CS 1110 Spring 2022, Assignment 5: A4 ReMixed

Updates:

- Wed Apr 6, 12pm Ithaca time:
  - Footnote 4: clarification that it is A4 *constructor* calls that are re-used, not A4 code in general.
  - In a5.py, the spec for _prompt() is missing double-quotes around the `q`. See test cases for what is intended.
  - In file a5_test.py, the following typos have been corrected: (a) Line 130: two instances of `next+pos` should be changed to `next_pos`(b) Lines 175 and 183: textttresults should be `result`.

CS 1110 Spring 2022, Assignment 5: A4 ReMixed*

The files you'll need are in this zip file:[1]
http://www.cs.cornell.edu/courses/cs1110/2022sp/assignments/assignment5/a5_skeleton.zip . a5_test.py contains test cases for the functions you'll complete in file a5.py, except for method Loop.play(). When done, you'll be able to have interactions like that below: your web browser should play through videos of the Songs in whatever Mixes you like!

```
[ljl2@dragonday A5] python a5_music.py
Select from the following choices of Mixes
   0:  LCD Soundsystem (2 songs)
   1:  why the big suit (5 songs)
   2:  Talking Heads Songs and Covers (7 songs)
   3:  Recs from M (9 songs)
   4:  Songs in the key of Falsetto (4 songs)
   5:  Eccentrica (13 songs)
   6:  Prof. Bracy's list (6 songs)
   7:  Chris's list (1 song)
   8:  Lifan's list (2 songs)
   9:  Gary's list (8 songs)
  10:  Prof. Lee's list (5 songs)
  11:  Sad songs (7 songs)
  12:  instrumentals (3 songs)
  13:  electronic music (3 songs)
  14:  hip-hop/rap (2 songs)
  15:  CS1110 picks (22 songs)
  16:  Spring Break! (2 songs)
Which number do you want? 16
Enter Y for given Song order, N for shuffled order: Y
Starting Loop Spring Break!
Hit Enter/Return to start the next video, Here Comes the Sun by The Beatles, or "q" to quit:
[YouTube video opens in browser!]
Hit Enter/Return to start the next video, Nana Party (Chuang 2021 live version) by OJM,...:
[YouTube video opens in browser]
Hit Enter/Return to start the next video, Here Comes the Sun by The Beatles, or "q" to quit:
[YouTube video opens in browser. Notice we're looping.]
Hit Enter/Return to start the next video, Nana Party (Chuang 2021 live version) by OJM,...: q
Exiting this Loop

[ljl2@dragonday A5] python a5_music.py
Select from the following choices of Mixes
  [omitted]
Which number do you want? 4
Enter Y for given Song order, N for shuffled order: N
Starting Loop Songs in the key of Falsetto
Hit Enter/Return to start the next video, Bennie and the Jets by Elton John, or "q" to quit:
[YouTube video opens in browser. Note the Song order is different from the original Mix.]
Hit Enter/Return to start the next video, Redbone by Childish Gambino, or "q" to quit:
(No video available)  # We got too tired to find videos for all Songs
Hit Enter/Return to start the next video, Bohemian Rhapsody by Queen, or "q" to quit:
[YouTube video opens in browser]
Hit Enter/Return to start the next video, Bennie and the Jets by Elton John, or "q" to quit: q
Exiting this Loop
```

*Authors: Lillian Lee.

[1] If you are using a Windows computer: when you download a zip file, it is represented in File Explorer as a folder with a little zip on it. That is not a real folder, which is why you have may have had trouble saving changes to files in it! So, first create a new folder, such as "a5-real". Double-click on that zip pseudo-folder. Then, move the contents of the pseudo-folder to folder a5-real.

Contents

1 Rules

See Sections 1.1-1.3 of Assignment 1[2], Sections 3.1-3.2 of Assignment 2[3], and Sections 3.1.3 and 3.2 (pre-submission checklist) of Assignment 3[4]. If you have questions about whether some Python is "legal", ask on Ed Discussions.

