201- GGLT (Roll for Your Life.)
Tysen Wills, Guy Saltsman, Gerardo Sandoval, and Luis Jovel Lainez

Explanation of files:
Final_pjt.py– Our final code that we used for the presentation.

Prj_deliverable.py– The file we used for our deliverable assignment.

Intro. py– We used the file for the class activity to ensure the repository worked.

Algorithms/dice_roll.py– Both the same algorithm, Luis's dice roll function that we changed for the final submission, can be ignored.

Player_store.py– Luis's store algorithm, he made for the game, tweaked a bit for final submission, but should generally be the same, can be ignored as well.

How to run:

```
python3 file_name.py --name NAME
```

- python3 as usual
- Then the file name of the file
- --name is a flag
- NAME is the value associated with that flag

How to use:

- After we start out with the first terminal command, we are then prompted by our first roll. We are presented with what the outcome of our roll was, and how many current coins we received from said roll.
- We then have the option of buying items from the store, but are only presented with the items that are less than or equal to the amount of coins we currently have.
- We select an item, for example Small potion for 1 coin to heal us.
- Our objective is then to beat the Computer Player who is also rolling a die and trying to deal damage to us. So we must strategise when to attack, heal, or save our coins for a later turn.

Annotated Bibliography

Argparse - parser for command-line options, arguments and subcommands. Python documentation. (2025, May 13). https://docs.python.org/3/library/argparse.html

This python documentation added a useful aid on how to use argparse better, a concept that I struggled with a lot during the course. I used this resource to step up and improve my abilities with parsing arguments so I could help the team.

Kettler, R. (2012). *A guide to python's magic methods*. A Guide to Python's Magic Methods " rafekettler.com. https://rszalski.github.io/magicmethods/

This guide offers clear explanations and practical examples of Python's magic methods (**str**,len,repr), helping deepen understanding of how they enable operator overloading and object customization. It was essential for effectively implementing and leveraging these methods in our code.

Attribution:

Method/function	**Primary author**	**Techniques demonstrated**
Def parse_args	**Tysen Wills**	**The ArgumentParser class from the argparse module**
Def __init__ (class function)	**Tysen Wills**	**Super()**
Def play_turn	**Gerardo Sandoval**	**F-String containing expressions**
Def get_opponent	**Gerardo Sandoval**	**Conditional Expression**
def __init__(self, name, hp, max_hp, attack_power=0, heal_power=0):	**Guy Saltsman**	**Optional parameters and/or keyword arguments**
def __repr__(self):	**Guy Saltsman**	**Magic methods**
def show_store	**Luis Jovel Lainez**	**Usage of Dictionaries**
def get_store_item	**Luis Jovel Lainez**	**Iteration**