Richard Hayes Crowley

06/28/2021

CSC\_242\_Lab\_06

**SOURCE CODE:**

**Library\_Class.py**

*from* random *import* randint

*from* tabulate *import* tabulate

*import* pickle

class List2D(object):

''' 2d List with the first column in each row containing the number of the row, used in Library class to indicate the floor of the library '''

def \_\_init\_\_(*self*, *rows*, *columns*) -> any:

super().\_\_init\_\_()

*self*.data = [i *for* i *in* range(*rows*)]

*# first item will be idx + 1, or, the row number*

*for* row *in* range(*rows*):

*self*.data[row] = [

None *for* i *in* range(*columns*)]

*return* *self*.data

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

*return* str(*self*.data)

class Library(List2D):

'''

Library Class Accepts rows, columns and genres as arguments and builds a multidimensional list representing a library

Two empty 2D lists are initialized with row x column dimensions, these represent the library categories matrix and the library quantities matrix

Library categories and library quantities matrices are cross referenced by using a genre dictionary, whose keys (ints) are referenced in the library categories matrix.

The position of the genre key in the library category matrix is cross-referenced with the library quantity matrix to produce a meaningful representation of how many items of each genre category are in each position (room) in the library matrix

'''

def \_\_init\_\_(*self*, *rows*, *columns*, *genres*) -> None:

*self*.libraryQuantities = super().\_\_init\_\_(*rows*, *columns*)

*self*.libraryCategories = super().\_\_init\_\_(*rows*, *columns*)

*self*.library = super().\_\_init\_\_(*rows*, *columns*)

*self*.rows = *rows*

*self*.columns = *columns*

*self*.genres = *genres*

*self*.genreKeys = [k *for* k *in* *genres*.keys()]

*# construct library categories matrix*

*for* row *in* range(len(*self*.libraryCategories)):

*# assign random integer from genre keys for each column in row*

*for* col *in* range(len(*self*.libraryCategories[row])):

*self*.libraryCategories[row][col] = randint(

*self*.genreKeys[0], *self*.genreKeys[(len(*self*.genreKeys) - 1)])

*# construct library quantities matrix*

*for* row *in* range(len(*self*.libraryQuantities)):

*# assign random integer between 1 and 50 for each column in row*

*for* col *in* range(len(*self*.libraryQuantities[row])):

*self*.libraryQuantities[row][col] = randint(

1, 50)

*# build multidimensional library by combining the two lists and cross referencing genre keys*

*for* row *in* range(len(*self*.library)):

*# assign random integer for each column in row, skipping the first column at index 1, which holds the row number...*

*for* col *in* range(len(*self*.library[row])):

*self*.library[row][col] = [*genres*[*self*.libraryCategories[row]

[col]], *self*.libraryQuantities[row][col]]

def getCategoryMatrix(*self*):

*return* *self*.libraryCategories

def getQuantitiesMatrix(*self*):

*return* *self*.libraryQuantities

def getLibrary(*self*):

*return* *self*.library

def getTable(*self*):

'''return dynamic, tabular representation of data'''

*return* tabulate(*self*.getLibrary(), *headers*=["Floor"] + [f"Room {i + 1}" *for* i *in* range(*self*.columns)], *showindex*=[(i + 1) *for* i *in* range(len(*self*.getLibrary()))])

def saveLibrary(*self*):

pickle.dump(*self*, open("library.p", "wb"))

**library.py**

*from* Library\_Class *import* Library

*from* simple\_term\_menu *import* TerminalMenu

*import* pickle

def main():

*# a dictionary of genres*

genres = {1: "hobby", 2: "romance", 3: "cooking", 4: "science",

5: "adventure", 6: "puzzles"}

lib = None

print("\n~\*~\*~\*~\* Welcome to the library simulator! ~\*~\*~\*~\n")

*while* True:

new\_library = TerminalMenu(

["Create new Library", "Load library", "Exit"], *title*="What would you like to do?").show()

*if* new\_library == 0:

lib = Library(4, 4, genres)

print("Library created.\n")

*break*

*elif* new\_library == 1:

*try*:

lib = pickle.load(open("library.p", "rb"))

print("Library loaded.\n")

*break*

*except*:

print("No library saved! Create a new library to get started.\n")

*else*:

print("Goodbye!")

exit()

print(lib.getTable() + '\n')

library = lib.getLibrary()

*while* True:

selection = TerminalMenu(

["Inspect room", "Save Library", "Exit"], *title*="What would you like to do?").show()

*if* selection == 0:

select\_floor = TerminalMenu([f"Floor {i + 1}" *for* i *in* range(

len(library))], *title*="What floor would you like to inspect?").show()

select\_room = TerminalMenu(

[f"Room {i +1}" *for* i *in* range(len(library[select\_floor]))], *title*=f"What room on floor {select\_floor +1}").show()

print(

f"\nRoom {select\_room + 1} on floor {select\_floor + 1} has {library[select\_floor][select\_room][1]} {library[select\_floor][select\_room][0]} magazines\n")

*elif* selection == 1:

*try*:

print("saving library...")

lib.saveLibrary()

print("Library saved!\n")

*except*:

print("Uh-oh! Was unable to save library! Bad developer!\n")

*else*:

print("goodbye!")

exit()

*if* \_\_name\_\_ == "\_\_main\_\_":

main()

**OUTPUT:  
  
Please see attached video for lab demo!**