# Department of Computer Science and Information Technology
# La Trobe University

# CSE4OOF Semester 1, 2019
# Assignment Part B

8%

This is an individual Assignment. You are not permitted to work as a Pair Programming partnership or any other group when writing this assignment.

## Due Date

Due: Tuesday 30th of April 2019 at 10.00 a.m.
Execution test: Week 8

Delays caused by computer downtime cannot be accepted as a valid reason for a late submission without penalty. Students must plan their work to allow for both scheduled and unscheduled downtime. **There are no days late or extensions on this assignment as execution test marking will begin on in your normal lab - Week 8.** After the submit server has closed, NO assignments can be accepted. Non-attendance at the **Week 8** lab you have signed up for on the LMS page will also result in your assignment being awarded 0, except as detailed below.

## Copying, Plagiarism

Plagiarism is the submission of somebody else's work in a manner that gives the impression that the work is your own. The Department of Computer Science and Information Technology treats academic misconduct seriously. When it is detected, penalties are strictly imposed. Refer to the subject guide for further information and strategies you can use to avoid a charge of academic misconduct.

## Assessment Objectives

- To practise using the **String** class. (Lecture/Workshop 2)
- To practice using switch and/or multiway selection branches
- To practise reading from and writing to the console and reading from a text file. (Lecture 4, Lab 3)
- To practise using loops (repetition)

## Submission Details and marking

Full instructions on how to submit electronic copies of your source code files from your **latcs8** account are given on **page 2**. All assignments in CSE4OOF are marked, face to face, in the lab, in an execution test. This means that we mark running code. Your code must compile and display a result to the screen. Regrettably, we don't have the time or resources to look at code. The smallest amount of code that produces and displays a correct result will gain more marks than lots of code that doesn't compile, run or display something to the screen. If you cannot attend the lab you have signed up for on the LMS page, please email me (O.Mahdi@latrobe.edu.au) to arrange another time.

## Marking summary

This assignment is worth **3%** of your final mark in this subject.
***Implementation*** **(Execution of code) 80%, explanation of code 20%**

| **Instant zeros or heavily reduced marks** | **Not submitting code**<br>**Using arrays**<br>**Not attending marking session**<br>**Not able to explain code that has not been taught yet**<br>**Does not run on latcs8**<br>**Uses System.exit()** |
|---|---|
| ChemicalReactionBalance.java | **100% (80% Execution and 20% Explanation)** |

## Using code not taught in OOF

Please also note carefully that whilst we encourage innovation and exploring java beyond what has been presented in the subject to date, **above all, we encourage understanding**.

All of the Tasks that follow can be solved using techniques that have been presented in lectures, lecture / workshops and labs so far.

These are the techniques and knowledge that we will later be examining in the Real Time Test (20 marks) and the exam (50 marks).

Code and techniques that are outside the material presented will not be examined, of course.

**For this task you cannot use arrays**

Other than this, you are free to solve the tasks below in any other way, with one exception and one condition.

Any assignment that uses code that is outside what has been presented to this point **must be fully explained at the marking execution test**. Not being able to **fully** explain code outside what has been presented in the subject so far will **result in the assignment being awarded a mark of 0**, regardless of the correctness of the program.

Submitting an assignment with code outside what has been presented so far and not attending the marking execution test will result in an automatic mark of 0, regardless of the correctness of the submission.

An example would be the **`split( )`** method in the **`String`** class. The reason being that this method returns an array and we haven't studied arrays yet. So using the **`split( )`** method would require you to be prepared to explain to the marker how arrays worked in Java.

## Electronic Submission of the Source Code

- Submit all the Java files that you have developed in the tasks above.
- The code has to run under Unix on the latcs8 machine.
- You submit your files from your latcs8 account. Make sure you are in the same directory as the files you are submitting. Submit each file separately using the **`submit`** command.

**submit MOF ChemicalReactionBalance.java**

After submitting the files, you can run the following command that lists the files submitted from your account:

**verify**

You can submit the same filename as many times as you like before the assignment deadline; the previously submitted copy will be replaced by the latest one.

Please make sure that you have read page 1 about the submission close off date and time and the compulsory requirement to attend the execution test in **Week 8**

Failure to do both of these things will result in your assignment be awarded a mark of 0, regardless of the correctness of the program.

