PYTHON DUNGEON CRAWLER

25/08/2023

SDV602 Assessment 1

Kira Byrne

ID #13509995

Table of Contents

Introduction 2
Game Design 2
Diary 3
Data Types & Comprehensions 5
Python vs JavaScript 6
References 8

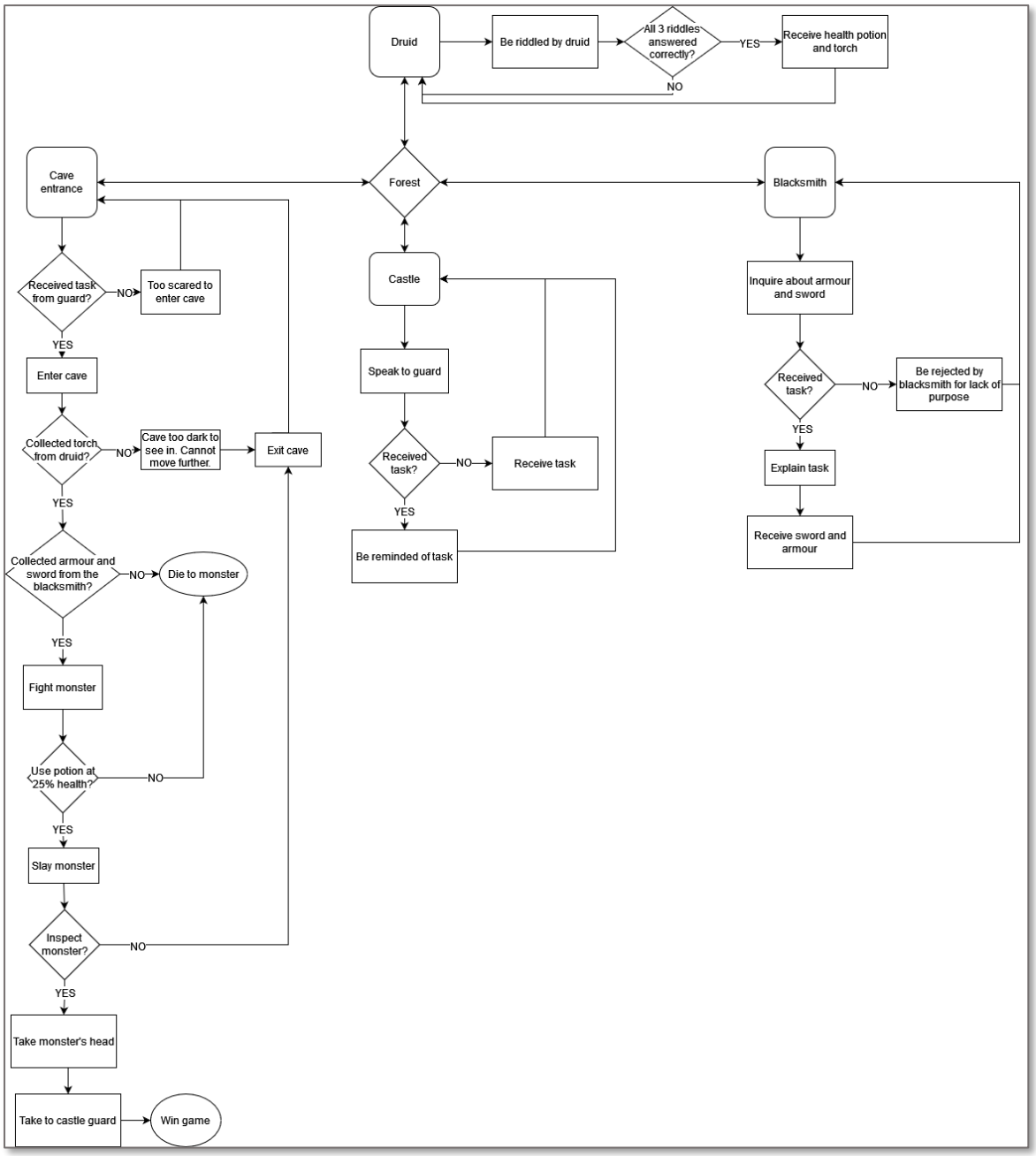
Introduction

In this assignment, I will be exploring the art of programming with the Python coding language. This exploration will cover the history, philosophy, and execution of Python programming as well as a comparison to JavaScript. The platform for this practice will be to create a simple turn-based dungeon crawler style game that takes specific text input and produces text and image output. It will have a set of systems that will allow the player to fight monsters, maintain a score, and an inventory.