2 Timeline

1. If you are partnering,[5] **form your group on CMS *before* submitting anything**. This links your submission "portals".
2. By 2pm Ithaca time on Sun Apr 17, submit your *partial* progress on CMS.[6] It is OK if you haven't finished working yet.[7]
3. By **11:59pm (Ithaca time) on Sun Apr 17**, make your final submission, and do steps 1-3 in the "Updating, verifying, and documenting assignment submission" section of https://www.cs.cornell.edu/courses/cs1110/2022sp/resources/cms.html

3 Add methods to class Mix

3.1 Task: Provide an __add__() method.

We had been creating new Mixes out of previously existing ones by constructor calls. Augment class Mix with an __add__() method. This will enable much more convenient expressions for combining Mixes, such as `new_mix = spring + elec`.

Note: for the first time, we are not providing you with a header for the requested function (method). When writing the method, follow the given directions:

```
# STUDENTS: you must modify the Mix class somehow so that if m1 and m2 are
# two mixes, then the expression
#   m1 + m2
# evaluates to a new Mix whose `contents` is the two-item list consisting
# of m1 and m2 and `title` is the concatenation of the two Mixes' titles
# separated by ' + '.
```

[2] https://www.cs.cornell.edu/courses/cs1110/2022sp/assignments/assignment1/a1.pdf
[3] https://www.cs.cornell.edu/courses/cs1110/2022sp/assignments/assignment2/a2.pdf
[4] https://www.cs.cornell.edu/courses/cs1110/2022sp/assignments/assignment3/a3.pdf
[5] Reminder: Both parties need to act on CMS in order for the grouping to take effect. See the "How to form a group" instructions at https://www.cs.cornell.edu/courses/cs1110/2022sp/resources/cms.html .
[6] And, as usual, perform steps 1-3 in the "Updating, verifying, and documenting assignment submission" section of https://www.cs.cornell.edu/courses/cs1110/2022sp/resources/cms.html
[7] The 2pm checkpoints provide you a chance to alert us if any problems arise, and for partners to become aware if something has gone wrong with the CMS grouping process (via us emailing people who haven't yet submitted — if your partner submitted but you got the warning email, that means CMS doesn't think you've grouped!). Since we'll send a reminder around 2pm to all those who haven't submitted yet, do not expect that we will accept work that doesn't make it onto CMS on time, for whatever reason. There are no so-called "slipdays" and there is no "you get to submit late at the price of a late penalty" policy. Of course, if some special circumstances arise, contact the instructor(s) immediately.

3.2 Task: Provide method songs() as a replacement for non-method function a4.songs_in()

.

We're asking you to include an additional parameter `keep_dups`. The idea is some people may construct Mixes with repeated Songs on purpose, maybe to achieve some sort of emotional effect; they would want to set `keep_dups` to `True`.

```
def songs(self, keep_dups=False):
    """Returns a list of Songs in this Mix.

    The list should be ordered thusly:
       * the first set of elements is all the Songs represented by the
         first item (a Song or itself a Mix) in this Mix's contents,
       * the second set is all the Songs represented by the second item
         in this Mix's contents.
       * etc.

    If `keep_dups` is True, then repeated Songs are included in the returned
    list; otherwise, a repeated Song occurs only once, in its leftmost
    possible position.

    Example using the variables in file a5_music.py:
       if m is Mix('duplicates2', [llee, weird]), then

       m.songs() ->
       [s_ba, s_ds, s_np, s_dyc, s_lme, s_ttabeftbou, s_r, s_batj, s_br,
        s_tmbtp, s_pk, s_gib, s_oial, s_bdth, s_tmbtp2, s_bdth2]
       m.songs(keep_dups=True) ->
       [s_ba, s_ds, s_np, s_dyc, s_lme, s_ttabeftbou, s_r, s_batj, s_br,
        s_tmbtp, s_pk, s_gib, s_oial, s_bdth, s_tmbtp2, s_bdth2, s_dyc, s_lme]
    Preconditions: keep_dups is a Boolean
    """
```

