

## BIT100 Assignment 2

**Due Date:** Week 13

**Value:** 15%

**Assessment mode:** Individual Assessment

---

### Rationale

This assignment has been designed to give students the opportunity to demonstrate their skill in:

- solving a fairly complex problem involving the design of more than one user defined class;
- using container classes to maintain a list of objects and allow management of them.
- managing an application which involves a list of objects
- writing and using methods which enable objects to show desired behaviours
- using and complying with a supplied specifications for classes to be written
- saving data persistently
- using good programming style.

The assignment assesses the following learning outcomes

- CLO 3: Interpret algorithms and program code.
- CLO 4: Apply the concept of object orientation as an approach to data abstraction.
- CLO 5: Write programs to solve basic computing problems.

### SUBMISSION REQUIREMENT

Your assignment has to submit to TurnItIn, with the following all contain in a single file:

1. All your Python source files, printed in Word document format
2. Printed output (showing your interactivity with your program) is to be included at the end of your Python source files, in Word document created in (1)
3. A Turnitin Report, again to be attached within the Word document created in (1)

#### Turnitin Report (<http://www.turnitin.com>)

Register yourself in 100\_SF2020 using the following details:

**class ID :** 23961437

**Enrollment password:** python2020

### Overview

For this assignment you will write a small application which enables an organizer of a diving competition to compute the scores for each competitor and at the end of it, display the winner. The problem focuses on a container class of user-defined objects and the main features being assessed include your ability to handle several classes working together, the dynamic adding of new objects to a collection and searching the collection for particular objects.

### Task

You are to write three classes as described below:

- The first is a class called `Diver` which defines a simple object type representing a diving competitor.
- The second class called `Competition` defines objects which are containers of `Competitor` objects. There is no restriction on the number of competitor being stored in the container.
- The final class, called `DivingCompetition` defines an application which creates one `Competition` object and allows the various methods of `Competition` to be called. This class will be an interactive application using the keyboard and the screen to interact with a human operator. It will pass user inputs as arguments to methods of `Competition` class.

**NOTE:** The final application will only execute correctly when all three classes have been defined completely and correctly but don't wait until you have completely written all three before you start compiling and testing your code. It is recommended that you save the first two classes in a single file named `divingAssoc.py`, and the third class as a separate file.

## The classes

The classes you will require are:

### Diver class

This class defines participant taking part in a diving competition. The objects have the following attributes:

- `name` (of type 'str');
- `id` (of type 'int', and is to be automatically assigned by the system, starting with 1); and
- `scores` (of type 'list', which contains 7 floating-point numbers, representing the scores awarded to the diver by different judges. The score in a diving competition ranges from 0 to 10, inclusive, with increments of 0.5 points.)

The methods of class `Diver` should include:

- ❑ A constructor ( `__init__` ) which accepts TWO parameters for a diver – name, and scores. The constructor should initialise its object's attributes with these parameter values. Note that the `id` is NOT passed in as it will be automatically assigned when object is created. You need to declare a **class** variable named `nextDiverID`, that stored the next diver's ID, and initialised it with 1.
- ❑ A reader (getter) method for each of the three attributes. That is, a simple 'getter' method for the name, id, and the scores.
- ❑ A writer (setter) method ONLY for the attribute name.
- ❑ A method `getAverageScore` that compute a score that represents the average score of the 5 remaining scores, after removing the highest and lowest score.
- ❑ A string method ( `__str__` ) which return a single string containing the details of a competitor. Such a string can be formed by concatenating the values of the attributes, and the method `getAverageScore` with format as shown:

```
<name> (<id>)with scores <scores> and average score [getAverageScore]
```

It is recommended that once you have written the `Diver` class, you create a tiny program to test it. The testing program should be placed in the same working directory as the `Diver` class and be used to create one or two `Diver` objects and call some of the `Diver` methods. Compile the `Diver` class and compile and run the test program to check your work.

### Competition class

This class declares object which maintains a list of `Competitor` objects. It will contain methods which enable the list to show the appropriate behaviours as required by the menu. This class should be saved into the same file as `Competitor` class.