As this is my debut in Python programming, I have recorded my progress through diary entries that will discuss my design, tasks, challenges, and solutions. At the end, these notes will be summaries in a video entry along with a walkthrough of the final game.

Game Design

The game will follow the story of an aspiring warrior who is tasked with investigating a cave to the west of the forest and slay whatever is causing a disturbance to the people of the area. Before they can make progress, they must visit the druid and the blacksmith to receive a set of items that will help them on their quest. Each area will either have a condition or a meet a condition to progress the player through the story. This way every area has its purpose and can be utilised to practice varying programming structures. Below is a drafted gameplay flowchart, outlining each major area, condition, and story line:



Diary

July 17th, 2023

Today was the first day of the semester, and the first class was spent on an introduction to Python. I've been excited to learn this language and have done some study on it over the break. So far, it seems like a simpler version of JavaScript that used more natural language. I feel that opinion will change very quickly.

After installing the latest version of python and calibrating Visual Studio Code to act at my development platform, we started on our introductory task which was to download a .py file that Todd had made and attempt to run it. In my python practice repository, I made a virtual environment where development and testing can occur (INSERT REF ON VENVs), installed the PySimpleGUI package for the application to appear, and ran the code. Todd had created a simple game within the

GUI that showed descriptive text, a corresponding image, a input text box, and two buttons. The player (being me) was “in a forest” and could enter either “North” to go to a cave or “south” to go a castle. It played just like the old DOS based games of the 80s and early 90s.

Looking at the code, it was relatively easy to see each functionality of the game. The GUI set up, the image calling, the argument parsing, and more. What we discussed in class was the creation of dictionaries and indexes and their relationships to each other. Back to the code, I could see that each section of the game existed as a variable (i.e. forest, cave, castle). Each of these variables acted as a dictionary of other related variables contained within curly braces, similar to JavaScript. Each area in the game had a “story” which contained the descriptive text of the section and arguments that could be parsed to move from one page to the other, in this case, north or south.

After my quick analysis of the code, I dove in and attempted to add some text to the “story” strings in each page. After running the code, I quickly realised that the GUI was not reactive and the text did not wrap, so I changed the GUI size from 305x200 to 400x200 and made sure to break the text up using “\n”. My added text was now visible and readable. By this time, class was over.

My next challenge is to see about adding new sections and images. From there I can start making changes to the game to make it my own dungeon crawler for the first assignment.

July 24th, 2023

I began to formulate a game design for my version of the dungeon crawler. I had the idea of a complex story of a mead maker going on a quest to make a brew worthy of the gods, but it was too much for my novice coding abilities. Scrapping this, I went with a simpler and more traditional fantasy story.

The player is known as Angmauða, an aspiring warrior in a quaint and ancient tribal land. Angmauða begins their journey in the forest cross-roads and it guided southwards to the castle where a house thane will speak with them. The thane reports that loud bellowing has been heard coming from the west of the forest. He suspects it is a vicious monster that poses a threat to the village. Angmauða is given the task of investigating the noise, for which they will be rewarded with a title of drengr (official warrior). The guard warns that Angmauða cannot do this in their current state and must gather resources from the druid and the blacksmith before going into battle.

Angmauða must travel north of the forest to greet the druid who will quiz them. Answering all three questions correctly will reward the player with a potion that will restore them to full health. This will be placed in their inventory.

To the east of the forest lives the blacksmith. From her, the player must convince the blacksmith to give them a set of armour and a weapon for free because they’re on a noble quest to save the village. Once the smithy is convinced, she will arm the player and provide them the materials for making a torch.

