CSC 122 Python Applications

Lesson 2: Project 1-A – Wake Golf Tour

Objectives

In this assignment, students will:

- Use basic Python program constructs, including loops, input and output to text files, string processing, lists and dictionaries.
- Use basic Python object-orientated programming constructs, including creating classes, creating objects and calling their methods.
- Code Python functions that read data from formatted text files and place the data into class objects.
- Learn how to carefully read a detailed program design specification and translate it into code.

Python Programming Instructions

Please use PyCharm or a text editor (notepad++ or IDLE are 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 **WakeGolfTourA**. The Python source code will be in the **WakeGolfTourA** folder. Place the Python program file, **golf_tour.py**, the three class definitions files (**golfCourse.py**, **hole.py**, and **golfer.py)** and the **input files** in the **WakeGolfTourA** 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 **WakeGolfTourA** folder. Place the Python program file, **golf_tour.py**, the three class definitions files (**golfCourse.py**, **hole.py**, and **golfer.py)** and the **input files** in the **WakeGolfTourA** 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 **WakeGolfTourA** folder using directions from the document, **Creating and Submitting Programs**. Submit the **WakeGolfTourA.zip** file to Blackboard for credit.

Programs that are submitted incorrectly will not be graded.

Program Specifications

Project Description

Please read the **Wake Golf Tour App** document. This is a short introductory description and program information for Project 1. It begins by explaining the **Game of Golf**. Golf is an easy game to understand, but it is hard to play.

This is an object-orientated programming project. This project has its data divided into seven (7) different classes. The different project parts build on one another. The code for Project 1-B starts with the code from Project 1-A, and the code for Project 1-C starts with the code from Project 1-B, and so on. The last part of this project has you place the data from the seven (7) classes into a database that will later be used to build a website for Project 2. It also has you create simple SQL queries to retrieve database information.

Project Specification

Please read the **Wake Golf Tour Specification** document. This document delves deeper into how the object-orientated program is built. The code for the project is contained in **golf_tour.py**, except for the class definitions, which are in their own files. The mainline logic is in the **main** function, which calls the rest of the functions in the file. The **input files**, the input parameters and returned lists for the **create functions**, and **class definitions** for Project 1 are outlined in the **Wake Golf Tour Specification** document.

You are required to follow the specifications given. Example algorithms are be provided for each of the functions. In addition, the code for some of the functions will be provided. You should follow the algorithms 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.

Project Part A:

This first part of the project, **Project 1A**, has one Python program file called **golf_tour.py**. It also has three class definition files, **golfCourse.py, hole.py** and **golfer.py**. The main program contains 5 functions, including the **main** function. Details about these functions are given to you in comments within the functions themselves. The functions are summarized in the **Wake Golf Tour Specification** document.
In Project 1A, you are to create the **Golfer,** the **GolfCourse** and the **Hole class definitions** and object lists. You must read the **GolfCourse**, and **Golfer** data from two **input CSV files**. The input data is used to **create** the

Golfer and the **GolfCourse object lists**. Each of the **create functions** produces a list of class objects containing the data for a specific database table, including the id fields. The **create_golf_courses** function also creates and returns a dictionary, **golf_course_holes_dict,** to be used as input data for creating the **Hole objects list**. The data created from these functions are displayed on the screen and returned to be used as an input parameter to other **functions,** including the **write_objs_to_file** function which will save the data to a file. The object data is written to the screen and to the file using the class string (**__str__**) method.

Create Functions

create_golf_courses(input_filename): golf_courses_list, golf_course_holes_dict
create_holes(golf_course_holes_dict): holes_list
create_golfers(input_filename): golfers_list

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
write_objs_to_file
GolfCourse in **golfCourse.py**

Required Code

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

create_holes
create_golfers

You will also need to complete the class functions in these class files:

Hole in **hole.py**
Golfer in **golfer.py**

Program Starter Code

There is a zipped folder in Blackboard called **'Project1AStarterCode.zip'** that contains the **input files,** the starter **golf_tour.py** program and the class definition files, **golfCourse.py, hole.py** and **golfer.py** for **Project1A**. The two input files are in comma-delimiter format (*.CSV), where each record has its fields separated by commas. 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"

The code that reads in the files from a CSV file uses the Python ***csv*** module. The following article from the Internet, describes how to use this module.