The `Competition` class will have the following attributes:

- ❑ A list for storing `Diver` objects
- ❑ A name that represents name of the competition, for example `HELP Open Diving Competition`.

The `Competition` class must also contain some methods which allow the collection of `Diver` to be managed. The methods of class `Competition` should include:

1. A constructor with one argument of type 'str', which is used to initialise the competition name. The constructor should also initialise an empty list for storing divers.
2. Getter (reader) and setter (writer) methods for the attributes, name and divers.
3. A method named **register** which accepts as an argument an object of class `Diver`. This method will store a reference to this `Diver` object into the next available cell in the list
4. A **highestAverageScoreSoFar** that does not accepts any argument, but returns the diver with the highest average score so far
5. A method named **noOfDivers** which returns the number of divers currently stored in the collection.
6. A string method **\_\_str\_\_()** that returns the details of all divers, one per line.
7. A method named **getDiver** which accepts an integer representing the index where the diver is stored in the list. If the index is out of bounds, the method should return `None`, otherwise the found diver with that index is returned. This method will be invoked when we need to edit the details of a diver.
8. A method **saveToFile** that accepts a string representing the filename to save, and all `Diver` objects will be saved to a text file, one per line, with each attribute separated by comma. Note: The diver's name will be saved as the first entry of a line in the text file.
9. A method **readFromFile** that accepts a string representing the filename where the data is to load from. Note: You need to set the `nextDiverID` to 1 before reading the data from file.

When you have written the `Competition` class - test it by creating a `Competition` object and invoking the methods from a small program.

### DivingCompetition class

The aim of this class is to provide a user-interface for a modest application which uses a `Competition` container class and should be saved in the same working directory as the previous file. It is recommended that this user-interface be written as a 'console' application using the normal screen and keyboard to interact with a user via a simple text-based menu.

The user-interface should create a single `Competition` object and provide a menu of choices to the user.

## Sample Runs

**Please note - I've truncated display of menu to save paper! The menu should display on the screen in full each time it is displayed.**

Enter competition title: `HELP Closed Diving Competition`

```
HELP Closed Diving Competition
~~~~~
1 Register diver
2 Display all divers
3 Display current leader
4 Update information of diver
5 Display all divers, sorted according to name or average score
6 Save data to file
7 Load data from file

0 Quit
```

Your choice? `2`  
No diver has signed up yet

```
HELP Closed Diving Competition
~~~~~
1 Register diver
:
7 Load data from file

0 Quit
```

Your choice? `0`  
Competition is cancelled due to lack of response.

---

Enter competition title: `HELP OPEN`

```
HELP OPEN
~~~~~
1 Register diver
2 Display all divers
3 Display current leader
4 Update information of diver
5 Display all divers, sorted according to name or average score
6 Save data to file
7 Load data from file

0 Quit
```

Your choice? `3`  
No diver has signed up yet

HELP OPEN

~~~~~

1 Register diver

:

7 Load data from file

0 Quit

Your choice? **1**

Adding diver

Name: **Superman**

Score 1: **6**

Score 2: **6.5**

Score 3: **7.0**

Score 4: **7**

Score 5: **7.5**

Score 6: **6.5**

Score 7: **7**

Addition success...

HELP OPEN

~~~~~

1 Register diver

2 Display all divers

:

7 Load data from file

0 Quit

Your choice? **2**

**Diver:**

1. Superman (1) with scores [6, 6.5, 7.0, 7, 7.5, 6.5, 7], and average score 6.80

HELP OPEN

~~~~~

1 Register diver

:

7 Load data from file

0 Quit

Your choice? **1**

Adding diver

Name: **Peter Parker**

Score 1: **8**

Score 2: **8**

Score 3: **8**

Score 4: **8**

Score 5: **7.5**

Score 6: **8.0**

Score 7: 8  
Addition success...