Execution test marks are provisional and subject to final plagiarism checks and checks on the compliance of your code to this assignment document.

As such, final assignment marks may be lower or withdrawn completely.

## Background - Task

You are tasked with writing **one** Java program related to chemical compounds and elements. In the task, given a set of compounds, two reactants and two products, and you will determine if the reactants and products are balanced.

## Background – Chemical Elements

A chemical element is a type of atom that make up all the ordinary mater in the universe, and is define as an atom that has the same number of protons in its nucleus. Currently there are 118 known elements. These 118 can be represented on the periodic table of chemical elements as shown below:

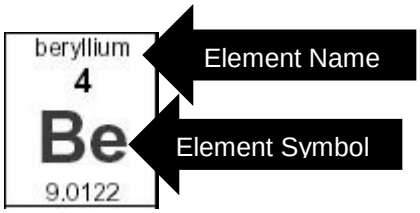
| | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| hydrogen 1 H 1.0079 | | | | | | | | | | | | | | | | | helium 2 He 4.0026 |
| lithium 3 Li 6.941 | beryllium 4 Be 9.0122 | | | | | | | | | | | boron 5 B 10.811 | carbon 6 C 12.011 | nitrogen 7 N 14.007 | oxygen 8 O 15.999 | fluorine 9 F 18.998 | neon 10 Ne 20.180 |
| sodium 11 Na 22.990 | magnesium 12 Mg 24.305 | | | | | | | | | | | aluminium 13 Al 26.982 | silicon 14 Si 28.086 | phosphorus 15 P 30.974 | sulfur 16 S 32.065 | chlorine 17 Cl 35.453 | argon 18 Ar 39.948 |
| potassium 19 K 39.098 | calcium 20 Ca 40.078 | scandium 21 Sc 44.956 | titanium 22 Ti 47.867 | vanadium 23 V 50.942 | chromium 24 Cr 51.996 | manganese 25 Mn 54.938 | iron 26 Fe 55.845 | cobalt 27 Co 58.933 | nickel 28 Ni 58.693 | copper 29 Cu 63.546 | zinc 30 Zn 65.39 | gallium 31 Ga 69.723 | germanium 32 Ge 72.61 | arsenic 33 As 74.922 | selenium 34 Se 78.96 | bromine 35 Br 79.904 | krypton 36 Kr 83.80 |
| rubidium 37 Rb 85.468 | strontium 38 Sr 87.62 | yttrium 39 Y 88.906 | zirconium 40 Zr 91.224 | niobium 41 Nb 92.906 | molybdenum 42 Mo 95.94 | technetium 43 Tc [98] | ruthenium 44 Ru 101.07 | rhodium 45 Rh 102.91 | palladium 46 Pd 106.42 | silver 47 Ag 107.87 | cadmium 48 Cd 112.41 | indium 49 In 114.82 | tin 50 Sn 118.71 | antimony 51 Sb 121.76 | tellurium 52 Te 127.60 | iodine 53 I 126.90 | xenon 54 Xe 131.29 |
| caesium 55 Cs 132.91 | barium 56 Ba 137.33 | 57-70 * | lutetium 71 Lu 174.97 | hafnium 72 Hf 178.49 | tantalum 73 Ta 180.95 | tungsten 74 W 183.84 | rhenium 75 Re 186.21 | osmium 76 Os 190.23 | iridium 77 Ir 192.22 | platinum 78 Pt 195.08 | gold 79 Au 196.97 | mercury 80 Hg 200.59 | thallium 81 Tl 204.38 | lead 82 Pb 207.2 | bismuth 83 Bi 208.98 | polonium 84 Po [209] | astatine 85 At [210] | radon 86 Rn [222] |
| francium 87 Fr [223] | radium 88 Ra [226] | 89-102 ** | lawrencium 103 Lr [262] | rutherfordium 104 Rf [261] | dubnium 105 Db [262] | seaborgium 106 Sg [266] | bohrium 107 Bh [264] | hassium 108 Hs [269] | meitnerium 109 Mt [268] | ununnilium 110 Uun [271] | unununium 111 Uuu [272] | ununbium 112 Uub [277] | | ununquadium 114 Uuq [289] | | | |

| | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| *Lanthanide series | lanthanum 57 La 138.91 | cerium 58 Ce 140.12 | praseodymium 59 Pr 140.91 | neodymium 60 Nd 144.24 | promethium 61 Pm [145] | samarium 62 Sm 150.36 | europium 63 Eu 151.96 | gadolinium 64 Gd 157.25 | terbium 65 Tb 158.93 | dysprosium 66 Dy 162.50 | holmium 67 Ho 164.93 | erbium 68 Er 167.26 | thulium 69 Tm 168.93 | ytterbium 70 Yb 173.04 |
| **Actinide series | actinium 89 Ac [227] | thorium 90 Th 232.04 | protactinium 91 Pa 231.04 | uranium 92 U 238.03 | neptunium 93 Np [237] | plutonium 94 Pu [244] | americium 95 Am [243] | curium 96 Cm [247] | berkelium 97 Bk [247] | californium 98 Cf [251] | einsteinium 99 Es [252] | fermium 100 Fm [257] | mendelevium 101 Md [258] | nobelium 102 No [259] |

Periodic Table
Source: LeVanHan
Creative Commons

## Background – Atomic Numbers and Element Symbols

**The elements are all given names but are often represented as a series of letters (element symbol), starting with an Upper-case letter and following with zero, one or two lower case letters**. These can be seen on the periodic table as shown below.




Beryllium
Source: LeVanHan
Creative Commons

## Background – “element_list.csv”

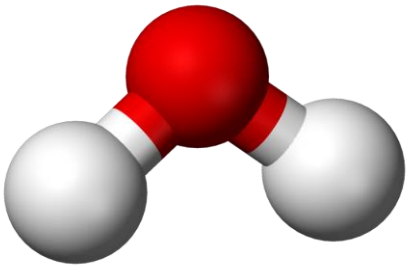
A list of all known elements has been put into a file for you named “**element_list.csv**” and is guaranteed to be without errors and 118 lines long. **(Note: The elements are not in Alphabetical order)**
**You will not need this for this task, but is included for your interest.**

This file can be copied from the csilib area

```
cp /home/student/csilib/cse4oofbu/assignBu19/assignB/element_list.txt .
```

## Background – Chemical compounds

A chemical compound or molecule is collection of atoms of more than one chemical element held together by chemical bonds. A common example of this is pure water, which is made up of two Hydrogen atoms and one Oxygen atom that are chemically bound.



Model of a Pure Water Molecule
Public Domain

## Background – Chemical formula (molecular formula)

A chemical formula lists the elements and the proportion of atoms per element using element symbols. More specifically the chemical formula written as a molecular formula includes the exact number of atoms per element using element symbols. This is done by listing the symbols and a subscript to indicate the number of atoms for a particular atom (with no subscript indicating one).

Example:

$$H_2O$$

Looking at our periodic table we can see the symbol H stands for Hydrogen, and O for oxygen. The subscript 2, indicates that there are two Hydrogen atoms and one Oxygen atom.

## Background – Chemical Reaction

A chemical reaction is a process where one or more compounds/elements known as the reactants are converted into one or more different compounds/elements known as the products. The reactants and products will have an equal amount of atoms of the same elements.

Example

When iron and sulphur combine they form iron sulphide

This can be represented using the following form (chemical equation)

$$Fe + S \rightarrow FeS$$

The compounds on the left-hand side of the arrow are the reactants and the products on the right. A + on the left-hand side can be thought of as “plus”, where as a + on the right-hand side can be thought of as “and”.

**For our program we will accept two reactants and two products.**

## Background – Balanced Chemical Equation

Given the following chemical equation

$$C_3H_8 + O_2 \rightarrow H_2O + CO_2$$

On the left hand side we can see there are
3 carbon (C) atoms
8 hydrogen (H) atoms
2 oxygen (O) atoms

On the right hand side we can see there are
1 carbon (C) atom
2 hydrogen (H) atoms
3 Oxygen (O) atoms

As these are not equal the chemical equation is NOT balanced.

