DNDutils

A Python CLI Application that provides utilities for playing the Tabletop RPG 'Dungeons and Dragons'

DNDutils is a simple python program that operates in a terminal and provides various utilities for playing Dungeons and Dragons.

It contains two main features:

DiceRoll - A dice rolling tool that allows users to roll a dice based on the standard xdy + m format (e.g. 1d6 + 2)

Fluff - A content generation tool that generates NPCs (non-player characters) or places based on a series of tags. The program will then allow the user to store the generated character or place in a Google Spreadsheet, which the program can later call data from to view the generated characters and places.

User Stories

DnD Players:

- As a DnD Player, I want it to be immediately obvious what the program is able to do.
- As a DnD Player, I want navigation through the program's functions to be simple and easy to control, so I can use it while in the middle of a game of DnD without any issues.
- As a DnD Player, I want the information provided by the program to be in a clear and readable format.
- As a DnD Player, I want to be able to see how the program has reached its conclusions as a result of its functions. (e.g. I want the diceroll function to give me the results of individual dice, as well as the sum total across all rolled dice) to ensure that it is working correctly.
- As a DnD Player, I want to be able to understand what inputs the program is requesting of me (for example, when rolling dice or selecting tags for Fluff) for ease of use.

DnD Dungeon Masters

- As a DnD Dungeon Master, I want to be able to create a variety of characters and places in a short timeframe, and store them for later use within my games.
- As a DnD Dungeon Master, I would like to be able to recall the stored characters and places so that I can see them after the generation process, in case I am generating a large number of instances at once.

What is Dungeons and Dragons?

Dungeons & Dragons is a Tabletop-based role-playing game made for two or more players. It is characterised by having two main categories of defining role while playing the game:

- The Player Characters, PCs, also known as Adventurers - the majority of players within a session of DnD will play as an adventurer within a small group of adventurers - heroes who seek out danger, whether for the sake of fame, fortune, or other more abstract motivations. These adventurers decide how to react to the setting, world, and story, which is usually dictated by the Dungeon Master.

- The Dungeon Master - one of the players will take the role of the Dungeon Master, or DM. This player decides the setting and narrative of the game, as well as being responsible for the actions and motives of non-player characters (NPCs) such as townsfolk, and monsters that the players go up against.

Dungeons and Dragons is a fun celebration of collaborative storytelling, but the mechanics of the game can sometimes leave a lot to be desired. A large amount of DnD is centered around rolling dice - to determine attack damage, the success chance of an event or opportunity transpiring, to even deciding what items appear within chests or on monsters within the world.

DNDUtils is a tool to help both types of player, the DM and the Player Characters, enjoy the game's storytelling components while easing two of the main burdens that both types of players face - rolling dice, and coming up with new content.

DiceRoll

DiceRoll is a dice-rolling utility that allows players to roll dice according to a predefined format used within Dungeons and Dragons. Players may need to roll multiple dice within a game to determine an ability's success, calculate the damage of spells and attacks, and pass skill checks. These typically take the form of a (x)d(y) notation - a number, followed by the letter d (indicating "dice") and the amount of sides on those dice. For example, a 1d6 would be 1 six-sided dice, or 3d12 would be 3 twelve-sided dice.

```
Welcome to DNDutils! Please select a function below.

DiceRoll [D] - A tool that allows you to roll several dice to determine the outcome of an event.
Fluff [F] - Create a randomly generated NPC or place using prepopulated lists of characteristics.

<input letter (D or F) for desired function here>
```

DiceRoll

```
Please provide the dice you want to roll in the following format, separated by commas:
Number of dice, number of sides on each side (e.g. 6, 8, 12), any modifiers (e.g. for +2: 2, for -2: -2)

<2,6,5>

Your dice total is: 17
### Dice Summary ###
Individual Rolls: [5, 7] = 12
Plus modifier: 5

Would you like to roll more dice? (Y/N)

<Y/N>
```

The user selects the dice they want to roll based on three factors:

- The number of dice to roll;
- The amount of sides on those dice (e.g. 6 for a 6-sided dice, 12 for a 12-sided dice);
- Any modifiers that are applicable (in DnD, your level of ability with a skill or attribute can influence your success chance - for example, someone with 14 or 15 points of Strength gets a +2 modifier to Strength-based rolls).

The user is then told the total of the dice they have rolled, as well as the outcomes of individual dice within the roll. This allows the user to see the breakdown to verify that the program has run correctly.

Fluff

Fluff is a content-generation tool used to assist Dungeon Masters - players who dictate the flow of the game by controlling the story and non-player characters (NPCS) such as monsters and townsfolk. It allows for the generation, saving, and storage of NPCs and places in JSON format.