~~In file a5_thoughtquestions.py, we ask you to explain the following puzzle: how is it that~~ Notice that we've added a parameter to __init__(), but by giving that parameter a default value, Song constructor calls from a4_test.py still work with just two explicit arguments.[8]

4 Playing a Loop of Songs

Now for our main job: making the interaction shown on the first page a reality! For this endeavor, a5.py supplies a skeleton for new class **Loop**:

```
class Loop:
    """
    A Loop is an individual's execution of a Mix, an infinite loop until the
    user requests the Loop to end.

    The idea is that two people A and B might have be simultaneously listening
    to Songs from the same Mix m; but A might be on the 5th Song, and B might
    be on the 17th. So, we have different Loop objects to represent these two
    different viewing states.

    Instance attributes:
        title [str]: title of the Mix this Loop was derived from.
        slist: a non-empty list of Songs.
        next_pos [int]: position/index in `slist` of the *next* Song to play.
           It is always a valid index for `slist`.

    """
```

Notice that a Loop is generated from a Mix. But a Loop is simpler than its source Mix. We wanted the capability to have Mixes be built from sub-Mixes; but there's no reason for Loops to have recursive structure.

[8] Observe that it is very convenient for the old A4 constructor calls to still work: we were able to just copy the Songs and Mixes creation code from the previous assignment as is! We hope this observation helps you appreciate this optional-argument mechanism.

4.1 Task: Provide the Loop __init__() method

Here's the relevant snippet of a5.py:

```
def __init__():
    """
    A new Loop with its `slist` set to m.songs(), its `title` set to the
    title of `m`, and its `next_pos` set to 0.

    EXCEPTION: If `shuffle` is True, then random.shuffle() should be applied
    to the contents of `slist` to put it in random order.

    Preconditions:
    `m` is a Mix (not None).
    `shuffle` is a boolean.
    """
    # STUDENTS: add parameters to the __init__ header and implement
    # according to the specification. The parameter names are given by the
    # docstring. The parameter `shuffle` should have default value False.
```

4.2 Task: Implement helper method _prompt()

You can see on the first page of this handout that we often want printouts like `Hit Enter/Return to start the next video, Here Comes the Sun by The Beatles, or "q" to quit` or `Hit Enter/Return to start the next video, Redbone by Childish Gambino, or "q" to quit:` Creating such strings is the job of this method.

```
def _prompt(self):
    """Returns string
        'Hit Enter/Return to start the next video, ' + \
        '<next song's title> by <next song's artist>, or "q" to quit: '
       where the stuff in angle brackets should be replaced by the reasonable
       thing.
    """
```

4.3 Task: Finish up Loop method play()

We've written most of the code for you, since we haven't yet covered while-loops in class. Basically, the idea of the Loop play() method is to continuously query the user if they want to play the next Song, or quit the Loop. All you need do is write the part of the loop body that plays the next Song and updates attribute `next_pos`. Be careful in that last step to account for the Loop looping back to the first Song after the last Song is played.

```
def play(self):
    """
    1. Print the title of this Loop.

    2. continuously query the user if they want to play the next Song.
    If the last Song has been reached, the next song is the *first* Song
    in this Loop's slist.

    Response of `q` means quit, any other response means play the next
    Song."""

    print('Starting Loop '+self.title)
    response=input(self._prompt())
    while 'q' not in response:
        # STUDENTS: implement the body of this while-loop so that it:
        # a. Plays the next Song
        # b. Changes the value of attribute `next_pos` appropriately.

        # STUDENTS: leave the next line as the last in the while-loop body.
        response=input(self._prompt())
    print('Exiting this Loop')
```

4.4 Grand finale and testing the Loop play() method

Try running a5_music.py a couple of times until you are satisifed you can handle all combinations of potential user inputs.