To balance these we apply coefficients to the compounds. These coefficients multiply the number of atoms in the compounds.
Hence a balanced version of the above equation is

$$C_3H_8 + 5O_2 \rightarrow 4H_2O + 3CO_2$$

On the left hand side we can see there are
3 carbon (C) atoms
8 hydrogen (H) atoms
10 oxygen (O) atoms (5x2 =10)

On the right hand side we can see there are
3 carbon (C) atom
8 hydrogen (H) atoms (4x2 = 8)
10 Oxygen (O) atoms (4x1 + 3x2 = 10)

## Background – Input

As we cannot use subscript, the number of atoms, will be always to the right of the element symbol.

Example:

$4H_2O$ will be written as 4H2O
$2BBe_4CaH_2$ will be written as 2BBe4CaH2

Errors:

It is expected that all molecular formulas will be correctly formatted but may not be balanced

## Background - Limitations and Notes

**Error checking – User input**

The data entered by the user will be in the correct format of a chemical equation

**Number of compounds – User input**

Any of the products or reactants may be an empty String.

**User input**

The user input must accept any compound, i.e. **do not** hard code the compounds.

**Compounds and Elements**

An element will not appear twice in any single compound but may appear in both products and/or both reactants.

The number of atoms for each element in any single compound will be between 1 and 9, but the total combined for the reactants and/or products may be higher

## Task 1 - Solution length

The code should be around 150-250 lines of code. If your code is significantly larger, you may want to reconsider your approach.

# Task 1 – ChemicalReactionBalance.java

Follow the steps to complete the task using the steps and information above. (Note indentation indicates that the indented parts are contained or related to the previous step. Eg. All steps within an if statement are indented)
**The task is to read in two chemical reactants and two chemical products that make up a chemical equation and then determine if the chemical equation is balanced or not.**

## Task 1 – Stage 1

We are going to set up the main components of the program to start, validate it works, then go back and add more code.

1. The program prints to the screen the student number, student name and the assignment number. This is enclosed by double asterisks. Ensure you follow the below format

   ```
   *** 37777777 John Smith ***
   ```
   (Replace with your details)
2. The program prints the message shown below as to indicate the programs purpose

   ```
   Chemical Reaction Balance Check
   ```
3. Prompt the user to enter each of the 4 compounds as Strings and store each individually: reactant 1, reactant 2, product 1 and product 2.
4. Create 4 integer type variables for storing the coefficient of each compound. Initialize each to 1.
5. For each compound check (x4)
   a. If the first character is numeric (2 to 9)
      i. Set the appropriate coefficient to this value (You will need to convert the char to an appropriate integer) and remove the first character of the original String using the substring method.
6. Set up variables as follows:

   An appropriate variable that stores the length of the reactant 1 String
   An appropriate variable that stores the length of the product 1 String
   A String variable that stores reactant 1 and reactant 2 concatenated (reactants)
   A String variable that stores product 1 and product 2 concatenated (products)
   An appropriate variable to indicate that the chemical equation is balanced, initialized to true

   An integer variable for storing the number of atoms for the current element in the reactants (We will look at each element one at a time), initialize this to 1
   A String variable for storing the current element
7. Store the first char of the reactants into the current element variable
8. Iterate by 1 from 1 using variable named "i", to the reactants length +1 (exclusive)
   (We start at one as we have already stored the first (0 index) element)
   a. After completing Stage 1, in this loop complete **balance check** steps as set out below.
9. After iterating through the line
   a. Print the chemical equation out using '+' between the reactants, a '+' between products and "->" between the reactants and products as shown in the **output example (don't forget the coefficients)**
   b. If the equation is balanced then print "is balanced" following the chemical equation
   c. Else if the compound is invalid then print "is NOT balanced" following the chemical equation
10. End the program appropriately.

