CSC 122 Python Applications

Lesson 4: Project 1-C – Wake Golf Tour

Objectives

In this assignment, students will:

- Use basic Python object-orientated programming constructs, including creating classes, creating objects and calling their methods.
- Code Python functions that read list data and place the data into class objects.
- Use the Python map function to convert a list of strings into a list of integers.

Python Programming Instructions

Please use PyCharm or a text editor (Notepad++ is good for Python) to type your program code files. You can use PyCharm or the command line to test your program. See documents in Course Resources for more details about these environments.

a. If you are using **PyCharm**, then name the project exactly **WakeGolfTourC**. The Python source code will be in the '**WakeGolfTourC'** folder. Place the Python program file, **golf_tour.py**, the seven class definitions files (**golfCourse.py**, **hole.py**, **golfer.py**, **tournament.py**, **tournGolfer.py**, **round.py**, and **golferRoundScores.py**) and the **input files** in the **WakeGolfTourC** project folder. The output files should be in the folder after running the program.

b. If you are using the **command line interface**, create a **WakeGolfTourC** folder. Place the Python program file, **golf_tour.py**, the seven class definitions files (**golfCourse.py**, **hole.py**, **golfer.py**, **tournament.py**, **tournGolfer.py**, **round.py** and **golferRoundScores.py**) and the **input files** in the **WakeGolfTourC** project folder. Run the program from that folder. The output files should be in the folder after running the program.

Once you have completed the assignment, zip up the **WakeGolfTourC** folder using directions from the document, **Creating and Submitting Programs**. Submit the **WakeGolfTourC.zip** file to Blackboard for credit.

Programs that are submitted incorrectly will not be graded.

Program Specifications

Project Description

Please re-read the **Wake Golf Tour App** and **Wake Golf Tour Specification** documents. Project 1 has four-parts that follow each other. The code for Project 1-B starts with the code from Project 1-A, …, and the code for Project 1-D starts with the code from Project 1-C, and so on.

You are required to follow the specifications given. Algorithms will be provided for each of the functions. In addition, the code for some of the functions will be provided. **You must follow the algorithm and code for the functions for which the code is given to gain an understanding of how the related functions are coded.** Then code up the functions for which you must provide your own code, using the given algorithms.

Project Part C:

This third part of the project, **Project 1-C**, builds on the Python program file called **golf_tour.py** from **Project 1-B**. It also has seven class definition files, **golfCourse.py**, **hole.py**, **golfer.py**, **tournament.py**, **tournGolfer.py**, **round.py,** and **golferRoundScores.py**. The main program contains 11 functions, including the **main** function and the 2 helper functions. Details about these functions are given to you in comments within the functions themselves and are summarized in the **Wake Golf Tour Specification** document.

In Project 1C, you are to create the **GolferRoundScores class definition** and **object list**. You are to read the **GolferRoundScores** data from the **input CSV file**, which is used to create the **GolferRoundScores** object list. Along with the data from the input file, the rest of the input parameters are object lists which were created from previously called **create functions**. Like the other create functions, this one will also produce a list of class objects containing the data for the corresponding database table, including and be saved in a file using the class string (**__str__**) method.

Create Function

create_golfer_scores (input_filename, golfer_list, tournament_list, rounds_list, tourn_golfers_list): round_scores_list

GolferRoundScores class definition: **golferRoundScores.py**

Provided Code

The **golf_tour.py** program has the following function and class definition code provided for you. **Do not change the code in these functions:**

main
create_golf_courses
get_tourn_golfer_id
write_objs_to_file
GolfCourse in **golfCourse.py**
GolferRoundScores in **golferRoundScores.py**

Required Code

You must copy your existing code from Project 1-A for:

- **create_golfers** – From Project 1-A
- **create_holes** – From Project 1-A
- **Hole** in **hole.py** – From Project 1-A
- **Golfer** in **golfer.py** – From Project 1-A

You must copy your existing code from Project 1-B for:

- **create_tourn_golfers** – From Project 1-B
- **create_rounds** – From Project 1-B
- **create_tournaments** – From Project 1-B
- **Tournament** in **tournament.py** – From Project 1-B
- **TournGolfer** in **tournGolfer.py** – From Project 1-B
- **Round** in **round.py** – From Project 1-B

Helpful Hint: After you copy your existing code from Project 1-A and Project 1-B, run **golf_tour.py** to be sure it runs without errors (exit code 0) and generates the correct output. This will assure that you have working code before you start writing the new code required for Project 1-C.

