```
'''
Lab 3

Trang Van
CIS 41B

Front End - GUI,

'''
import tkinter as tk
import tkinter.messagebox  as  tkmb

import sqlite3
import numpy as np

DATABASE = 'movieinfo.db'           #const for db name

'''
Master window running the display of movies and it's information. The
class also gets data from the SQL database to populate
lists that can be used in class functions
'''
class mainWin(tk.Tk):
    '''
    Initializes multiple lists to use in functions
    '''
    def __init__ (self):
        super().__init__()
        self._genresInfo = []                                   # Used to
load genre data from DATABASE
        self._ratingsInfo = []                                  # Used to
load rating data from DATABASE
        self._rate_increment = np.arange(5.5, -0.5, -0.5)       #
Generated to create radio buttons in ratingWin
        self._moviesInGenre = []                                # Stores
movies found based on given genre
        self._moviesInRatings= []                               # Stores
movies found based on given rating

        self.title("Movie Guide")

        # Get data from SQL database
        self._conn = sqlite3.connect(DATABASE)
        self._cur = self._conn.cursor()
        self._cur.execute("SELECT * from Genres")
        for tup in self._cur.fetchall():
            self._genresInfo.append(tup[1])
        self._cur.execute("SELECT MovieDB.rating FROM MovieDB")
        for tup in self._cur.fetchall():
            self._ratingsInfo.append(str(tup[0]))

        # Buttons for genre and rating
        F1 = tk.Frame(self)
```

```
        B1 = tk.Button(F1, text = "Genre", command =
self._getGenreWin).grid(row = 1, column = 0, padx=10)
        B2 = tk.Button(F1, text = "Rating", command =
self._getRatingWin).grid(row = 1, column = 1)
        F1.grid(row = 0, column= 0)

        # Listbox and Scrollbar
        S = tk.Scrollbar(self)
        self._LB = tk.Listbox(self, width = 75, yscrollcommand=S.set)
        S.config(command = self._LB.yview)
        self._LB.grid(row = 1, column = 0)
        S.grid(row = 1, column = 1,columnspan= 2, sticky = 'ns')


    '''
    Creates a dialogBox window to get user choice and display movies to
listbox. Uses _displayInfo to show movie details.
    '''
    def _getGenreWin(self):
        self._LB.delete(0,tk.END)

        dialog = dialogBox(self, self._genresInfo, "genre")
        self.wait_window(dialog)
        choice = dialog.getUserChoice()
        genre = self._genresInfo[choice]            # stores user's choice

        if choice != -1:
            # Find movie names based on genre from database
            self._cur.execute('''SELECT MovieDB.movie FROM MovieDB JOIN
Genres
                              ON MovieDB.genre_id = Genres.id  AND
Genres.genre = ? ''', (genre,))
            for record in self._cur.fetchall() :
                self._moviesInGenre.append((record[0]))

            tk.Label(self, text = genre).grid(row = 10,column =
0,columnspan = 5,sticky = 's',padx= 10)
            self._LB.insert(tk.END, *self._moviesInGenre)
            self._LB.bind('<ButtonRelease-1>', self._displayInfo)

    '''
    Creates a dialogBox window to get user choice and display movies to
listbox. Uses _displayInfo to show movie details.
    '''
    def _getRatingWin(self):
        self._LB.delete(0,tk.END)

        dialog = dialogBox(self, self._rate_increment, "rating")
        self.wait_window(dialog)

        choice = dialog.getUserChoice()

        print(choice)
```

```
        rating = self._rate_increment[choice]       # stores user's choice
        print(rating)
        if choice != 1:
            # Find movie names based on ratings from database
            self._cur.execute("SELECT MovieDB.movie FROM MovieDB WHERE
MovieDB.rating = ? ORDER BY MovieDB.rating", (rating,))
            for record in self._cur.fetchall() :
                self._moviesInRatings.append((record[0]))


            if len(self._moviesInRatings) == 0: tk.Label(self, text ="No
movies").grid(row = 10,column = 0,columnspan = 5,sticky = 's')
            else: tk.Label(self, text = str(rating)+" star(s)").grid(row =
10,column = 0,columnspan = 5,sticky = 's')

            self._LB.insert(tk.END, *self._moviesInRatings)
            self._LB.bind('<ButtonRelease-1>', self._displayInfo)
    '''
    Displays movie details: rating, release date, genre by getting the
information from database

    '''
    def _displayInfo(self, event):

        description = tk.StringVar()

        # Get information from database and prints (accounts for
star/stars)
        self._cur.execute('''SELECT MovieDB.rating, MovieDB.date,
MovieDB.genre_id FROM MovieDB JOIN Genres
                      ON MovieDB.genre_id = Genres.id AND MovieDB.movie
= ?''', (self._LB.selection_get(),))
        for record in self._cur.fetchall():
            if record[0] >1:
                description.set(str(record[0]) + " stars Released: " +
record[1]+ " Genre: " + self._genresInfo[record[2]-1])
                info = tk.Label(self, text= description.get()).grid(row =
10, column = 0, columnspan = 5, sticky = 's', padx= 10)
            else:
                description.set(str(record[0]) +" star Released:" +
record[1]+ "Genre:" + self._genresInfo[record[2]-1])
                info = tk.Label(self, text= description.get()).grid(row =
10, column = 0, columnspan = 5,sticky = 's', padx = 10)

'''
Dialog box derived from TopLevel to get user's choice of genre.

'''
class dialogBox(tk.Toplevel):
    def __init__(self, master, category, title):
        super().__init__(master)
        self.transient(master)
        self.grab_set()
        self.focus_set()
```

```
        self.protocol("WM_DELETE_WINDOW", self._close)

        self._category = category
        self.title("Choose a " + title)
        self._controlVar = tk.IntVar()
        self._controlVar.set(0)

        # Radio Buttons using a sorted genre list
        for i,cat in enumerate(self._category):
            tk.Radiobutton(self, text=cat, variable=self._controlVar,
value=i).grid(row=i, column=0, padx=8, sticky = 'w')
        # OK Button
        okBT = tk.Button(self, text = "OK", command =
self.destroy).grid(column = 0)

    '''
    Returns user's choice with the controlVar
    '''
    def getUserChoice(self):
        return self._controlVar.get()
    def _close(self) :
        self._controlVar.set(-1)
        self.destroy()
'''
main create mainWin obj and runs the application/GUI
'''
def main():
    app = mainWin()
    app.mainloop()

main()
```