Lab-Methods

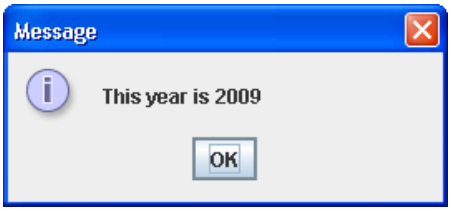
1. Unzip `Code.zip` onto your USB memory stick or C:\Users\ab123\Downloads\, and create a new project in NetBeans.
2. An application is required to calculate the volume of a fish tank, given its length, width and depth in millimetres. The volume should be given to the nearest litre.

    a) Analyse this problem. What are the inputs and outputs? What types of data are needed? What calculations are needed?

    b) Having determined the variables you will need, and their types (`long` and/or `double`), use NetBeans to write your program in a new class called `Tank`. Use the `JOptionPane` class for I/O.

    Notes:
    - The volume of a large tank, say 3000mm long by 1000mm wide by 1000mm deep may be too big to store as an `int`. Use `long` rather than `int` to avoid "loss of precision" errors.
    - There are 1000000 cubic millimetres in a litre. You can calculate the volume in litres (as a `double vol`) by multiplying the length, width and depth (in mm) and dividing by 1000000.0. Then the volume to the nearest litre is `Math.round(vol)`.

3. The following application (included as one of the examples) demonstrates the use of Java library classes `Calendar` and `GregorianCalendar`[†] for date manipulation.




```
import static javax.swing.JOptionPane.*;
import java.util.*;

class DateDemo {

    public static void main(String[] args) {
        GregorianCalendar now = new GregorianCalendar();
        int thisYear = now.get(Calendar.YEAR);
        showMessageDialog(null, "This year is " + thisYear);
    }
}
```

Calendar objects have several fields that you can *get* and *set*, including this one, **Calendar.MONTH**, **Calendar.DAY_OF_WEEK** and so on

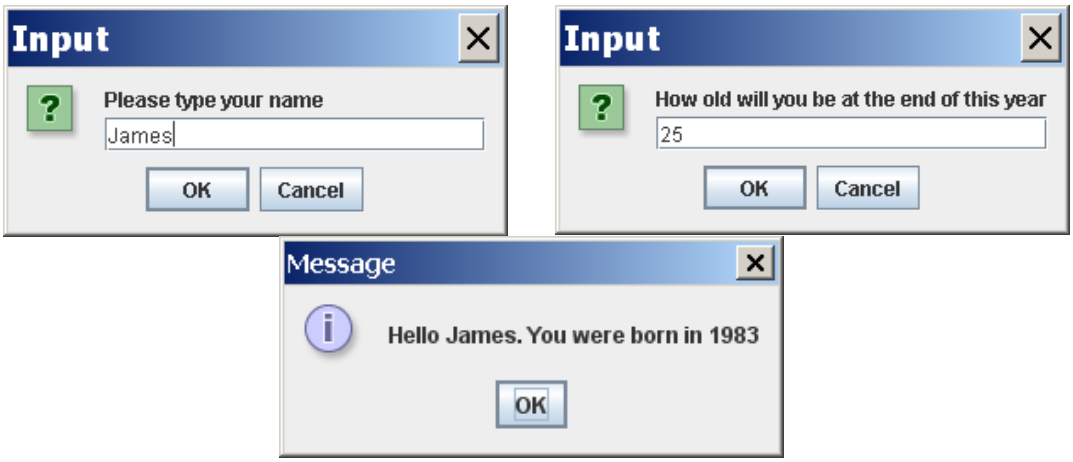
[†] The standard international calendar is the Gregorian Calendar which was introduced by Pope Gregory XIII in 1582 (see http://en.wikipedia.org/wiki/Gregorian_calendar). Java allows for other calendars but this is the only one in the API

Here a `GregorianCalendar` object `now` is created with the current date and time and the statement

```
int thisYear = now.get(Calendar.YEAR);
```

sets `thisYear` to 2013 (or 2014 if you run the program next January) using the `get` method of the `GregorianCalendar` class.

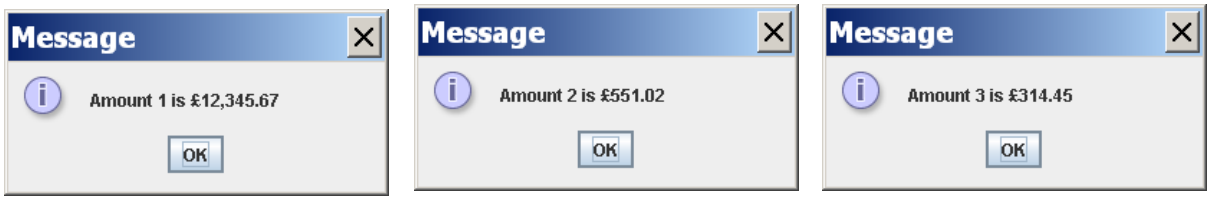
Make use of these classes in a similar fashion to modify `HelloAge2.java` so that it asks for the user's age *at the end of this year* and displays the user's year of birth. Save your modified program as `HelloAge5.java`.



4. The following application demonstrates the use of the Java library class `DecimalFormat`:

This is a *format string.* Format strings can be quite complicated, as here. This one causes the amounts to be displayed with the £ sign 'floating' to the first non-zero digit if there are more than two before the decimal point, and rounds off the fractional part to two significant figures. It would cause 0 to be displayed as £0.00

```
import static javax.swing.JOptionPane.*;
import java.text.DecimalFormat;
class PoundsDemo
{
   public static void main(String[] args)
   {
      DecimalFormat pounds = new DecimalFormat("£###,##0.00");
      double amount1 = 12345.667;
      double amount2 = 551.0204475308642;
      double amount3 = 314.4463734567901;
      showMessageDialog(null,
         "Amount 1 is " + pounds.format(amount1));
      showMessageDialog(null,
         "Amount 2 is " + pounds.format(amount2));
      showMessageDialog(null,
         "Amount 3 is " + pounds.format(amount3));
   }
}
```



Make use of the `DecimalFormat` class in a similar fashion to modify the Carpet Calculator application so that it always displays the carpet costs using

two decimal places as here (sometimes it will do this anyway, but only if you are lucky – see the examples on page 13 of the Java workbook).

Save the new program as `CarpetCalculator3.java`

5. Define a method

```
private static double calculateCost(double roomArea,
double carpetPrice) {
    return ... ;
}
```

to calculate the cost of a carpet. In place of ... you must write an expression involving `roomArea` and `carpetPrice` with the correct (`double`) value. Alternatively you can declare a `double` variable `answer` (say), assign the correct value to `answer` and finish with

```
    return answer;
```

6. Copy `CarpetCalulator2.java` `CarpetCalculator4.java` and modify it to make use of the `calculateCost` method you wrote in exercise 5.

7. Redesign the application `Tank.java` to make use of a suitable method `calculateVolume` which calculates the volume (in whole litres) of the fish tank. You will need a method something like

```
private static long calculateVolume(long l, long w, long d) {
    ...
    return ... ;
}
```

where you fill in the dots.

8. The application `Triangle.java` prompts the user for three arguments representing the sides of a triangle, and calculates and displays the area of that triangle using a well-known mathematical formula. (It assumes that the user types the three arguments correctly and that they do indeed form a triangle.)

Rewrite this application to make use of a function method

```
private static double findArea(double a, double b, double c)
...
```

which calculates the area of the triangle with sides `a`, `b`, `c`.