CS917 Programming Coursework Assignment

Administrative Details

- This project must be submitted by **12pm, Friday 2nd November 2018**.
- This assignment counts for one-third of your total mark for this course.
- The work that you submit should be **your own work.** Do not be tempted to copy work, this assignment is **not a team exercise**. There are both human and automated techniques to detect plagiarised code.
- If you have general or administrative problems please email Adam Chester (a.p.chester@warwick.ac.uk) immediately.

Submission Requirements

- Some portions of your work may be marked by an automated test suite. You should make sure that you follow these instructions carefully; failure to do so will result in the severe loss of marks in this assessment.
- Your code should output data in the formats specified where required. Do not add in any additional input or output or add user interaction; failure to follow this may result in a severe loss of marks.
- All submissions should run under the default python 3 installation on the lab machines.
- You may use any functions defined in the Python standard library to complete this assignment. Other libraries such as NumPY, SciPy and pandas are **not** permitted.
- You should submit your solutions to the coursework through the Tabula system. You are required to submit three files:
    - part1.py
    - part2.py
    - part3.py
- You should not change the filenames, or submit any additional files.
- Before you submit your solutions, you should test your solutions to ensure that they produce the correct output. You might like to create additional input files to test them.

Part 1 - Data Extraction (25%)

To complete this part of the assignment, you will process a phone log which contains telephone numbers in international formats.

Your task is to:

1. Extract the telephone numbers for the UK
2. Remove the international dialling code, replacing it with a zero.
3. Order them lexicographically
4. Print them in a specific format.

The phone numbers have a prefix of +44 and are followed by ten digits. Phone numbers may include spaces.
Example valid numbers: +44 7777 888888, +441231231231, +44 12444 444 4 4
Example invalid numbers: +34 7777 888888, +441111111111222211

Your program will be invoked with a filename as follows:
> python phone.py <datafile>

The file phone_log.txt is provided as a sample data file. A sample of the file can be seen below:

```
Phone call from +55 3138 745765 to +60 5757 946460
```

```
Whatsapp from +36 6535 355865 to +38 6092 270440
Whatsapp from +8 8779 628569 to +51 8241 571938
Phone call from +20 3314 854413 to +60 4981 365126
```

Your program should output numbers in the format XXXXX XXXXXX, that is five characters then a space followed by six characters.

```
01111 111111
01234 111112
```

The mark allocation for this section is as follows, 15% for correctly functioning program, 10% for quality of code giving rise to the output.

Part 2 - Shapes on the plane (45%)

This part of the assignment does not focus on mathematical problems or any advanced programming techniques. Instead the focus is on your ability to solve a problem using a small number of classes in the Python programming language.

You are given a set of data stored in a text file. The data defines a set of shapes (circles and squares) defined in a 2D space. A sample of the data file format is shown below.

```
Circle          1.2222     3.333          1.44444
Circle          -0.3       0.13           0.0000005
Square           0.0       0.0222         6.5555
```

The format comprises:

1. The type of shape represented by the line
2. The x coordinate of the centre of the circle or the top left corner of the square
3. The y coordinate of the centre of the circle or the top left corner of the square
4. The radius of the circle or side length of the square

Your task is to write some code which will classify and analyse the set of shapes. The information that you need to extract from this set of data is:

1. The total number of shapes in the set. The largest and smallest area of each type of shape in the file.
2. The average (mean) area for each shape type in the file.
3. The standard deviation of areas of each shape in the file.
4. The median of the areas.

When reading the descriptions of the shapes from the file, you will need to ignore those which are “singular”, i.e. those circles for which the radius is equal to zero or squares which have a side length of zero, within a certain tolerance (described below).

Programming Instructions and Hints

The code should be structured in a object oriented manner, with classes describing their actions.

The files provided on the module webpage have some predefined methods and some methods which you are required to complete. You must keep the names of all methods as they are in the templates. The file defines five basic classes for you project.

- **Point** - which represents a point in 2D space
- **Shape** - a class which contains the Tolerance constant required.
- **Circle** - which extends Shape to add functionality specific to circles.
- **Square** - which extends Shape to add functionality specific to squares.
- **Assignment** - should use the above classes to do the required analysis.

In addition to the template file there are also two data files (small and large) for you to test your solutions with.

