Worksheet Seven

""Forty-two!", yelled Loonquawl. "Is that all you have to show for seven and a half million years's work?"
"I checked it very thoroughly", said the computer, "and that quite definitely is the answer. I think the problem, to be quite honest with you, is that you've never actually known what the questions is".
Douglas Adams, The Hitch Hikers Guide to the Galaxy 1979

Unit Learning Outcomes Addressed by this worksheet: 1, 2 & 3

Make electronic copies of all of your algorithms. Place these, along with your Java code, in your P07 directory.

For worksheets 7, 8 and 9 you will be developing a set of classes which will build upon each other (e.g. worksheet seven classes will be required for worksheet eight, worksheet seven and eight classes will be required for worksheet nine. It is therefore essential that you keep up to date with these exercises (remember the senior tutor and I are available for help).

Background
A major step forward in the understanding of the physical universe has allowed Earth scientists to develop technology (known as a star-portal) which is capable of creating stable work holes in space. Spaceships can be sent through these worm holes, allowing space travel across the universe. A network of star portals is be created by sending ships through existing portals with the equipment required to set up a new star portal at their destination. This is only possible because additional technology has been developed which ensures that every star portal maintains its position in space (i.e. zero drift). Travel is achieved by instructing a star portal to open a worm hole whose destination is the same as another star portal. If a traveller arrives at a location where there is not star portal then they will be marooned in space.

The Task
Other engineering teams are working to develop suitable spaceships with all of the technology required to allow safe exploration of deep space using the star portals. Your job will be to develop a set of classes which are to be used by the spaceships when navigating from one portal to the next.

The first thing required is a 3D coordinate system which can be used to specify the location in space of a star portal. All star portals are placed near a star (not to near of course!). Each hyper portal is located in exactly the same position relative to its star. Hence, input to a star portal is accomplished by specifying the star name and its location in 3D space. The hyper portal then makes the adjustment required so that the worm hole will locate its destination close to the hyper portal (and not the star!).

In order to define a coordinate system three beacons have been located in space. Each beacon defines an axis direction from our sun. Strangely enough these beacons have been named x-axis, y-axis and z-axis.

Other teams are developing classes and algorithms for compensating star drift. This means you can always assume that the stars, the beacons and the star portals are fixed in space and do not move.

Exercise One.
Design, in pseudo code, a class called DistanceClass which is used specify each component of a 3D coordinate (i.e. x, y and z will each be objects of DistanceClass). To allow for the large distances required, each distance will be described by two values:

1. lightYears (LY): an integer value representing the number of light years.
2. kiloMetres (km): a real value specifying the distance in kilometres which needs to be added to the distance in light years to get the precise distance.

For example 3 Light years, 678,986,354.678 km means a total distance of 3 light years plus 678,986,354.678 km.

. The class:

- Should be able to be constructed without specifying any information. The default measurement is 0 LY, 0.0 km.
- Should be able to be constructed using a measurement in LY and km supplied as arguments to the constructor.
- Should be able to be constructed using another DistanceClass object as an argument to the constructor.
- Should be able to update the value of light years and/or kilometers contained within a DistanceClass object.
- Can retrieve the LY value contained within the object.
- Can retrieve km value contained within the object.
- Can express distance in String form. e.g.:
  "15 LY and 689572.567 km"
- Should have an equals method. kilometer measurements are considered equal if they lie within three decimal places of each other.

Please note that 1 LY = 9,454,254,955,488 km.

Exercise Two.
Make a list of the tests required to fully test the functionality of DistanceClass. Develop a test harness which conducts the required constructor tests.

Exercise Three.
Translate your pseudo code design from exercises two and three into Java.

There are more exercises on the next page.

Exercise Four.
Design, in pseudo code, a class called LocationClass. The class is used to hold the location of a body in 3D space. Position is defined via x, y and z values where each value:

- is represented via a DistanceClass object.
- is relative to our sun being located at the Galactic origin and the x, y and z-axis being defined by vectors connecting the sun to the x-axis, y-axis and z-axis beacons.

The class should:

- Should be able to be constructed without specifying any information. The default ilocation is the location of our sun.
- Should be able to be constructed by supplying values for x, y and z as arguments (DistanceClass objects) to the constructor.
- Should be able to be constructed using another LocationClass object as an argument to the constructor.
- Should be able to IMPORT and EXPORT x, y and z (as DistanceClass objects).
- Can express the location in String form. e.g.:

```
"X = 15 LY and 689572.567 km:: Y = 3 LY and 967.321 km:: Z = 1 LY and 472.375 km"
```

- Should have an equals method.

Exercise Five.
Develop a minimal pseudo code test harness for LocationClass.

Exercise Six.
Translate your pseudo code design into Java.