Richard Hayes Crowley

07/02/2021

CSC\_242\_Lab\_08\_QA

**(1)** In the **MovieScript** class, **self** is frequently used.

Is **self** a keyword in Python? Run the following statements to check if it appears within the list of keywords.

**import keyword**

**print (keyword.kwlist)**

“**Self” is not a keyword and in the MovieScript class, you could substitute “Self” with “This” or “banana” for all Python cares. The important thing is that the first parameter in a class method list is always a pointer back to the instance of the class itself.**

**(2)** In the **MovieScript** class, as shown within the starter code, why do we have the double underscore symbols before and after  
 the **init** method?

**# the class constructor**

**def \_\_init\_\_(self) :**

**# pass**

**Certain class methods (\_\_init\_\_, \_\_str\_\_, \_\_iter\_\_, etc.) are extended from a python Object prototype which names them in this way. Specifying any method with double underscores before and after the method name makes these methods “fully private” and encapsulated within the class they are defined in. This prevents any unintended overwriting of these special methods in super classes by subclasses.**

**(3)** This project uses a list data structure, which illustrates the sequential nature of the appending process with textual scripts.

Explain how you would randomize the list after it has been populated with its elements.

**To randomize the list in a real world situation, I would reach for the random module and use random.shuffle() (and in fact I do so in my project code).**

**But in a whiteboarding interview or other theoretical situation where I don’t have access to this module, I would iterate through the list, and swap each index based on some arbitrary criteria (to generate pseudo-randmoness) until I reach the end of the list.**

**(4)** Did you utilize adictionary when completing your application code? Would a dictionary, having key - value pairs, simplify the text searching process? Explain your answer.

**Yes, I used a dictionary for crafting a call sheet whose structure is:**

**[Scene Number]:**

**{**

**Sequence: “this is a description of the plot point”**

**Actors: [“Tom Cruise”, “Jennifer Lopez”, “etc.”]**

**}**

**This structure was helpful for printing the call sheet, wherein the user requests to see the actors in each scene, and was also helpful for searching for actors in scenes, because in one call I could return both the actors in the scene and the description of the scene itself.   
  
Because the key in this dictionary is the scene number, this helped with the text searching process because I could return both the actors in the scene and the sequence itself with a one liner. I used a “reducer” function and list comprehension to build this dictionary from the sequence list and the actors list.**

**(5)** Let us pretend that we are movie script writers. Explain how your program can help organize our understanding of the individual scenes that comprise a movie.

**This program could be useful for understanding the individual scenes that comprise a movie by allowing script writers to search for keywords and find their scenes’ sequence number, and to perhaps help them make decisions about scene placement (we don’t want too many scenes where the keyword “romance” are found in sequential order, since that would be too gushy for most audiences).**