HELP OPEN

~~~~~

1 Register diver

:

7 Load data from file

0 Quit

Your choice? 1

Adding diver

Name: Alex

Score 1: 6.5

Score 2: 6.5

Score 3: 7.0

Score 4: 6.5

Score 5: 6.5

Score 6: 6

Score 7: 6.0

Addition success...

HELP OPEN

~~~~~

1 Register diver

2 Display all divers

:

7 Load data from file

0 Quit

Your choice? 2

**Divers:**

1. Superman (1) with scores [6, 6.5, 7.0, 7, 7.5, 6.5, 7], and average score 6.80
2. Peter Parker (2) with scores [8, 8, 8, 8, 7.5, 8.0, 8], and average score 8.00
3. Alex (3) with scores [6.5, 6.5, 7.0, 6.5, 6.5, 6, 6.0], and average score 6.40

HELP OPEN

~~~~~

1 Register diver

:

5 Display all divers, sorted according to name or average score

6 Save data to file

7 Load data from file

0 Quit

Your choice? 5

Sort according to <N>ame or <A>verage score? n

1. **Alex** (3) with scores [6.5, 6.5, 7.0, 6.5, 6.5, 6, 6.0], and average score 6.40
2. **Peter Parker** (2) with scores [8, 8, 8, 8, 7.5, 8.0, 8], and average score 8.00
3. **Superman** (1) with scores [6, 6.5, 7.0, 7, 7.5, 6.5, 7], and average score 6.80

HELP OPEN

~~~~~

1 Register diver

:

5 Display all divers, sorted according to name or average score

6 Save data to file

7 Load data from file

0 Quit

Your choice? **5**

Sort according to <N>ame or <A>verage score? **a**

1. Alex (3) with scores [6.5, 6.5, 7.0, 6.5, 6.5, 6, 6.0], and average score **6.40**

2. Superman (1) with scores [6, 6.5, 7.0, 7, 7.5, 6.5, 7], and average score **6.80**

3. Peter Parker (2) with scores [8, 8, 8, 8, 7.5, 8.0, 8], and average score **8.00**

HELP OPEN

~~~~~

1 Register diver

2 Display all divers

3 Display current leader

:

7 Load data from file

0 Quit

Your choice? **3**

Current leader is: Peter Parker (2) with scores [8, 8, 8, 8, 7.5, 8.0, 8], and average score 8.00

HELP OPEN

~~~~~

1 Register diver

:

4 Update information of diver

:

7 Load data from file

0 Quit

Your choice? **4**

Divers:

1. Superman (1) with scores [6, 6.5, 7.0, 7, 7.5, 6.5, 7], and average score 6.80

2. Peter Parker (2) with scores [8, 8, 8, 8, 7.5, 8.0, 8], and average score 8.00

3. Alex (3) with scores [6.5, 6.5, 7.0, 6.5, 6.5, 6, 6.0], and average score 6.40

Which diver to update? **4**

Invalid number

HELP OPEN

~~~~~

```
1 Register diver
:
4 Update information of diver
:
7 Load data from file

0 Quit
```


Your choice? **4**


Divers:

1. Superman (1) with scores [6, 6.5, 7.0, 7, 7.5, 6.5, 7], and average score 6.80
2. Peter Parker (2) with scores [8, 8, 8, 8, 7.5, 8.0, 8], and average score 8.00
3. Alex (3) with scores [6.5, 6.5, 7.0, 6.5, 6.5, 6, 6.0], and average score 6.40

Which diver to update? **1**

Superman (1) with scores [6, 6.5, 7.0, 7, 7.5, 6.5, 7], and average score 6.80

New name? <Enter> to skip 

Update score? <Y>es of <Ener> to skip 

Pressing <Enter> key

Update aborted

HELP OPEN

~~~~~

```
1 Register diver
:
4 Update information of diver
:
7 Load data from file

0 Quit
```


Your choice? **4**

Divers:

1. Superman (1) with scores [6, 6.5, 7.0, 7, 7.5, 6.5, 7], and average score 6.80
2. Peter Parker (2) with scores [8, 8, 8, 8, 7.5, 8.0, 8], and average score 8.00
3. Alex (3) with scores [6.5, 6.5, 7.0, 6.5, 6.5, 6, 6.0], and average score 6.40