With the potion, armour, and weapon secured, Angmauða can now move into the cave west of the forest. The cave will be dark and will need lighting to navigate. The player will be prompted to take out the torch. On inspection of the cave, Angmauða will come across a very small frog. It will croak loudly, echoing deep from the cave then hopping away. Angmauða laughs and begins to sheath their sword before noticing two glowing red eyes peering from the shadows. A large and angry cave troll slowly approaches with a low growl. The player enters a combat state where they will face off against the troll.

I plan for the troll to have higher damage points than the player can take, even armoured so that when the time is right (at 25% health) they can use their turn to take the potion and regain their health to finish off the troll.

From the death of the troll, the player will receive its dismembered head which they can take back to the longhall and present it to the thane. The player will then enter the longhall and be congratulated by the Jarl and given the title of drengr.

July 30th, 2023

I began to add new areas in the game. I first added to the story dictionary, testing the abilities of a new direction going east from the forest. Following the map I created last week, going east would take the player to the blacksmith. After writing the index for east of the forest, the corresponding story text, and the movement options from there, I realised that I had to create "east" as an ability in the game play arguments index.

I copied the layout of the north and south sections of the code to make east and west, also setting the parameters to limit that direction to the story areas of the game so the player can't move further east or west than allowed. Testing the code showed that the new arguments worked, but I still can't get any images to work other than Todd's originals. Even converting them to .png still reads the error "image data cannot be read". I've looked up tutorials and they say I have to use the Pillow add on to use other file types but it doesn't say anything on .png files not being read. All tips are coming from particular cases so I'll have to wait for Todd's guidance.

August 1st, 2023

List of movements:

North, South, East, West, Enter, Talk, Fight, Attack, Potion, and Look.

I asked Todd about how to create game states and he talked about conditions. I plan for the design of the game to be somewhat open-world but the player cannot progress without following the story as guided in the game text. Todd's explanation on conditions helped me to understand how I can implement the game states so that the player can only progress when meeting certain requirements. For example, the player begins in the forest and through the text, are guided to move south to the castle to receive their task to search the west for the monster. If the player chooses to ignore the castle and move west, they will find the monster's cave but will not be allowed to enter because they have not received their task yet. I plan to add some explanatory text such as *"The foreboding atmosphere frightens you. There is no reason you should yet submit yourself to the unknown horrors within this cave."*

Data Types & Comprehensions

Python uses a multitude of data types to define variables for programming (PanKaj, 2022). In this project, focus has been pulled to understand the following data types and how to apply them to the game.

Dictionary – A dictionary is a mutable data structure that is used as a storage space for information (Simplilearn, 2022). Similar to an array of objects in JavaScript, a dictionary is a set of unique identifying keys that call back a designated item. In the dungeon crawler game, a dictionary is used

to store a set of places in the game. Each game place is called with its key title, e.g., 'Forest', 'Cave', 'Castle', etc. to display the objects within that key, including story text, and image, and applicable player input.

Lists – Similar to arrays in JavaScript, lists and ordered storage structure that can store multiple instances of data at one time (Programiz, 2021). Lists can store many data types like strings, integers, and decimals. Lists can also be empty which are essential for loop creation so that the data created in the loop has a storage space when executed, like using an empty glass to pour water into. A list is used in the dungeon crawler to store items that the player can store and use in their inventory.

List Comprehensions – A list comprehension is used to define a set of expressions that can each be looped over in a loop using parameters to create a new list (GeeksForGeeks, 2023). A list comprehension is used in the dungeon crawler game to define the list of items in the player's inventory as individual items that can be added and a removed in a loop.

Sets – Similar to lists, sets are used to store multiple data items. The main difference between the two being that lists are ordered whereas sets are unordered and cannot be indexed like list items can. In the dungeon crawler, a list is used to store the valid vocabulary tokens used to parse commands during the game.