The Point Class

The point class should contain:

1. Constructors, setters, getters and the equals method which allows us to compare two points.
2. In particular the comparison between two points is important and different for the usual mathematical notion of equality. Due to the way that floating point numbers are represented in computer memory, we never usually test whether two floating point numbers are equal; just whether they are 'close' to within a given tolerance. The tolerance used for this course work is 0.000001 and this is stored as a constant accessible as **`Shape.TOLERANCE`**
3. Implement a distance method which computes the Euclidean distance between two points.

The Shape Class

This class is abstract, and cannot be instantiated. The tolerance used for calculations is stored as Shape.TOLERANCE. You do not need to edit this class.

The Circle Class

You will now need to define the methods which are pertinent to Circle.

1. You should start by defining two variables to store the centre of the circle as a Point and the radius of the circle as a floating point number.
2. Define standard constructors, getters and setters
3. Override __`str`__ to print out a string in the format below.
   `This circle has its centre at (2,3) and a radius of 2.4.`
4. The equals method compares two circles. Two circles are equal if they have the same centres and radii (within the given tolerance).
5. Complete the area method.
6. Complete the envelops method which tests whether the current instance of Circle envelops (contains) the shape provided as a parameter. The shape could be either a circle or a square.
7. Complete the compare method. This method should return -1 if this object's area is smaller than otherObject's area, 0 if they have the same area and 1 if this object has a larger area then otherObject

The Square Class

You will now need to define the methods which are pertinent to Square.

1. You should start by defining two variables to store the top-left corner of the square as a Point and the length of the side as a floating point number.
2. Define standard constructors, getters. setters
3. Override __`str`__ to print out a string in the format below.
   `This square's top left corner is at (2,3) and the side length is 4.`
4. The method equals that compares two squares. Obviously two squares are equal if they have the same corners and side lengths within the given tolerance.
5. Complete the area method.
6. Define the envelops method which tests whether the current instance of Square envelops (contains) the shape provided as a parameter. The shape could be either a circle or a square.
7. Complete the compare method. This method should return -1 if this object's area is smaller than otherObject's area, 0 if they have the same area and 1 if this object has a larger area then otherObject.

The Assignment Class

In this class you will perform the statistical analysis of the data.

Your analyse method will be called to compute the required statistics. Only non-singular shapes should be considered in the calculations.

You should then return the statistics via the methods defined in the file:
shape_count(self): - the number of shapes in the file
circle_count(self): - the number of circles in the file
square_count(self): - the number of squares in the file
max_circle_area(self): - the maximum area of a circle in the file
min_circle_area(self): - the minimum area of a circle in the file
max_square_area(self): - the maximum area of a square in the file
min_square_area(self): - the minimum area of a square in the file
mean_circle_area(self): - the mean of the circle areas
mean_square_area(self): - the mean of the square areas
std_dev_circle_area(self): - the standard deviation in the area of the circles
std_dev_square_area(self): - the standard deviation in the area of the circles
median_circle_area(self): - the median area of the circles
median_square_area(self): - the median area of the squares

You may add instance variables and methods as required, however the existing method names and parameters should not be altered.

You should create data files of you own to test your system.

The mark allocation for this section is as follows, 35% for correctly functioning program, 10% for quality of code produced.

Part 3 - Word Games (30%)

For this part of the assignment you are going to write the class that will represent a board for a word game (e.g. Scrabble or Words with Friends). This part of the assignment is more difficult than the first, and will require you to work with less guidance.

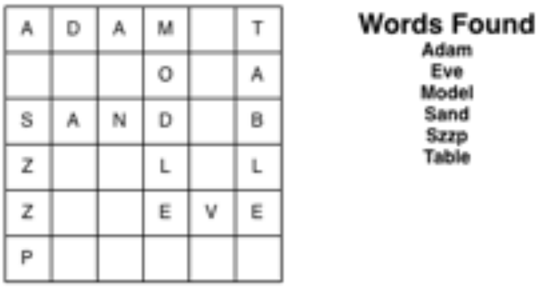
The LetterTile Class

This class is complete. You do not have to edit this class.

The GameBoard Class

This is the class that represents the board. It has a number of incomplete methods that you are required to implement. Please refer to the part3.py file for full description.

For the purpose of this assignment, a word is valid if it consists of any two or more characters e.g. AFFEETTRTWZ is a valid word. The image below is an example:



A	D	A	M		T
			O		A
S	A	N	D		B
Z			L		L
Z			E	V	E
P					

Words Found
Adam
Eve
Model
Sand
Szzp
Table

The mark allocation for this section is as follows, 15% for correctly functioning program, 15% for quality of code produced for the application.