Which diver to update? **3**


Alex (3) with scores [6.5, 6.5, 7.0, 6.5, 6.5, 6, 6.0], and average score 6.40


New name? <Enter> to skip 


Update score? <Y>es of <Ener> to skip **y**

Score 1 <Enter> to skip **7**


Score 2 <Enter> to skip **7.0**

Score 3 <Enter> to skip 

Score 4 <Enter> to skip 

Score 5 <Enter> to skip 

Score 6 <Enter> to skip **6.5**

Score 7 <Enter> to skip 

Information updated

HELP OPEN

~~~~~

```
1 Register diver
:
5 Display all divers, sorted according to name or average score
6 Save data to file
```



7 Load data from file

0 Quit

Your choice? 5

Sort according to <N>ame or <A>verage score? n

1. **Alex** (3) with scores [7, 7.0, 7.0, 6.5, 6.5, 6.5, 6.0], and average score 6.70
2. **Peter Parker** (2) with scores [8, 8, 8, 8, 7.5, 8.0, 8], and average score 8.00
3. **Superman** (1) with scores [6, 6.5, 7.0, 7, 7.5, 6.5, 7], and average score 6.80

HELP OPEN

~~~~~

1 Register diver

:

5 Display all divers, sorted according to name or average score

6 Save data to file

7 Load data from file

0 Quit

Your choice? 5

Sort according to <N>ame or <A>verage score? A

1. Alex (3) with scores [7, 7.0, 7.0, 6.5, 6.5, 6.5, 6.0], and average score **6.70**
2. Superman (1) with scores [6, 6.5, 7.0, 7, 7.5, 6.5, 7], and average score **6.80**
3. Peter Parker (2) with scores [8, 8, 8, 8, 7.5, 8.0, 8], and average score **8.00**

HELP OPEN

~~~~~

1 Register diver

:

6 Save data to file

7 Load data from file

0 Quit

Your choice? 6

File name to save? **divers.txt**

Data successfully saved to divers.txt

**divers.txt**

Superman,6,6.5,7.0,7,7.5,6.5,7

Peter Parker,8,8,8,8,7.5,8.0,8

Alex,7,7.0,7.0,6.5,6.5,6.5,6.0

HELP OPEN

~~~~~

1 Register diver

:

7 Load data from file

0 Quit

**The winner is displayed when program is terminated**

Your choice? 0

The winner is: Peter Parker (2) with scores [8, 8, 8, 8, 7.5, 8.0, 8], and average score 8.00

---

---

Enter competition title: **HELP CLOSED**

HELP CLOSED

~~~~~

- 1 Register diver
- 2 Display all divers
- 3 Display current leader
- 4 Update information of diver
- 5 Display all divers, sorted according to name or average score
- 6 Save data to file
- 7 Load data from file

0 Quit

Your choice? 2

No diver has signed up yet

HELP CLOSED

~~~~~

- 1 Register diver
- :
- 7 Load data from file

0 Quit

Your choice? 7

File name to load: **divers.txt**

Data successfully loaded from divers.txt

HELP CLOSED

~~~~~

- 1 Register diver
- 2 Display all divers
- :
- 7 Load data from file

0 Quit

Your choice? 2

Divers:

- 1. Superman (1) with scores [6, 6.5, 7.0, 7, 7.5, 6.5, 7], and average score 6.80
- 2. Peter Parker (2) with scores [8, 8, 8, 8, 7.5, 8.0, 8], and average score 8.00
- 3. Alex (3) with scores [7, 7.0, 7.0, 6.5, 6.5, 6.5, 6.0], and average score 6.70

HELP CLOSED

~~~~~

- 1 Register diver
- :
- 7 Load data from file

0 Quit

Your choice? 1

Adding diver

Name: Benjamin  
Score 1: 9  
Score 2: 9  
Score 3: 9  
Score 4: 9  
Score 5: 8.5  
Score 6: 9.5  
Score 7: 9.0  
Addition success...

