

SCHOOL OF COMPUTER SCIENCE



PRIFYSGOL  
**BANGOR**  
UNIVERSITY

# Object Oriented Programming in Java

---

## Laboratory 1 Revising the Basics

**Dave Perkins**

## Welcome to Object Oriented Programming in Java

Welcome to the first laboratory session for *ICP 1023: Object Oriented Programming in Java*. In these weekly sessions you will be provided with a range of exercises designed to help you master some of the basic concepts and techniques of object oriented programming. Three of these laboratory sessions involve assessed work and each piece of work must be completed and submitted for marking. Note that these exercises make up a *significant percentage* of your final mark for this course, so it is important that you submit them.

All of the material - lecture slides, notes, exercises - for this module will be delivered via the course organiser program known as *Blackboard*. The Blackboard site is located at:

<http://blackboard.bangor.ac.uk>

You will need to use your Bangor University *username* and *password* in order to log in to the system. If you cannot log in using these, please contact Dave Perkins in one of the laboratory sessions or by e-mail at:

[d.perkins@bangor.ac.uk](mailto:d.perkins@bangor.ac.uk)

Alternatively, ask one of the demonstrators during the laboratory sessions.

*Please submit all of your laboratory work via Blackboard.* Do not e-mail Java source code or any other material to me as it will not be marked!

You should try to complete the exercises within the 2 hours that have been timetabled in the laboratory. If you have been unable to complete the laboratory work in this period, make sure to finish it in your own time.

This week's laboratory is an introductory session, designed to revise some of the basics from the preceding module *ICP 1022 – Programming Fundamentals*.

## Exercise 1: Compute Your Grade in ICP 1023

The algorithm for computing a student's grade in the module *ICP 1023* is given below:

- Calculate mark for class tests as a percentage and then weight by 0.4.
- Calculate mark for laboratory work as a percentage and then weight by 0.3.
- Calculate mark for programming assignment as % and then weight by 0.3.
- Add weighted totals to derive final percentage.

Note that there are *four* assessed pieces of laboratory work and *four* class tests. To clarify the algorithm a fully worked example is shown below:

### Class Tests (Marked out of 20)

Class Test 1	7/20
Class Test 2	13/20
Class Test 3	15/20
Class Test 4	10/20
<b>Total</b>	<b>45/80 = 56%</b>

### Laboratory Work (Marked out of 10)

Lab 1	8/10
Lab 2	8/10
Lab 3	7/10
Lab 4	9/10
<b>Total</b>	<b>32/40 = 80%</b>

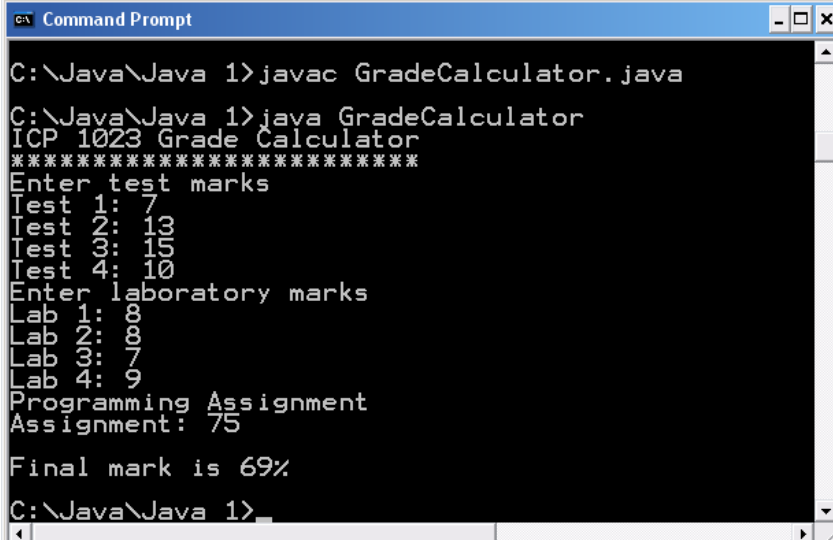
### Programming Assignment

Assignment	<b>75%</b>
------------	------------

$$\begin{aligned}\text{Weighted Total} &= (56\% * 0.4) + (80\% * 0.3) + (75\% * 0.3) \\ &= 22.4 + 24 + 22.5 = 68.9 \\ &= \mathbf{69\%}\end{aligned}$$

Once you are sure that you understand how a grade is calculated write a Java program called **GradeCalculator** to compute a student grade based on marks data entered via the keyboard.