## Task 1 – Stage 2 - Balance Check

1. If the next character (index i character) in the input line is a lower-case letter (And i is not equal to the length of reactants String), then concatenate it to the current element. Eg. if input is “Zn”, then before iterating “Z” would be stored in the current element, then character at index 1 is lower case, hence current element would now store “Zn”.
2. Else if the next character is a number, (And i is not equal to the length of reactants String) then store the number of atoms of current element. Eg. If the input is “Zn3”, then initially “Z” would be stored as the current element and empty as the number of atoms, then on the next iteration “Zn” would be stored as the current element. Then on the next iteration 3 would be stored in the number of atoms (convert to an integer).
3. Else (You have the element symbol and the number of atoms (if blank, number of atoms is 1))
    a. If the current element is part of the reactant 1 (Remember we stored the length of the first reactant)
        i. Then check if reactant 2 contains the same element using the indexOf method on the reactants String and store the index if it does.
            1. You will then need to find the number of atoms of this second element (1 or 2 to 9), and store the total number of atoms for the element using the atoms of the element for reactant 1 and reactant 2 and the coefficients of both reactant 1 and reactant 2. (No more information is given here).
            2. Also, remove the second element from the reactants String so that it does not get evaluated twice
        ii. Else if the same element does not appear in the second element then calculate the total number of atoms for the element using the number of atoms for the element and the coefficient of reactant 1
    b. Else calculate the total number of atoms for the element using the number of atoms for the element and the coefficient of reactant 2
    c. Set up variables as follows:
        i. A string for storing the current product element **(hint: you will need to store a character in this String before entering the loop, like you did for the current reactant element)**
        ii. An integer for storing the number of atoms for the current product element initialized to 0
        iii. An integer for storing the total number of atoms for the element we are searching for in the products initialized to 0, referred to as balanceCheck below
    d. Iterate by 1 from 1 using variable named “j”, to the products length +1
        i. Check each product element against the current reactant element, if it is the same, then add to the balanceCheck the number of atoms for the element in the products (remember to multiply by the coefficient) (There will be a maximum of two matches). (No more information is given here).
    e. If the balanceCheck does not equals the number of atoms for the element in the reactants, then set the variable to indicate that the chemical equation is balanced to false And print out the element symbol, a tab, then the number of atoms in the reactants, then the String “ != “ and then the number of atoms in the products. Refer to output examples
    f. If i is not equal to reactants length
        i. Then assign number of atoms of current element to 1
        ii. And assign the current element to the empty string
        iii. Store the i’th character of the reactants into the current element variable

## Task 1 - Example outputs

```
*** 37777777 John Smith ***
Chemical Reaction Balance Check
Enter reactant1 1: C3H8
Enter reactant1 2: O2
Enter Product 1: H2O
Enter Product 2: CO2

C       3 != 1
H       8 != 2
O       2 != 3

C3H8 + O2 -> H2O + CO2
is NOT balanced
```

```
*** 37777777 John Smith ***
Chemical Reaction Balance Check
Enter reactant1 1: C3H8
Enter reactant1 2: 5O2
Enter Product 1: 4H2O
Enter Product 2: 3CO2


C3H8 + 5O2 -> 4H2O + 3CO2
is balanced
```

```
*** 37777777 John Smith ***
Chemical Reaction Balance Check
Enter reactant1 1: PCl5
Enter reactant1 2: 4H2O
Enter Product 1: H3PO4
Enter Product 2: 5HCl


PCl5 + 4H2O -> H3PO4 + 5HCl
is balanced
```

```
*** 37777777 John Smith ***
Chemical Reaction Balance Check
Enter reactant1 1: 2FeCl3
Enter reactant1 2: MgO
Enter Product 1: Fe2O3
Enter Product 2: MgCl2

Cl      6 != 2
O       1 != 3

2FeCl3 + MgO -> Fe2O3 + MgCl2
is NOT balanced
```

## Transferring files between Windows and Unix

Be very careful transferring files from Windows to Unix. If you do transfer a file from Windows to Unix open the file, in Unix, using vi.
For example, if you transferred a file named **`b.txt`** from Windows to Unix

open the file in Unix with the command

vi –b b.txt

you will (might) see a lot of **`^M`**'s at the end of each line.

These MUST be removed using the command shown below or else your input file will have too many newline characters and will not translate properly. That is, your code will not correctly read the input file for Tasks 1 and 2.

Your code will work on Windows but NOT on Unix.

Still in vi, in command mode (press the Esc key first) do the following

:%s/ctrl-v ctrl-m//g

**`ctrl-v ctrl-m`** means hold down the control key and with the control key down press v then press m.

## Final notes

There will be consultation sessions for the assignment, the times will be posted on LMS, if you have problems come to consultation.

Do a little bit every night; before you know it, you will be finished. The assignment is marked with running code, so you are better to have 1 or 2 Tasks completed that actually compile and run, rather than a whole lot of code that doesn't compile.

The execution test is done on **latcs8** so please make sure that your code runs on **latcs8** before you submit.