PPA Assignment 5

This coursework is designed to test the content from Weeks 1, 2, 3, 4, and 5.

Do you think there is a problem with any of the content below? Let us know immediately at programming@kcl.ac.uk.

Read through this brief carefully before starting your attempted solution and make sure you follow it as closely as possible, paying particular attention to the names required for elements of your code. Also ensure that you comment your code, as part of the assignment documentation.

The total marks available for this assignment are 12.

For this week's assessment, consider the following scenario, and then complete the tasks that follow it:

We would like to create a simple simulation of an aeroplane travelling to various destinations. An aeroplane may travel to a single destination, which will cover a certain distance. It may also travel to multiple destinations over the course of a single day. Moreover, it can repeat its daily journeys over a number of days. After the aeroplane travels a certain distance, it is a requirement for it to be inspected for repairs, as a safety precation. This process takes 7 days, during which time the aeroplane cannot fly.

1. Model this scenario based on the following requirements:

    1. The location, of a destination or an aeroplane (at any point in time), is specified via `Coordinates`. This has an `x`, which is the value along the x-axis (on a map) of the location, and similarly a `y`, for the y-axis. These are both whole numbers. (1 mark)

    2. A `Destination` represents the town or city that an aeroplane can travel

to. It has a `name` and `coordinates`. (1 mark)

3. An `Aeroplane` has a `name` and `coordinates`. It also has a `speed`, which specifies how far the aeroplane can move in one hour in both the x-axis and the y-axis. Next, an aeroplane keeps track of the `totalDistance` that it has travelled, as the sum of the adjustments made to its x coordinate plus the the adjustments made to its y coordinate, across all of the journeys it has undertaken. Lastly, each aeroplane has a `repairDistance`, which is the distance that the areoplane can travel after which it must be taken away for 7 days to undergo repairs. (1 mark)

4. An `Aeroplane` can take a `singleFlight` which will attempt to move the aeroplane towards a supplied `destination`. This will also calculate and return the distance travelled in this single journey. The aeroplane moves towards its destination as follows:

    1. The aeroplane keeps moving towards the destination until its x coordinate matches the x coordinate of the of the destination, and its y coordinate matches the y coordinate of the destination.

    2. The x coordinate of the aeroplane is adjusted as follows. It will increase or decrease, depending on whether it is lower or higher, respectively, than the x coordinate of the destination. The aeroplane should never travel beyond the destination. Therefore, when the distance left to travel is less than the aeroplane's `speed`, the x coordinate is increased by only the distance left to travel. Otherwise, the x coordinate is increased by the `speed` of the aeroplane.

    3. The y coordinate of the aeroplane is adjusted in the same way as the x coordinate. Therefore, the aim is to adjust the aeroplane's y coordinate to match the destination's y coordinate. Again, ensure that the aeroplane never travels beyond the destination.

    4. The total distance that the aeroplane has travelled (across all of its journeys) must be updated with the distance of this single journey. The single journey distance must also be returned.

   (2 mark)

2. Create a class `FlightSimulation`, which can be compiled and run from the command line. Use this class to do the following (in order), using the classes and methods you have created for Question 1.

    1. Create a `Destination` and name the variable holding it *destination1*. Set

its attributes as follows:

- the `name` as *Zurich*
- the `x` coordinate as *10*
- the `y` coordinate as *35*

(1 mark)

2. Create a `Destination` and name the variable holding it *destination2*. Set its attributes as follows:

- the `name` as *Doha*
- the `x` coordinate as *15*
- the `y` coordinate as *45*

(1 mark)

3. Create a `Destination` and name the variable holding it *destination3*. Set its attributes as follows:

- the `name` as *Tokyo*
- the `x` coordinate as *150*
- the `y` coordinate as *125*

(1 mark)

4. Create an `Aeroplane` and name the variable holding it *aeroplane*. Set its attributes as follows:

- the `name` as *Wright Flyer*
- the `x` coordinate as *10*
- the `y` coordinate as *35*
- the `speed` as *17*
- the `totalDistance` as *0*
- the `repairDistance` as *2000*

(1 mark)

5. Make your aeroplane simulate its journeys in a single day, by doing the following:

1. Print the name of your aeroplane and the journey start destination (which is `destination1`) and the journey end destination, which is `destination2`. Make your aeroplane take a single flight to `destination2`. Then, print this single journey's distance and the total distance travelled by your aeroplane. Finally, print your aeroplane's current location coordinates.

2. Print the name of your aeroplane and the journey start destination (which is now `destination2`) and the journey end destination, which is `destination3`. Make your aeroplane take a single flight to `destination3`. Then, print this single journey's distance and the total distance travelled by your aeroplane. Finally, print your aeroplane's current location coordinates.

3. Print the name of your aeroplane and the journey start destination (which is now `destination3`) and the journey end destination, which is `destination2`. Make your aeroplane take a single flight to `destination2`. Then, print this single journey's distance and the total distance travelled by your aeroplane. Finally, print your aeroplane's current location coordinates.

4. Print the name of your aeroplane and the journey start destination (which is now `destination2`) and the journey end destination, which is `destination1`. Make your aeroplane take a single flight to `destination1`. Then, print this single journey's distance and the total distance travelled by your aeroplane. Finally, print your aeroplane's current location coordinates.

(1 mark)

6. Now, repeat the aeroplane's daily schedule of four single flights (as specified in the previous question) over 120 days. Guidelines for this are as follows:

1. Print the current day's number. Then, print the total distance travelled by the aeroplane up until the start of the day.

2. Now make your aeroplane complete it's daily schedule of four single flights, however this should only occur if the aeroplane has not travelled so far a total distance that it needs to be sent for repairs. Otherwise, the aeroplane must not take any flights for 7 days and its total distance travelled is reset to 0.

3. Once the 120 days have been completed, print out the number of times that the aeroplane had to undergo repairs.

(2 marks)

Once completed, submit your assignment using the link marked `Assignment 5: Nexus Submission Link' on KEATS.

You must complete the plagiarism and collusion training before submitting this assignment.

You must also submit complete documentation of your solution. You will find a sample piece of documentation in the Support section on KEATS marked `Sample Assignment Documentation'. Submit your documentation using the link marked `Assignment 5: Documentation Submission' on KEATS.

Students who do not submit documentation along with their code, or vice-versa, will receive a mark of zero.

Any submitted code or documentation that is found to be unduly similar to the code or documentation submitted by any other student(s), will result in a penalty for those involved.

Provisional mark for your code will be released on KEATS within one week of submission. Final assignment grades will be submitted to the exam board at the end of the semester, and will take into consideration the quality of your documentation and the quality of the comments written into your code directly.

For all other queries, see the Support section on KEATS, specifically the document marked `Introduction'.