Media Player

Overview

Objectives

- Design a program using object-oriented programming, including class inheritance
- Gain experience working with JSON data from a web API
- Gain familiarity with Linked Lists

Knowledge Required

There are a number of topics you'll have to master to complete this assignment. Most of them are things you've seen before but may need to recall. Some of them are new this semester. And a couple are brand new, but easy to master with a pointer or two.

The following are topics we've covered in 507 this semester:

- Obtaining, validating, and processing user inputs
- Functions
- Classes and inheritance
- JSON
- Linked Lists

You have seen following to at least some extent in either 506 or 507, if not you could do some googling.

- String manipulation (e.g., extracting substrings)
- json parsing using the json module

The following haven't been covered in 506, but you'll learn them through this project:

- Named parameters with default values
- Importing self-written modules and the meaning of __name__ == "__main__"

Tasks Background

linked list

In this assignment, we delve into the fundamentals of data structures by exploring linked list. Understanding linked lists is crucial because they offer a dynamic and flexible way to store and manipulate data. Unlike arrays(lists), linked lists allow efficient insertion and deletion of elements without reallocating or reorganizing the entire data structure, making them ideal for applications where the size of the dataset changes frequently.

Moreover, learning about linked lists sets a foundation for understanding more complex data structures like binary trees, which will be covered later in the semester. Binary trees extend the principles of linked lists to a hierarchical format, where each node can have two links (or 'children'). This connection is essential as it helps develop an intuition for navigating and manipulating tree structures, and implementing various algorithms.

Related readings https://en.wikipedia.org/wiki/Linked_list

Media player

The world of digital media has transformed how we interact with media, bringing the convenience of enjoying our favorite tunes right at our fingertips. In this assignment, you will dive into the realm of digital media players, using a linked list to simulated some of its basic behavior. Your challenge is to design and implement a Player class in Python, simulating the core functionalities of a media player application. We have provided some sample json of the most recent popular tracks using a public Spotify API. Even though inappropriate contents have been removed using one of the api method, but such method provided by Spotify might not be exhaustive. Reading and listening these tracks are not required as part of the homework, and these tracks does not reflect any option from the teaching team. Feel free to change these songs, as long as the data follows the same format.

Before you get started

We have given you some files to start with:
main homework:

- media_starter.py: informaiton about step 1
- linked_list_starter.py: informaiton about step 2
- player_starter.py : informaiton about step 3
- base_data.json : json file which the player could load from
- insert_data.json : extra media which could be used for testing addMedia method of player

extra credit:

- my_queue_starter.py
- my_stack_starter.py

Tasks

Step 1 Implement a class system

In this step, you will implement a set of classes to model a part of the multimedia data model using the media_starter.py. Develop a base class (“Media”) and two subclasses (“Song” and “Movie”) that represent items in the iTunes store (there are other media types that you do not have to worry about). The details is given in the starter code about each class you’ll need to implement.
You should name your implementation media.py

Assessment

You will be assessed on but not limited to:
Whether you have correctly implemented the Media, Song, and Movie classes that have all aforementioned instance variables and methods
Whether you have used inheritance and used super( ) correctly to avoid repeating code.

Step 2 Implement the LinkedList class

The docstring is provided in the starter code. Based on this information, implement a linked list class that supports all the methods listed in the docstring.
You should name your implementation linked_list.py

Step 3 Implement the Player class

With the class representing different types of media and the linked list serving as the container for these media instances, we can finally build a media player. The player should support loading from a JSON file (iTunes API result), and 'play' the media stored in various ways. The detailed implementation is included in the starter file.
You should name your implementation player.py

Notice

Please submit all of three files listed above to Gradescope.
Your file should not have any function calls out side of if __name__ == "__main__":. Related reading
https://stackoverflow.com/questions/419163/what-does-if-name-main-do
Pay attention to the format of you print related method.
Please follow the exact naming in the docstring.

Extra Credit(10 Points)

Create two files, **my_stack.py** and **my_queue.py**. Import the linked list class and implement the Stack and Queue class using it based on the docstring. You should finish this section in less than 25 lines of code. You should also submit them to Gradescope.