HELP CLOSED  
~~~~~  
1 Register diver  
2 Display all divers  
:  
7 Load data from file  
  
0 Quit

Your choice? 2  
Divers:  
1. Superman (1) with scores [6, 6.5, 7.0, 7, 7.5, 6.5, 7], and average score 6.80  
2. Peter Parker (2) with scores [8, 8, 8, 8, 7.5, 8.0, 8], and average score 8.00  
3. Alex (3) with scores [7, 7.0, 7.0, 6.5, 6.5, 6.5, 6.0], and average score 6.70  
4. Benjamin (4) with scores [9, 9, 9, 9, 8.5, 9.5, 9.0], and average score 9.00

HELP CLOSED  
~~~~~  
1 Register diver  
2 Display all divers  
3 Display current leader  
:  
7 Load data from file  
  
0 Quit

Your choice? 3  
Current leader is: Benjamin (4) with scores [9, 9, 9, 9, 8.5, 9.5, 9.0], and average score 9.00|

HELP CLOSED  
~~~~~  
1 Register diver  
:  
5 Display all divers, sorted according to name or average score  
6 Save data to file  
7 Load data from file  
  
0 Quit

Your choice? 5  
Sort according to <N>ame or <A>verage score? N

1. **Alex** (3) with scores [7, 7.0, 7.0, 6.5, 6.5, 6.5, 6.0], and average score 6.70
2. **Benjamin** (4) with scores [9, 9, 9, 9, 8.5, 9.5, 9.0], and average score 9.00
3. **Peter Parker** (2) with scores [8, 8, 8, 8, 7.5, 8.0, 8], and average score 8.00
4. **Superman** (1) with scores [6, 6.5, 7.0, 7, 7.5, 6.5, 7], and average score 6.80

HELP CLOSED

~~~~~

1 Register diver

:

5 Display all divers, sorted according to name or average score

6 Save data to file

7 Load data from file

0 Quit

Your choice? **5**

Sort according to <N>ame or <A>verage score? **A**

1. Alex (3) with scores [7, 7.0, 7.0, 6.5, 6.5, 6.5, 6.0], and average score **6.70**
2. Superman (1) with scores [6, 6.5, 7.0, 7, 7.5, 6.5, 7], and average score **6.80**
3. Peter Parker (2) with scores [8, 8, 8, 8, 7.5, 8.0, 8], and average score **8.00**
4. Benjamin (4) with scores [9, 9, 9, 9, 8.5, 9.5, 9.0], and average score **9.00**

HELP CLOSED

~~~~~

1 Register diver

:

7 Load data from file

0 Quit

Your choice? **7**

Do you want to save current data? (Y/N) **Y**

File name to save? **bbc.txt**

Data successfully saved to bbc.txt

#### bbc.txt

Superman,6,6.5,7.0,7,7.5,6.5,7

Peter Parker,8,8,8,8,7.5,8.0,8

Alex,7,7.0,7.0,6.5,6.5,6.5,6.0

Benjamin,9,9,9,9,8.5,9.5,9.0

File name to load: **divers2.txt**

Data successfully loaded from divers2.txt

#### divers2.txt

James,5.5,5.0,5.5,5.0,5.0,5.5,7.0

Alex,5,5,5,5,5,5.5,5.5

Abraham,7.0,8.0,7,6.0,6.0,6.0,6.5

Danny,10,10,10,10,10,10,10

HELP CLOSED

~~~~~

1 Register diver

2 Display all divers

:

7 Load data from file

0 Quit

Your choice? **2**

Divers:

1. James (1) with scores [5.5, 5.0, 5.5, 5.0, 5.0, 5.5, 7.0], and average score 5.30
2. Alex (2) with scores [5, 5, 5, 5, 5, 5.5, 5.5], and average score 5.10