You are responsible for coding up the following functions. See the Appendix for example algorithms for these functions.

- **create_golfer_scores**
- **get_round_id**

Program Starter Code

There is a zipped folder in Blackboard called **'Project1CStarterCode.zip'** that contains the **input files**, the starter **golf_tour.py** program and the class definition files, **golfCourse.py**, **hole.py**, **golfer.py**, **tournament.py**, **tournGolfer.py**, **round.py**, and **golferRoundScores.py**, for **Project 1-C**.

Each of the provided files should be placed in the same directory as your program, **golf_tour.py**.

Program Input Files

golf_courses_infile = "golfCoursesInput.csv"
golfers_infile = "golfersInput.csv"
tournaments_infile = "tournamentsInput.csv"
golfer_scores_infile = "roundScoresInput.csv"

Program Output Data

The input data are read into the **golf_tour.py** program in the **create_golfer_scores** function and the data returned is used to create a list of objects containing that data.

The objects in the list are written to the screen, and to the following file:

Program Output File

golfer_scores_file = "golferRoundScores.csv"

Project Part C: create_golfer_scores function

The data in the input csv file (**roundScoresInput.csv**) contain, the golfer name, tournament name, and the tournament round day, in addition to the scores. However, the **GolferRoundScores** object (and database table) needs **ids** instead of names, so for the tournament and golfer names, use dictionaries as lookup tables (maps). You must build these from the tournament objects list and the golfer objects list.

For example, for the golfer lookup map, created in Project1A:

```
# Create a lookup map for golfer_name to golfer_id
golfer_name_to_id = dict()
for golfer in golfer_list:
    golfer_name_to_id[golfer.get_golfer_name()] =
                                golfer.get_golfer_id()
```

Then to get the id:

```
# lookup the golfer_id for the golfer name
golfer_id = golfer_name_to_id[golfer_name]
```

For the **round_id** and **tourn_golfer_id**, create helper functions -

get_tourn_golfer_id (**tourn_golfers_list**, **tourn_id**, **golfer_id**)

Helper function to get the **tourn_golfer_id** based on the specified **tourn_id** and **golfer_id**, using the **tourn_golfers_list**. As a gift, this has been provided in the starter code.

get_round_id (**rounds_list tourn_id**, **tourn_day**):

Helper function to get the **round_id** based on the specified **tourn_id** and **tourn_day** using the **rounds_list.**

In the **create_golfer_scores** function consider using Python's 'map' function to convert the list of strings (the scores) to a list of integers.

Here is the definition of the map function:

The map() function applies a given function to each item of an iterable (list) and returns a map object containing the results. The returned value from map() (the map object) then can be passed to other functions, like list() (to create a list).

The syntax of map() is:

map *(function, iterable, ...)*
where

- ***function*** *– this is the function that acts on each item of the list (iterable).*
- ***iterable*** *– list (iterable) which is to be mapped*

Check out other references for additional information on the **map()** function, if needed. The following site provides an interactive example:

Python map() Function (Ctrl-Click to Open in a new Window)

PlainText: https://www.w3schools.com/python/ref_func_map.asp

In our case the function is **int()** and our iterable is the list of strings representing the golfer scores. So each string in the list is passed to the int() function resulting in a map object with a list of integers.

Use this code in your **create_golfer_scores:**

```
scores_list = list(map(int, scores[3:]))
```

Program Execution

Please re-read the **Wake Golf Tour App** and **Wake Golf Tour Specification** documents, until you have grasped the purpose of Project 1.

To Begin:

1. Please open the **Project1CStarterCode.zip** file, now!

2. Replace the **golfCourse.py**, **hole.py**, and **golfer.py** class definition files with the ones from Project 1-A

3. Replace the **tournament.py**, **tournGolfer.py**, and **round.py** class definition files with the ones from Project 1-B

4. Copy the code for the **create_golfers** and **create_holes** from Project 1-A.

5. Copy the code for the **create_tournaments, create_rounds,** and **create_tourn_golfers** from Project 1-B.

6. To better understand this project, organize document notes on paper, note cards, or on a whiteboard.

7. Come to Open Lab to get help.

8. Email me questions and code to review, if you need help.

9. You may team up with others. Use the Students Helping Students Discussion Board in Blackboard to find partners.

You are not on your own. Your teacher, the lead instructor and other students can help you.