Generating a Place

```
Would you like to generate an NPC or a Place?

<Place>

Please enter the type of place you would like to generate.
The list of available types are as follows:
[Town, Dungeon, PoI]

<Town>

Generating Town...

Your town is called "Slatehall".
It was founded by a nomadic tribe of humans 250 years ago.
It is currently led by a council of elders.
Notable landmarks include: The Lion's Head Tavern, a Blacksmith, a shrine to Bahamut
It is near to a goblin fortress.
Notable rumors include: An unsolved murder, cattle illness, strange lights

Would you like to copy this description to the clipboard? (Y/N)

<Y>

Description copied to clipboard!

Would you like to convert this place to an object? (Y/N)

<Y>

slatehall object created in class Town.

<print(slatehall.leadership)>

council of elders

<slatehall.add_rumor("poisoned well")>

<print(slatehall.getrumors())>

Rumors for Slatehall:

["an unsolved murder", "cattle illness", "strange lights", "poisoned well"]
```

The user first selects whether they are looking to generate an NPC or a place.

They are then provided with a list of characteristics they are looking for the NPC/Place to have:

For Places, this can be a:
- Town
- Dungeon
- PoI (Point of Interest)

For NPCs, this can be:
- Their 'Lawfulness' Alignment [Lawful, Neutral (Law), Chaotic]
- Their 'Moral' Alignment [Good, Neutral (Moral) Evil]

The program then generates a place or person based on random selections from a series of lists to describe the characteristics of that entity.

For example, with a town, it will display:
- The town's name
- When it was founded
- Who leads the town
- Interesting rumors around the town

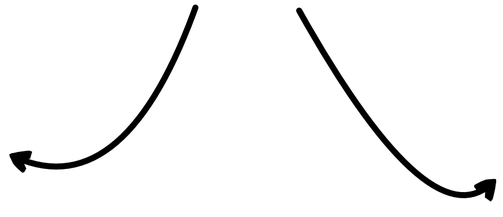
For an NPC, it will display:
- The NPCs name
- Their race
- Their age
- Any interesting motivations that help the dungeon master play the character

Generating an NPC (Non-player Character)

```
Would you like to generate an NPC or a Place?

<NPC>

Please enter the tags you would like your NPC to have from the list below:
[Lawful, LNeutral, Chaotic, Good, MNeutral, Evil, CivOccupation, AdvOccupation]

<LNeutral, Good, CivOccupation>

Generating NPC...

Your NPC is named "Bingles McGraw"
He is a male Orc. They are 32 years old.
He is lawfully neutral. He is morally good.
He was born in Slatehall. He is a Carpenter.
They have recently been distracted by a family illness.
He likes dragons, violets, and precious gems.
He dislikes mushrooms, alchemists and wheat bread.

<Y>

Description copied to clipboard!

Would you like to convert this NPC to an object? (Y/N)

<Y>

bingles_mcgraw object created in class NPC.

<print(bingles_mcgraw.alignment)>

Bingles Mcgraw's alignment is Neutral Good.

<bingles_mcgraw.set_stats(13, 8, 10, 9, 7, 14)>

Bingles Mcgraw's stats are as follows:

Strength - 13 (1)
Dexterity - 8 (-1)
Constitution - 10 (0)
Intelligence - 9 (-1)
Wisdom - 7 (-2)
Charisma - 14 (2)

<bingles_mcgraw.roll(1, 20, strength)>

Bingles McGraw rolls a 1d20, adding their strength modifier of +1.

Dice result: 17
Total: 18
```

For places, will then be asked if they want to save the town as an instance in the Google Spreadsheet. The user is able to call the instance later using the viewer function to see data about that instance.