3. Abraham (3) with scores [7.0, 8.0, 7, 6.0, 6.0, 6.0, 6.5], and average score 6.50

4. Danny (4) with scores [10, 10, 10, 10, 10, 10, 10], and average score 10.00

HELP CLOSED

~~~~~

1 Register diver

:

7 Load data from file

0 Quit

Your choice? 1

Adding diver

Name: Christopher

Score 1: 8

Score 2: 7.5

Score 3: 7.5

Score 4: 7.0

Score 5: 7

Score 6: 8.0

Score 7: 8

Addition success...

HELP CLOSED

~~~~~

1 Register diver

2 Display all divers

3 Display current leader

:

7 Load data from file

0 Quit

Your choice? 3

Current leader is: Danny (4) with scores [10, 10, 10, 10, 10, 10, 10], and average score 10.00

HELP CLOSED

~~~~~

1 Register diver

:

5 Display all divers, sorted according to name or average score

6 Save data to file

7 Load data from file

0 Quit

Your choice? 5

Sort according to <N>ame or <A>verage score? A

1. Alex (2) with scores [5, 5, 5, 5, 5, 5.5, 5.5], and average score 5.10

```
2. James (1) with scores [5.5, 5.0, 5.5, 5.0, 5.0, 5.5, 7.0], and average score
5.30
3. Abraham (3) with scores [7.0, 8.0, 7, 6.0, 6.0, 6.0, 6.5], and average score
6.50
4. Christopher (5) with scores [8, 7.5, 7.5, 7.0, 7, 8.0, 8], and average score
7.60
5. Danny (4) with scores [10, 10, 10, 10, 10, 10, 10], and average score 10.00
```

HELP CLOSED

~~~~~

1 Register diver

:

7 Load data from file

0 Quit

Your choice? 0

The winner is displayed when program is terminated

The winner is: Danny (4) with scores [10, 10, 10, 10, 10, 10, 10], and average score 10.00

## Notes

1. Selecting menu item 5 **does not** affect the original data set, it merely shows how the data would look if sorted. That is selecting item 2 immediately after 5 should show the data set again unchanged.
2. Apart from the necessary entry of data via menu item 1, menu items should be selectable in any order.
3. The sample runs shown above are to be used as a guide to check whether your program is correct. It should be correct when your program produced the same output as shown. When you submit your own execution, you should use your OWN DATA, and not the one shown above. Marks will be deducted if you are using the sample run data.

## Documentation (All Levels)

- You should include comments in your code stating what each method does and explaining any complex sections of code.
- You should also include your name and student ID as comments within the code.
- You should of course use meaningful variable names so that your code is to some extent self-documenting.

## What To Submit

You should submit the following:

- A cover-sheet stating your name and student number.

- Printouts of your source code using **Courier-New 10 point** size font. Make sure that your long Python statements, if any, do not have the second line printed starting from the left-hand margin. You must **break** the **long statement** in **appropriate length**.
- You are **NOT** allowed to print in **landscape orientation**.
- Printouts demonstrating real interaction between yourself and your application
- And a compressed file containing the source files **.py** uploaded to elearning.
- Upload your program code, together with sample output, in Word format, to [turnitin.com](https://turnitin.com)

**NOTE:** Refer to the Excel file, *msA2BIT100\_SF20.xlsx*, for detailed breakdown of the marks allocated for each task, as well as the requirement for each task.

### **Note:**

If your program does not meet the requirements by the due date you should obtain help from the lecturer and notify the lecturer that you will submit the assignment late (marks will be deducted).

### **Note about testing and plagiarism**

It is very important that you **complete** this assignment **alone**. You may of course obtain general assistance from the lecturing staff in the subject and your peers, but the coding must be carried out yourself. It is normally quite easy to detect when two or more students work together on their coding.

It is also very important that the demonstration of the results of your program using the given test data is produced using the identical version of the program to the printout of your source code. **Students who hand in substantially similar assignments or whose programs do not match their demonstration of testing will fail the assignment.**

Any student suspected of copying, or of not producing the work himself or herself, can be called for **oral examination**, where the student will be expected **to demonstrate sufficient knowledge of the application** to show that it is his or her own original work.