Program Output to Screen:

Wake Golf Tour Project 1

The GolfCourse object list:

< See Project 1-A output >

The Hole object list:

< See Project 1-A output >

The Golfer object list:

< See Project 1-A output >

The Tournament object list:

< See Project 1-B output >

The Round object list

< See Project 1-B output >

The TournGolfer object list

< See Project 1-B output >

The GolferRoundScores object list

1,1,1,74,4,3,5,4,4,5,5,3,5,4,3,4,5,4,4,4,4,4
2,2,1,76,6,3,4,6,4,5,6,4,4,3,4,4,3,4,5,4,3,4
3,3,1,76,4,3,4,4,4,5,4,4,5,4,3,4,5,4,6,4,5,4
4,4,1,74,4,3,4,4,5,5,4,3,4,4,3,4,6,4,5,4,4,4
5,5,1,67,3,3,3,4,4,4,3,4,3,5,3,4,4,4,5,3,4,4
...
...
672,222,45,76,6,4,4,5,3,4,5,6,4,4,5,4,3,4,2,4,5,4
673,223,45,74,4,4,4,5,5,4,3,4,3,5,5,4,4,4,4,4,4,4
674,224,45,78,5,4,6,5,4,4,5,5,4,4,4,4,4,4,4,4,4,4
675,225,45,72,4,4,5,5,4,3,3,3,4,5,4,5,4,4,3,4,4,4

Process finished with exit code 0

Appendix

Example algorithms

create_golfer_scores

```
"""
1. Create lookup dictionaries ...
   a. Create a lookup dictionary (golfer_name_to_id)
      for associating a golfer_name to golfer_id
   b. Create a lookup dictionary (tourn_name_to_id)
      for associating tourn_name to tourn_id

2. Create an empty list called 'round_scores_list' that will be
   filled in with GolferRoundScore objects whose data comes
   from the input file and each of the object list parameters:
   golfers_list, tourns_list, rounds_list, tourn_golfers_list

3. Initialize the golfer_scores_id

4. Use a try/except block to capture a File Not Found Error
   a. Open the input file object for reading the input file
   b. Call the csv.reader function, passing in the input file
      and capturing the CSV file contents.
   c. Create a list from the file contents: 'golfer_scores_list'
   d. Close input_file object

5. Create an outer loop to read each set of scores in
   'golfer_scores_list'
   Loop:
     a. Get the golfer_name, tourn_name, and day from the
        the first three elements, stripping whitespace.
     b. The rest of the elements (using slice scores[3:])
        are converted to a list of ints - scores_list.
        Use Python's 'map' function to convert the strings to
        ints and then use the 'list' function to convert the
        object returned from the map to a list.
     c. Get the golfer_id using the golfer_name_to_id
       (from step 1a)
     d. Get the tourn_id using the tourn_name_to_id
       (from step 1b)
     e. Call helper functions to get round_id, and the
        tourn_golfer_id
     f. Set the total_round_score by summing the scores_list
```

g. Create a new GolferRoundScores object, call it golfer_scores, passing in golfer_scores_id, tourn_golfer_id, round_id, total_round_score, and the scores list (from step 5b.)
h. Append the GolferRoundScores object to the round_scores_list
i. Increment the golfer_scores_id

6. Print the round_scores_list objects to the console
7. Return the round_scores_list
"""

get_round_id

"""
Loop looking for the Round object whose
tourn_id instance variable value matches
the specified tourn_id and whose
day instance variable value matches the specified day
When the matching Round object is found,
return the round_id from the round object
"""

get_tourn_golfer_id

"""
Loop looking for the TournGolfer object, tourn_golfer,
whose golfer_id instance variable value matches the specified
golfer_id and whose tourn_id instance variable value matches the
specified tourn_id

When the matching TournGolfer object is found, return the
tourn_golfer_id from the TournGolfer object, tourn_golfer
"""
A gift: See starter code for this function