1. *Is this individual project or group work?*

   This is individual course work (ICW). You have to finish it by yourself.

2. *Do I have to write my code in Java?*

   Yes.

3. *What is the percentage of this project in the overall assessment?*

   This project counts 60% of the overall assessment.

4. *Do I need to complete a sub-question first before going to the next one?*

   Not necessarily.

5. *Do I need to comment my code?*

   As this project requires a few hundred lines of codes, the readability of your program is critical and will be considered in marking: this includes proper comments, naming conventions, indentations, cleanness, code refactoring….

6. *What is the deadline for submitting this project?*

   13 December 2019 (11:59PM)

7. *How do I submit the project?/What should I submit?*

   See document **CSC1025_ICW.pdf**: **Submitting your work** and **Upload Instructions**

8. *Am I asked to write a compiler?*

   No. You are asked to implement some simple functionalities for a compiler.

9. *Am I asked to write my code in weeJava?*

   No. weeJava code is the input that your (Java) program will analyse.

10. *Can I discuss with other people (lecturer/TA/colleagues/peer mentors) the course work?*

    Yes. Healthy discussion is always encouraged.

11. *Can I use regex in my program?*

    No.

12. *What is a new line?*

    It is a character: `'\n' or '\r'.` Depending on the operating system, sometimes it can also be "\r\n".

13. *"In Q4 part D, we are asked to create 10 examples of compile time errors for our program to check. I was wondering if we could simply edit the Q2Example.java file we made earlier 10 times, each with different errors,…"*

    Yes, you can.

14. *As for the String token names, should we include the "" or just have the text inside for example "abcdefg" or just abcdefg*

    "abcdefg" is expected.

15. *"Should Q5 output all previous task such as Q2, Q4 and then Q5 if there is no errors in Q4 output or just output the expected result for Q5?"*

    Just output the expected result for Q5.

16. *"In Q5, are we meant to assume that the weeJava program has no errors (For example: The main method definitely exists, all identifiers have valid names , etc.)"*

    Yes.

17. *"For question 2 of the coursework, the question states that I should design the Q2Example.java file myself. It says that this should contain all the token types in wee java. I was wondering how big the example file needs to be because I was starting to create it and to include all the token types it will require most of the topics we have covered."*

Yes. This is what I want; the example program will cover most what we learned: variable declaration, if selection, switch case, while/for loop, method declaration. But there is no need to spend too much time on designing Q2Example.java; you can even copy/paste codes from lectures and practicals.

18. *Some questions about the MyTokenizer.java.*

    a) *"It seems to confuse students at it uses techniques we didn't cover in the module as well as there is no comments to explain what the specific method is or what it does."*

    Students can come to peer mentors or me for clarification of its use. More importantly, MyTokenizer.java is totally **optional**.

    b) *"Some of the students also thought that is how it should be done and wanted to re-do their project using that class, I advised that they **don't need to re-do** the project and it is **just there for help if they are still struggling with Q2**."*

    Exactly. Many of you have already managed to output most token types for Q2, in which case there is no need to use MyTokenizer.java

    c) *"Also when you open the file there is a S2 variable which breaks the program as it is undefined and there is no indication to what it is meant to be."*

    Fixed now.

    d) *"From my point of view the tokenizer.java seems confusing at first especially with how it is laid out with nested if statements bunched up close to each other as well as the while loops, I think that the first help example that you put out for Q2 is much easier to understand"*

    I agree.

    e) *"couple of students were worried that they are going to be marked down because they created a solution that resembles the one that you provided."*

    Nobody will be marked down because of MyTokenizer.java
    Don't worry if your solution resembles parts of MyTokenizer. That is normal and I even encourage you to see how MyTokenizer.java is written.

19. *Some questions about the Q2.*

    a) *To detect STRING: "I know that for the string values if a quotation mark is found, i need to iterate it until another quotation mark is found, adding the characters as it goes along. I'm just not sure how to properly implement this"*

    Starting from the first quotation mark, copy and paste the code from Line 79 to Line 92 https://canvas.qub.ac.uk/courses/8455/files/495693 you need to change

the test condition in Line 84 to !(ch== "") You also need to extend the else part: 1) add the second quotation mark into word, and 2) and increase index.

b) *To detect INTEGER and DOUBLE*

Starting from the first digit, it is the same: copy and paste the code from Line 79 to Line 92 https://canvas.qub.ac.uk/courses/8455/files/495693 you only need to change the test condition in Line 84 to isDigit(ch)

Then, to check if the number is an int or double, traverse the word and check if there is the dot '.' It is a match program (see Lecture 18).

20. *In question 4d of the ICW do we import all 10 example files at the same time?.*

No.

One possible format can be the following.

```
public static void main(String[] args) {
	String prog = readFile("src/Q2Example1.java");
	// String prog = readFile("src/Q2Example2.java");
	// String prog = readFile("src/Q2Example3.java");
	// String prog = readFile("src/Q2Example4.java");
	// String prog = readFile("src/Q2Example5.java");
	// String prog = readFile("src/Q2Example6.java");
	// String prog = readFile("src/Q2Example7.java");
	scan(prog);
}
```