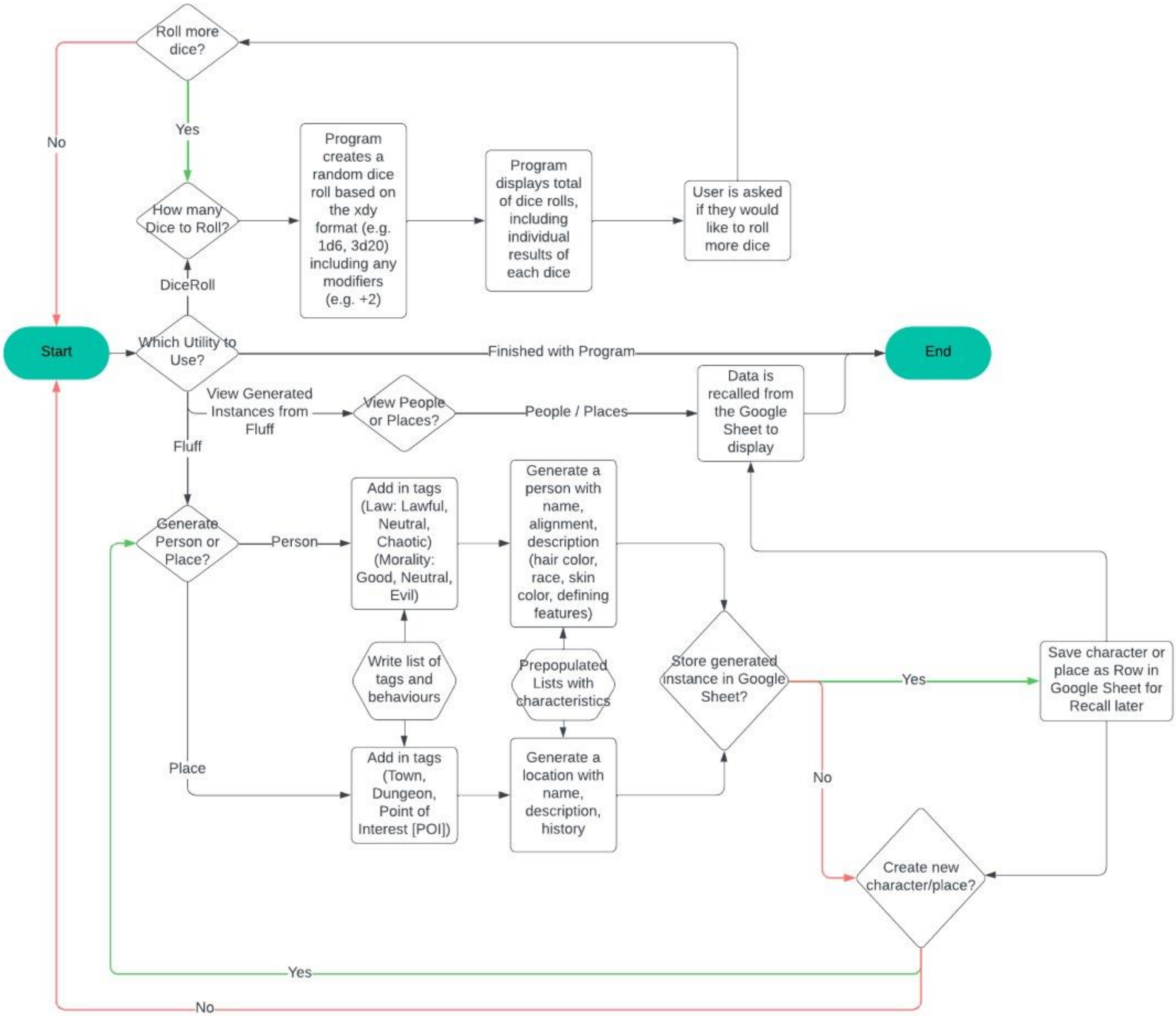
Examples of attributes a place could have include:
- When it was founded
- Its leadership structure
- Landmarks
- Rumors



For NPCs, the user will then be asked if they want to save the person as an object in to the Google Spreadsheet associated with storing people.

Examples of attributes an NPC could have include:
- Their race
- Their age
- Their alignments (Moral and Lawfulness)
- Their rumors/motivations

Logic Flowchart



Fluff - Classes

Fluff uses two primary classes for its objects - Place and NPC. These are defined below.

Place

A location within the world, such as a dungeon, town or point of interest.

Attributes:

place_type
Name
Age
Leadership
Attitude

```
def __init__(self, place_type, name, age, leadership, attitude):
    self.place_type
    self.name = name
    self.age = age
    self.leadership = leadership
    self.attitude = attitude
```

NPC

A character within the world, such as a townsperson.

Attributes:

NPC_type
Name
Age
Gender
Gender Pronouns
Race
Law Alignment
Moral Alignment
Hair Color
Skin Color
Description

```
def __init__(self, npc_type, name, age, gender, race, l_alignment, m_alignment, hair_color, skin_color, attitude)
    self.npc_type = npc_type
    self.name = name
    self.age = age
    self.gender = gender
    self.race = race
    self.alignment = l_alignment + " " + m_alignment
    self.hair_color = hair_color
    self.skin_color = skin_color
    self.attitude = attitude
```

Challenges

Large amounts of pre-generated Data

As Fluff requires large amounts of individual list items as strings, it's important to understand:

- Where am I sourcing the data from?
  - Possible answers: Lists of characteristics found online, compiled data on existing locations used for DnD

Import/Export of Objects

Importing and Exporting Data used in Fluff requires use of a Google Spreadsheet, using gspread and APIs. Learning to use the API may be difficult and may require creating the right kind of spreadsheet to be the foundation of data storage for the program.

Interface Usability

Fluff allows a great amount of functionality through the use of methods and attributes. Trying to use these within the main program may be unwieldy.

- Fluff potentially as a write-only tool - use a different part of the program for accessing the data
- Make the CLI program as "fool-proof" as possible - only allowing predefined inputs with automatic validation

Libaries

os - Writing a function that clears the terminal screen for readability

random - Random number generation for use with prepopulated lists for character and place generation

colorama - Colored terminal text using ANSI color codes

cutie - Command line User Tools for Input Easification. Handles multiple choice user selections, validates user choices to minimise exceptions.

gspread (and dependent libraries) - Interface for working with Google Sheets

google auth - Library for using Google's server-to-server auth mechanisms

APIs

Google Sheets API - For data storage within Google Sheets

Google Drive API - For storing the Google Sheet in my personal Google Drive