This program works as follows:



```
C:\Java\Java 1>javac GradeCalculator.java
C:\Java\Java 1>java GradeCalculator
ICP 1023 Grade Calculator
*****
Enter test marks
Test 1: 7
Test 2: 13
Test 3: 15
Test 4: 10
Enter laboratory marks
Lab 1: 8
Lab 2: 8
Lab 3: 7
Lab 4: 9
Programming Assignment
Assignment: 75
Final mark is 69%
C:\Java\Java 1>
```

Figure 1 Sample Data for GradeCalculator

Your solution should have the following form:

```
public class GradeCalculator
{
    public static void main(String[] args)
    {
        // All code should go here
    }
}
```

Note that in this solution no use is made of methods other than **main**.

## Exercise 2: GradeCalculator with Methods

Using *stepwise refinement*, the problem of grade calculation can be broken down into a series of sub-problems:

- Get marks for class tests and compute summary mark;
- Get marks for laboratory work and compute summary mark;
- Get mark for programming assignment;
- Add weighted marks to calculate final mark;
- Display final mark.

This example of *problem decomposition* can be used to develop a program built out of a collection of methods, each of which corresponds to one of the sub-problems listed above; you should therefore restructure your solution to Exercise 1 on the basis of this decomposition.

## Exercise 3: File Processing

Assume that student marks for the module *ICP 1023* are stored in a text file called **marks.dat**. The marks for *each* student are recorded on a *single line* and are represented as Comma Separated Values(CSVs) For example,

**7, 13, 15, 10, 8, 8, 7, 9, 75**

These values correspond to the example given in Exercise 1.

Write a program to read a *single* set of marks stored in a file and display the corresponding grade. Once this program is working, extend it so that it can handle a file containing one or more set of student marks.

Some sample data is provided below:

**7, 13, 15, 10, 8, 8, 7, 9, 75**

**12, 12, 13, 15, 9, 9, 10, 9, 80**

**0, 3, 7, 9, 9, 7, 0, 5, 50**

Using the algorithm described in Exercise 1 perform some calculations to determine the final grade. Use these results to test your program.

## Exercise 4: Using a Generator to Create Test Data

Use a random number generator to create a text file containing ten sets of student marks. Be careful when generating the integer values for each line to comply with the following conditions:

- The first four integer values are in the interval [0,10]
- The next four values are in the interval [0,20]
- The last value is in the interval [0, 100]

Using this data re-run the program developed in Exercise 3.

## Exercise 5: Adding Functionality (Challenge)

Extend the program developed in Exercise 3 so that it can calculate the *mean*, *mode* and *median* with respect to a set of final marks. For a reminder of the GCSE mathematics involved see:

[http://www.bbc.co.uk/bitesize/ks2/maths/data/mode\\_median\\_mean\\_range/read/1/](http://www.bbc.co.uk/bitesize/ks2/maths/data/mode_median_mean_range/read/1/)

Use the data generated in Exercise 4 to test your solution. Also ensure that the program makes effective use of methods.

\*\*\*\*\*

## Useful Links

The Internet provides a huge mass of information (and disinformation) about Java. Some of the more useful sites are listed below:

- Oracle
  - <http://www.oracle.com/technetwork/java/index.html>
  - <http://docs.oracle.com/javase/tutorial/>
  - <http://docs.oracle.com/javase/1.4.2/docs/tooldocs/windows/java.html#seealso>
- Java World
  - <http://www.javaworld.com/>
- The O'Reilly Java site
  - <http://oreilly.com/java/>
- IBM provides a lot of technical information about Java at:
  - <http://www.ibm.com/developerworks/java/>
- Some good tutorial sites include:
  - <http://www.javacoffeebreak.com/tutorials/index.html>

Note that the above list is a work in progress so if you know of any other good sites let me know. Equally, if you find a link no longer works please let me know.

## NetBeans Tutorials

If you have never used NetBeans you may find the following tutorials helpful.

- Oracle's official and very comprehensive tutorial site  
<https://netbeans.org/kb/docs/java/quickstart.html>
- Good set of beginners tutorials from Columbia University  
[http://www.cs.columbia.edu/~cmurphy/summer2008/1007/netbeans/1\\_intro.html](http://www.cs.columbia.edu/~cmurphy/summer2008/1007/netbeans/1_intro.html)