Entity class. 2. Hero constructs the map when any method requires access to it. 3. Hero has move methods that update the hero's location and returns the character in the map at that location. 4. Hero overrides attack method.					
Map class (separate file)**:** 1. Singleton with class variables and overridden new method. 2. init method reads in file. 3. show_map method returns a string with the formatted map with 'x's in unrevealed spaces, map values in revealed spaces, and a '*' at the hero's location. 4. Overrides [] and len().					

Main file (in separate file)**:** 1. Constructs Hero and Map. 2. Allows user to choose a direction to move in (cannot move out of bounds). 3. Error checks all user input. 4. Displays map with hero's location. 5. Monster attacks in 'm' room. Hero can attack or run away. Fight continues until monster is destroyed. 6. Hero heals in 'i' room. 7. Map is updated when hero destroys monster or drinks potion. 8. Game repeats until user quits, finish is found, or hero dies.					
Code Formatting: 1. Correct documentation. 2. Meaningful variable names. 3. No exceptions thrown. 4. No global variables (other than singleton) or accessing attributes directly. 5. Correct spacing.					