# COM1003 Java Programming
# Autumn Semester 2022-3
# Programming Assignment 1

Dr Siobhán North
Department of Computer Science, The University of Sheffield

## Learning outcomes

This assignment will assess your ability to:

- Write a program from a specification;
- Write clear, good quality program code;
- Use numbers and strings in Java;
- Use the `sheffield` package.

This assignment is worth 12% of your mark for the first semester of the module and must be submitted by Friday 21st October. You will find information about the exact deadline, the marking scheme and how you must submit your work at the end of this document.

You do not need to do everything described below to hand in your work and get marks for it. The marks will be awarded depending on how much you have achieved (see below for details) but if you hand in a program it must compile and run to get any marks *including marks for programming style*. So a well written program that does something is always better than a program which would have done more if it had worked.

The specification below is quite precise in telling you what sort of input you should expect and what you must output but usually does not tell you how to get from one to the other. This is deliberate and part of the test. However it is not meant to be ambiguous so if there is something you don't understand you can ask questions, ideally by email. I will put all the email questions and their answers on an FAQ page as they come in but I will not answer questions by email that I have already answered on the FAQ page. Even if you think you understand everything it is a good idea to check this page before you submit.

## The First Task

Write a Java program called `Assignment1` which, when it is run, asks for a weight as a number of pounds, reads it in and outputs a message saying what that weight is in kilograms using the conversion rate 1 pound = 0.453592 kilogram.

The program should ask for the weight in pounds using the wording "Please type in a weight in pounds : " and expect an integer as input. The output must appear on the following line and must say "That is " then, on the same line, the weight in kilograms to three decimal places followed, still on the same line, by " kilograms" followed by a blank line. So if your input was 123 your command window should look like this

```
U:\myjava>java Assignment1
Please type in a weight in pounds : 123
That is 55.792 kilograms


U:\myjava>
```

Remember that following the specification is part of the test. Your program must be called `Assignment1`, the number of decimal places shown and the input and output text must be letter perfect copies of what appears above including the spaces and punctuation. You can only use the facilities of the `sheffield` package for all your input and output and you should use them as well as possible - see the marking scheme at the end of this document.

You can assume the user of your program will type in an integer when asked for a number of pounds and you need make no allowances for the possibility that they do anything else.

If you get this far with a perfect program you can expect to get 40%

## The Second Task

Weight is a function of gravity. Your program should open a file called `planets.txt` (stored in the same directory as the program) and read it in. The first line of the file will tell you the surface gravity on another planet. For example the first line of the file could be "Jupiter's gravity is 2.53g". It is in the file I have given you to practice with although that is not the file your program will be tested with. The test file will contain a line in the same format so the name of a planet, followed by "'s gravity is ", then a number, then the letter "g".

Given that the surface gravity on Earth is 1g the weight of something on Jupiter is 2.53 times as heavy. Modify your program to use this information. After the output from Task 1 first print out the line from the file and two more lines saying what the weight the user typed in earlier is on both Earth and the planet from the file. Again end the output from the task by a blank line. So your output now could be as shown on the next page.

```
U:\myjava>java Assignment1
Please type in a weight in pounds : 123
That is 55.792 kilograms

Jupiter's gravity is 2.53g
123 pounds on Earth weighs 55.7918 kilograms
123 pounds on Jupiter weighs 141.1533 kilograms


U:\myjava>
```

Again the wording must be exact. So the integer typed in at the start followed by "pounds on " then the name of the planet, the weight on that planet this time to four decimal places and finally the word "kilograms". And you are still restricted to the `sheffield` package for all your input and output.

If you get this far with a perfect program you will get 80%

## The Third Task

Actually the file contains a second line giving the gravity on another planet in the same format as the first line. Read that in too and then, after the Task 2 output, output information about the weight of the initial value on all three planets but in a different format like this.

```
U:\myjava>java Assignment1
Please type in a weight in pounds : 123
That is 55.792 kilograms

Jupiter's gravity is 2.53g
123 pounds on Earth weighs 55.7918 kilograms
123 pounds on Jupiter weighs 141.1533 kilograms

Earth           55.79
Jupiter        141.15
Mercury         21.20


U:\myjava>
```

Assuming that the second line in the file was "Mercury's gravity is 0.38g" So the names of the planets appear on the left and the weight of the original number of pounds converted to kilograms on that planet appears on the right to two decimal places but the rightmost digits must line up vertically so each line will need to be 20 characters long. Finally the three lines are followed by one blank line. As it happens the length of the words Jupiter and Mercury are the same and the gravity is given, in your test file, to the same number of decimal places but this may not be true of the file I test your program on. However you can assume that both planets will have a gravity of less than 10g although not that they will be planets of our solar system.

The same rules about getting the layout letter perfect and using only the sheffield package apply.

You will need to do all this with a perfectly written program to get 100%.

## Submission and deadline

You must submit a file called `Assignment1.java` via the submission point on Blackboard by 3pm on Friday 21st October. Do not submit anything else and do not submit it in any other format. Blackboard is very bad at displaying Java programs so you may think that Blackboard has messed up the layout of your program but I will download the program text and the layout will be preserved.

You can submit your work multiple times so you can submit the first task when you have got it working, to be sure you have those marks safely banked before your edit your program to start the next task and the same with the other tasks. The last submission before the deadline is the one that will be marked or, if your first submission is after the deadline, the first submission after the deadline but late work will be penalised using the standard University scale (a penalty of 5% per working day late; work will be awarded a mark of zero if it is more than 5 working days late).

This is an individual assignment. You must work on it alone and hand in your own work. If you work collaboratively and then pretend you did the work alone we will find out (we have a very good plagiarism checker and all submitted work will go through it) and we take the use of unfair means in the assignment process very seriously. Don't even think about handing in work you didn't do yourself.

## The Marking Scheme

This marking scheme is designed to test how well you know the work from weeks 1 and 2. So to get the marks you need to demonstrate skills from these lectures. Using different skills to achieve the same effect will not get you the marks.

I am aware that the clause that says that if you hand in a program it must compile and run to get any marks including marks for programming style is draconian but a program is not an essay, it is intended to do a task and if it does nothing it is useless. The reason for this rule for the first assignment is because such a high proportion on the marks rest on the program's style and I intend to make sure that well formatted gibberish gets zero. Having said that I will allow style marks only to stand for honest attempts that fail to compile or run due to a single trivial error and I will test compile all the submissions immediately after the deadline and allow students whose programs don't compile and, to a limited extent, run to resubmit albeit with a lateness penalty so check your email on the next working day.

See the next page for the marking scheme details.

### The Working Program

| | Task 1 | Task 2 | Task 3 |
|---|---|---|---|
| Style | 14 | 14 | 7 |
| Meeting Specification | 9 | 7 | 5 |
| Use of sheffield package | 9 | 9 | 4 |
| Calculation and String Manipulation | 8 | 10 | 4 |
| Total | 40 | 40 | 20 |

* The mark for following the specification means the specification as far as the program is completed so incomplete programs will not be penalized under this heading.

### Program Style

The Programming Style mark will be calculated as follows and scaled to the

| | |
|---|---|
| Correct use of variables | 30 |
| Correct use of types | 5 |
| Correct use of constants | 15 |
| Indention and layout | 25 |
| Comments | 25 |

maximum for whatever version you submit. There is a mark for comments and adding your name at the top of the program is best practice but please don't do it because the assignments are supposed to be marked anonymously.