**Tutorial 2 Question 6 – KWIC**

Matriculation numbers:

* A0193543B
* A0183208E
* A0193145E

**Introduction:**

While planning and writing this KWIC implementation it was immediately apparent that modularisation and coupling will be crucial if we wanted our program to be both intuitive to create and understand. We attempted to follow Parnas’ advice and we focused on abstracting the problems into five main modules: Input, Characters, Shifter, Sorter and Output. These worked as our guide and interface during our design process. To achieve this, we focused on the ADT solution as stuck close to the solution presented in the Garlan, Shaw paper. In our program we take Master Control to just be our Main class which controls the entire execution of the program. It’s important that our implementation is fairly barebones, used mainly to show off the ADT implementation. The input method is a simple text file with a matching KWIC output.

**Architecture / Methodology:**

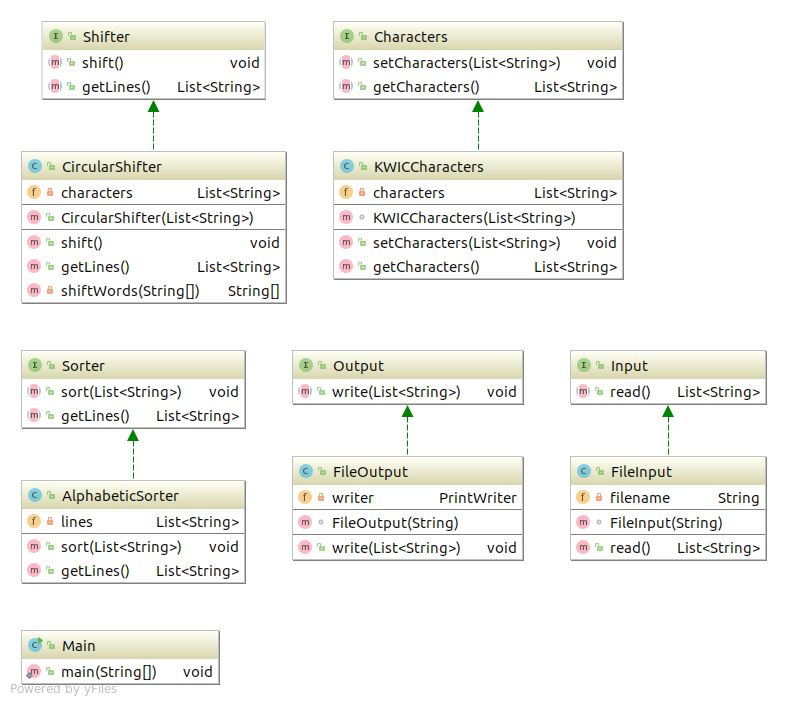
Changes:

Our solution to KWIC follows closely to the ADT solution originating from the previously named papers. What could be considered a slight deviance from the original solution is that Master Control (MC) appears to have more control over Alphabetic Shifts ( called Sorter in our implementation ). This is because MC calls Circular Shift and Alphabetic Shift directly, rather than abstracting this part of the implementation away.

Another change is that Circular Shift and Alphabetic Shift only take the entire line as arguments. This is due to the simple nature of our program. ‘Word’ is taken from the lines themselves, further abstracting our implementation:

*Shifter shifter = new CircularShifter(characters.getCharacters());*

Implementation:



This UML diagram represents our domain classes relations. Our program implemented according to P2I principle: all main functions depend only on the interfaces, not implementations.

**Input / Output:**

The form of input and output in our KWIC implementation is quite straightforward. The argument in the Input constructor in Main.java is the file read in. Likewise, the argument in Output will be the file with KWIC. Input should have the path specified or be located in the source folder.

Example: input.txt

This is a test input

Output: output.txt

**[a, test, input, This, is]**

**[input., This, is, a, test]**

**[is, a, test, input, This]**

**[test, input, This, is, a]**

**[This, is, a, test, input]**