Tuples – Tuples are another storage-based data type in python, similar to lists, sets, and dictionaries. The main difference being that items within a tuple cannot be changed, they are immutable. The items in a tuple have a defined order that allows duplicates (W3Schools, 2023).

Python vs JavaScript

As an absolute beginner in programming, this is my first experience with using python. The only other programming language I have used has been JavaScript. To compare the two most popular languages, I have chosen to highlight the advantages and contexed of each.

The first advantage I noticed was the beginner friendliness of Python. The use of more natural language and the formatting structure of Python far easier to follow, replicate, and learn from than JavaScript. I would consider this a great strength over JavaScript from the perspective of someone who is new to coding.

The ability to interact with the program during the writing process through direct communication with the interpreter (Sudhan, 2022) has been incredibly useful during the development and testing phase of the game.

The libraries we are using for Python seem more temperamental and clunky compared to JavaScript libraries. I have game-building experience in JS with making a command line version of minesweeper using the node.js, readme.js and inquirer.js libraries. I had little to no issue or bugs with the functionality of these but so far, PySimpleGUI has been somewhat difficult to use, namely with adding images and some random crash events. I would like to explore more about Python libraries as I progress, and hopefully my first impressions will be proven wrong.

Aside from the complexity of JavaScript, it has its advantages in compatibility and readability, such as in web development (Johansson, 2023) as it is readable in the web browser in comparison to the backend server dependence of Python. As of 2023, both languages are at the height of popularity (Oses, 2022), and with each having valid arguments to being better or worse, I conclude that they

each has a respected context to which one will excel over the other. The features and characteristics of JavaScript are better suited for frontend web development, handling structural design of user-end features, whereas Python is best applied to back-end development where mathematics and analytics are handled (Pavlou, 2022).

Overall, I feel that Python is much easier to understand than JavaScript in the way of writing. Python has similar technical and mathematical complexities to JavaScript such as tuples vs const, global definitions, etc. However, given python's philosophy of *simplicity, readability, and versatility* (Stambaugh, 2023), I believe it is better suited to someone with little to no programming experience to create something as simple and engaging as a text-based adventure game.

References

GeeksForGeeks. (2023, August 18). *Python - List comprehension*. GeeksforGeeks. https://www.geeksforgeeks.org/python-list-comprehension/

Johansson, A. (2023, July 12). *5 reasons JavaScript is still better than Python*. IEEE Computer Society. https://www.computer.org/publications/tech-news/build-your-career/5-reasons-javascript-is-still-better-than-python

Oses, A. (2022, December 28). *What your software partner should know: The top programming languages of 2023*. Forbes. https://www.forbes.com/sites/forbestechcouncil/2022/12/28/what-your-software-partner-should-know-the-top-programming-languages-of-2023/?sh=4ede1b65182b

Pankaj. (2022, August 3). *Python data types (With complete list)*. DigitalOcean. https://www.digitalocean.com/community/tutorials/python-data-types

Pavlou, C. (2022, April 25). *Front-end developer job description*. Recruiting Resources: How to Recruit and Hire Better. https://resources.workable.com/front-end-developer-job-description

Programiz. (2021, September 10). *Python list (With examples)*. Programiz: Learn to Code for Free. https://www.programiz.com/python-programming/list

RetroSpirit. (2023). *The elder scrolls: Chapter II - Daggerfall*. The Retro Spirit. https://retro.gg/game/the-elder-scrolls-chapter-ii-daggerfall/505

Simplilearn. (2022, August 1). *What is a dictionary in Python?* https://www.simplilearn.com/dictionary-in-python-article

Stambaugh, N. (2023, May 5). *The philosophy of Python*. Medium. https://medium.com/@nick-stambaugh/the-philosophy-of-python-a235bc97592c

Sudhan, S. (2022, November 30). *Python vs JavaScript - Which one is better?* Hire the World's Most Deeply Vetted Remote Developers | Turing. https://www.turing.com/kb/python-vs-javascript-complete-introduction

W3Schools. (2023). *Python tuples*. https://www.w3schools.com/python/python_tuples.asp