Alan Nelson
12/8/2024
CS5001

# Final Project Report

## Project Description

For my final project, I created an interpretation of Pathfinder 2E that can be played inside the Python terminal. The program consists of eleven classes: a parent class, Creature, which contains common attributes and methods used by the other ten, either directly or to calculate/generate their own attributes and methods; four player classes, creating objects with attributes and methods to enable the player to control a 5th level Fighter, Wizard, Cleric, and Rogue, each with their own abilities and interactions; and six enemy classes, creating objects for a Goblin, Hobgoblin Soldier, Orc Warrior, Orc Commander, Hobgoblin General, and Young Green Dragon, each again with their own ability methods and unique artificial intelligence function.

The game application consists of a series of encounter loops. The player is prompted to provide names for each player character which are stored in each as an attribute, then four encounters are started in series: versus four goblins and a hobgoblin soldier; versus two orc warriors and an orc commander; versus four goblins and a hobgoblin general; and versus the young green dragon alone. Each encounter's enemies are stored in a list, which is combined with the list of player characters and used to roll initiative to determine turn order. The encounter loop iterates through the initiative list, prompting each creature in turn to take their turn. On a given turn, the creature has 3 actions it can spend; player characters are given a list of options to choose from, while enemies have their actions determined algorithmically. Once a creature is out of actions, its turn ends. If either team (player characters or enemies) runs out of hit points, the encounter ends; if the player was the one to run out of HP, the game ends. Otherwise, it proceeds to the next encounter or, if the final encounter is successfully completed, prints a message congratulating the player.

Referring to the list of targets and goals I set in Final Project Planning Document 1, I accomplished all of my must-have features, but was not able to implement any of the optional goals due to time. With the benefit of hindsight, I was significantly underestimating at the time the amount of work even the basic features I planned would take.

## Changes From Planning Document #2

I opted to use lists instead of dictionaries to access the player abilities; I ended up needing them to be ordered for consistency purposes. Instead, the function that converts the list into the information displayed to the player converts the lists into a dictionary briefly while restructuring the information for display.

I ended up moving most of the attribute calculation from child classes into Creature instead; since everything needed the calculated attributes like saves and skill proficiencies it made more sense in practice to have them there rather than in the child classes. The only exceptions ended up being the various bonuses to attack rolls, which have a dummy value in creature for testing purposes but are calculated properly (or in the case of monsters just assigned the desired value) in the child class.In retrospect I don't know why I did it this way; it made sense at the time. I also abandoned the idea of having the artificial intelligence functionality in separate classes from the monsters, instead making ai() a function present in each monster that does different things depending on the user.

## Reflection

In the course of completing this project I have gained a ton of familiarity with working with classes and objects and with Python in general. Part of that is just a result of volume, as this project works out to be more lines of code than all nine projects from this semester combined.

Were I to do it over again there are several changes I would make. Foremost among them, I would refactor the take_turn() method found in the four player classes to be class-agnostic and properly testable; the take_turn() methods were among the few things that were not tested, along with the contents of dice.py (which all just perform random number generation between 1 and the dice number; nothing to test), helper_functions.py (which contains the function for displaying the player character's options, which has no output except for the result of player input: this is here instead of in application.py because I needed it for take_turn() in each class and couldn't do a circular import; and a function for selecting a random target that again just does random number generation), and application.py.

We discussed take_turn() in office hours, and since then I've been brainstorming how I would refactor it, but there simply wasn't time to implement it. My current idea consists of pulling take_turn() out of each class, combining them into one, and putting the result into helper_functions.py, at least keeping the untestable functions in one place. I'd then pull the target collection and action verifications into separate functions to streamline it.

Another refactor I would've liked to do would be to set up a basic saving throw method in creature and redo methods like the Wizard's fireball and Dragon's breath weapon to refer to it instead of being bespoke each time. Not doing this was a byproduct of writing the Wizard class first after Creature, so by the time it occurred to me that I could make things easier on myself by doing so I had already implemented almost all of the cases where it would be needed.

Beyond that, there are smaller changes; moving raise_a_shield() into Creature since it ended up being repeated in multiple classes, functions replacing blocks of code that ended up recurring in multiple places, and the like.

## Acknowledgements/Citations

- https://stackoverflow.com/questions/8853966/the-inheritance-of-attributes-using-init
- https://stackoverflow.com/questions/25046306/sort-a-2d-list-first-by-1st-column-and-then-by-2nd-column
- https://www.geeksforgeeks.org/inheritance-in-python/
- https://www.geeksforgeeks.org/python-call-parent-class-method/

- https://www.geeksforgeeks.org/sorting-list-of-lists-with-first-element-of-each-sub-list-in-python/
- https://www.geeksforgeeks.org/python-lambda-anonymous-functions-filter-map-reduce/
- https://www.geeksforgeeks.org/sleep-in-python/
- time.sleep from office hours with Professor Domino

Archives of Nethys pages used as reference:

- GM Screen: https://2e.aonprd.com/GMScreen.aspx
- Cleric: https://2e.aonprd.com/Classes.aspx?ID=33
  - https://2e.aonprd.com/Spells.aspx?ID=1498 - Divine Lance spell
  - https://2e.aonprd.com/Spells.aspx?ID=1451 - Bless spell
  - https://2e.aonprd.com/Spells.aspx?ID=2033 - Sudden Blight spell
  - https://2e.aonprd.com/Spells.aspx?ID=1524 - Fear spell
  - https://2e.aonprd.com/Spells.aspx?ID=1554 - Heal spell
  - https://2e.aonprd.com/Feats.aspx?ID=4646 - Healing Hands feat
  - https://2e.aonprd.com/Actions.aspx?ID=2316 - Raise a Shield action
  - https://2e.aonprd.com/Weapons.aspx?ID=362 - Mace item
  - https://2e.aonprd.com/Feats.aspx?ID=4642 - Deadly Simplicity feat
- Creature:
  - https://2e.aonprd.com/Conditions.aspx?ID=19 - Frightened condition
  - https://2e.aonprd.com/Conditions.aspx?ID=58 - Off-Guard condition
  - https://2e.aonprd.com/Rules.aspx?ID=2416 - Hidden rules
  - https://2e.aonprd.com/Conditions.aspx?ID=79 - Hidden condition
  - https://2e.aonprd.com/Rules.aspx?ID=2539 - Initiative rule
  - https://2e.aonprd.com/Actions.aspx?ID=2301 - Seek action
  - https://2e.aonprd.com/Feats.aspx?ID=5125 - Battle Medicine feat
  - https://2e.aonprd.com/Actions.aspx?ID=2399 - Treat Wounds action
  - https://2e.aonprd.com/Actions.aspx?ID=2395 - Demoralize action
  - https://2e.aonprd.com/Actions.aspx?ID=2404 - Hide action
  - https://2e.aonprd.com/Skills.aspx?ID=24&General=true - Recall Knowledge action; really just used the name of this one, the Seek action I implemented used on non-hidden targets is closer to the actual effect
  - https://2e.aonprd.com/Actions.aspx?ID=2382 - Trip action
  - https://2e.aonprd.com/Actions.aspx?ID=2370 - Tumble Through action
  - https://2e.aonprd.com/Feats.aspx?ID=4920 - Tumble Behind feat
  - https://2e.aonprd.com/Actions.aspx?ID=2306 - Strike action
- Fighter: https://2e.aonprd.com/Classes.aspx?ID=35
  - https://2e.aonprd.com/Feats.aspx?ID=4775 - Vicious Swing feat
  - https://2e.aonprd.com/Feats.aspx?ID=4782 - Intimidating Strike feat
  - https://2e.aonprd.com/Feats.aspx?ID=4795 - Swipe feat
  - https://2e.aonprd.com/Weapons.aspx?ID=379 - Greatsword item
- Rogue: https://2e.aonprd.com/Classes.aspx?ID=37
  - https://2e.aonprd.com/Feats.aspx?ID=4921 - Twin Feint feat
  - https://2e.aonprd.com/Weapons.aspx?ID=398 - Shortsword item

    - https://2e.aonprd.com/Feats.aspx?ID=4930 - Dread Striker feat
- Wizard: https://2e.aonprd.com/Classes.aspx?ID=39
    - https://2e.aonprd.com/Spells.aspx?ID=1509 - Electric Arc spell
    - https://2e.aonprd.com/Spells.aspx?ID=1539 - Frostbite spell
    - https://2e.aonprd.com/Spells.aspx?ID=1565 - Ignition spell
    - https://2e.aonprd.com/Spells.aspx?ID=1671 - Shield spell
    - https://2e.aonprd.com/Spells.aspx?ID=1550 - Gust of Wind spell
    - https://2e.aonprd.com/Spells.aspx?ID=1536 - Force Barrage spell
    - https://2e.aonprd.com/Spells.aspx?ID=1577 - Invisibility spell
    - https://2e.aonprd.com/Spells.aspx?ID=1721 - Thunderstrike spell
    - https://2e.aonprd.com/Spells.aspx?ID=1530 - Fireball spell
    - https://2e.aonprd.com/Spells.aspx?ID=1553 - Haste spell
    - https://2e.aonprd.com/Weapons.aspx?ID=358 - Dagger item
- https://2e.aonprd.com/Monsters.aspx?ID=133 - Young Green Dragon
- https://2e.aonprd.com/NPCs.aspx?ID=3025 - Goblin Commando
- https://2e.aonprd.com/Monsters.aspx?ID=3055&Weak=true&NoRedirect=1 - Hobgoblin General (weak)
- https://2e.aonprd.com/Monsters.aspx?ID=3053&Elite=true&NoRedirect=1 - Hobgoblin Soldier (elite)
- https://2e.aonprd.com/NPCs.aspx?ID=3132&Elite=true&NoRedirect=1 - Orc Commander (elite)
- https://2e.aonprd.com/Monsters.aspx?ID=325&Elite=true&NoRedirect=1 - Orc Warrior (elite)