Using the CSV Module (Ctrl-Click to Open in a new Window)

PlainText:
https://www.pythonforbeginners.com/csv/using-the-csv-module-in-python

Program Output Data

The input data are read into the **golf_tour.py** program in the **create_golf_courses** and **create_golfers** functions and the data returned is used to create a list of objects containing that data. The **create_golf_courses** function also returns a dictionary, **golf_course_holes_dict,** which is used as input to the **create_holes** function that uses it to create a list of objects containing that data.

The object lists are written to the screen, and to the following files:

Program Output Files

golf_courses_file = "golfCourses.csv"
holes_file = "holes.csv"
golfers_file = "golfers.csv"

Project Part A: create_golf_courses function

The algorithm, along with the code for the **create_golf_courses** function is given below.

```
"""
1. Create an empty list called golf_course_list that will
   contain GolfCourse objects whose data comes from the input
   file
2. Create a dictionary, golf_course_holes_dict, having the
   golf_course_id as the key, and a list of 18 tuples
   containing  (hole_num, par_value) as the value
   Each entry will have:
         golf_course_id: [(hole_num, par_value),
                          (hole_num, par_value),
                          ...,
                          (hole_num, par_value)]
```

```
3. Initialize the golf_course_id to 1
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: courses_list
    d. Create an outer loop to read each golf course in
       courses_list
       Outer Loop
       1. Get the golf course name from the first element
          stripped of whitespace.
       2. Create an empty list, hole_info, to hold the hole
          number and par value
       3. Create an inner loop to traverse the 18 hole
          par values using the range function
          Inner Loop
          a. Convert the string hole par values to integers
          b. Add par value to the total par
          c. Append hole_num and par value to the hole_info list
       4. Add entry for this golf course's hole_info to the
          golf_course_holes_dict
       5. Create a new GolfCourse object, call it golf_course,
          passing in golf_course_id, golf_course_name, and
          total_par
       6. Append the golf_course object to the golf_course_list
       7. Increment the golf_course_id

    e. Close input_file object
5. Print each golf_course object in the golf_course_list to the
   console
6. Return the golf_course_list
"""
import csv
def create_golf_courses (filename):
    print ("\nGolf Courses List: golf_course_list\n")

    golf_course_list = []
    golf_course_holes_dict = dict()
    golf_course_id = 1

    try:
        input_file = open(filename, 'r')
        file_lines = csv.reader(input_file)
        courses_list = list(file_lines)
        for golf_course in courses_list:
            golf_course_name = golf_course[0].strip()
            holes = []
            for i in range (1, 19):
                par = int(golf_course[i])
                total_par = total_par + par
                holes.append((i, par))

            golf_course_holes_dict[golf_course_id] = holes
            golf_course = GolfCourse(golf_course_id,
                          golf_course_name, total_par)

            golf_course_list.append(golf_course)
            golf_course_id = golf_course_id + 1

        input_file.close()

    except IOError:
        print ("File Not Found Error.")

    for gc in golf_course_list:
        print (gc)
    return golf_course_list, golf_course_holes_dict
```

Program Execution

Please **do not be overwhelmed** by the amount of reading for this course. There is no textbook. Thus, the lab documents are lengthy. The bulk of the reading assignments for the next 4 weeks are the Project 1 documents. 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 **Project1AStarterCodeFiles.zip** file, now!

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

3. Come to Open Lab to get help.

4. Use the Students Helping Students Discussion Board in Blackboard to get help.

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

6. 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

Golf Course List: golf_course_list

1,Raleigh Golf Course,72
2,WTCC Golf Course,72
3,Garner Golf Course,72
4,Cary Golf Course,72
5,Apex Golf Course,72

The Hole object list

1,1,1,4
2,1,2,3
3,1,3,4
4,1,4,4
5,1,5,4
6,1,6,5
7,1,7,4
8,1,8,4
9,1,9,4
10,1,10,4
11,1,11,3
12,1,12,4
13,1,13,4
14,1,14,4
15,1,15,5
16,1,16,4
17,1,17,4
18,1,18,4
19,2,1,4
20,2,2,4
21,2,3,3
22,2,4,4
23,2,5,4
24,2,6,4
25,2,7,5
```

```
26,2,8,4
27,2,9,4
28,2,10,4
29,2,11,4
30,2,12,3
31,2,13,4
32,2,14,4
33,2,15,4
34,2,16,5
35,2,17,4
36,2,18,4
37,3,1,4
38,3,2,4
39,3,3,4
40,3,4,4
41,3,5,5
42,3,6,4
43,3,7,4
44,3,8,4
45,3,9,3
46,3,10,4
47,3,11,4
48,3,12,4
49,3,13,5
50,3,14,4
51,3,15,4
52,3,16,4
53,3,17,3
54,3,18,4
55,4,1,4
56,4,2,4
57,4,3,4
58,4,4,4
59,4,5,3
60,4,6,4
61,4,7,5
62,4,8,4
63,4,9,4
64,4,10,5
65,4,11,4
66,4,12,4
67,4,13,4
68,4,14,3
69,4,15,4
```

70,4,16,4
71,4,17,4
72,4,18,4
73,5,1,4
74,5,2,4
75,5,3,4
76,5,4,5
77,5,5,4
78,5,6,4
79,5,7,3
80,5,8,4
81,5,9,4
82,5,10,4
83,5,11,5
84,5,12,4
85,5,13,4
86,5,14,4
87,5,15,3
88,5,16,4
89,5,17,4
90,5,18,4

The Golfer object list:

1,Jerry Woods,1987-05-27
2,Patton Perez,1996-02-10
3,Andy Palmer,1982-01-23
4,Danny Burger,1978-12-07
5,Wes Bryant,1977-08-05
6,Dusty Johns,1977-02-17
7,Lee Trevor,1992-08-21
8,Sergio Rose,1987-04-21
9,Justin Garcia,1994-08-14
10,Randy Fowler,1984-04-11
11,Tom Casey,1991-10-03
12,James Thomas,1979-12-22
13,Jack Nickels,1986-02-24
14,Joey Watson,1994-06-11
15,Mark Leech,1984-08-30
16,Russ Homey,1996-11-17
17,Alex Rahm,1977-04-08
18,Jordan Speed,1991-12-27
19,Jason Duffy,1990-08-28
20,Liam Horse,1984-10-02

```
21,Kevin Kissme,1986-09-02
22,Bryan Harmony,1989-01-16
23,Matsu Hidey,1994-10-25
24,Booth Koepka,1986-04-03
25,Phil Mickey,1993-03-19
26,Chaz Hoffman,1981-07-29
27,Matty Kuch,1981-01-21
28,Brendan Stool,1985-01-05
29,Kyle Stands,1987-09-17
30,Ashton Hadwin,1995-08-10

Process finished with exit code 0
```

Appendix

Example algorithms

create_holes

```
"""
1. Create an empty list called holes_list that will contain
   Hole objects whose data comes from the input dictionary
2. Initialize the hole_id to 1
3. Create an outer loop to read the golf course dictionary
   using the items() method to retrieve
       the key (golf_course_id), and
       hole information (hole_info)
   Outer Loop:
   for golf_course_id, hole_info in
       golf_course_holes_dict.items():

       Create an inner loop to read each golf course's
       hole information (hole_info)
       Inner Loop
       for info in hole_info:
           a. Create a new Hole object, call it hole_obj,
              passing in
                 hole_id, golf_course_id, hole_num, and
                 par_value
           b. Append the hole object to the holes_list
           c. Increment hole_id
4. Print the holes_list objects to the console
5. Return the holes_list
"""
```

create_golfers

```
"""
1. Create an empty list called golfer_list that will be filled
   in with Golfer objects whose data comes from the input file
2. Initialize the golfer_id to 1
3. 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_input_list
   d. Create a loop to traverse the golfer_input_list,
      where the loop variable 'golfer_info' will contain one of
```

```
        the lists in golfer_input_list at each loop iteration
        Loop:
            1. Get the golfer_name and golfer_birthdate
            2. Create a new Golfer object, call it player,
               passing in golfer_id, golfer_name, and
               golfer_birthdate
            3. Append the player object to the golfer_list
            4. Increment the golfer_id
    e. Close the input file
4. Print the golfer_list objects to the console
5. Return the golfer_list
"""
```