**Assignment 1: KWIC-KWAC-KWOC**

Code Repository URL: <https://github.com/anand-sundaram/KWIC>

|  |  |  |
| --- | --- | --- |
| Name | Rupali Roy Choudhury | Anand Sundaram |
| Matriculation Number | A0102796B | A0118897J |

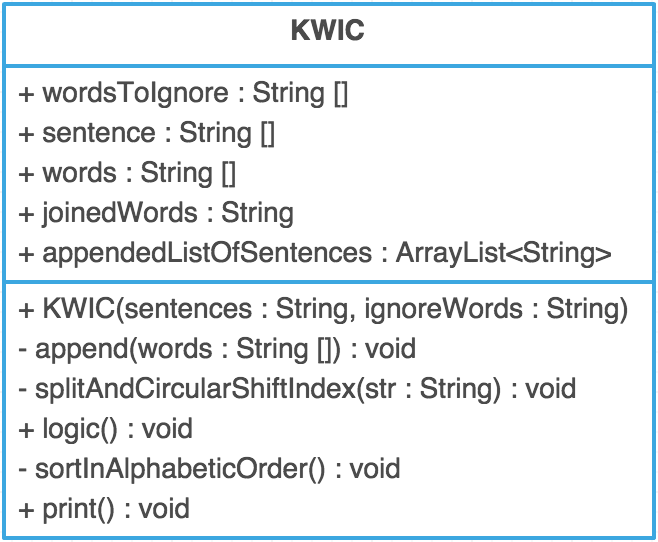
**1. Introduction**

KWIC is a system that presents a search mechanism for information in a long list of lines. For our assignments, we are required to implement functional as well as non-functional requirements. We have successfully implemented the basic functionalities of KWIC using mainly two architectures – 1) Shared Repository (done by Anand) 2) Data Abstractions and Object-Oriented Organisation (done by Rupali).

NOTE: Please use JavaSE-1.8 for JRE System Library

**2. Design**

Data Abstractions and Object-Oriented Organisation



KWIC class consists of mainly 6 functions, as can be seen from the class diagram. The user initially provides the sentence, words to ignore and the type of architecture. An object of KWIC is created. This object first calls the logic method, which in turn calls other methods as explained below. The purpose of the following methods is described below:

* KWIC(String sentences, String ignoreWords) – This constructor takes in the string values, which are provided by the user and separates them based on “,” (comma) or “ ,”(space + comma). These separated strings are stored inside an array.
* public void logic() – Takes in the array of string and sends one index at a time to the function splitAndCircularShiftIndex.
* private void splitAndCircularShiftIndex(String str) - It takes the individual index of the sentence array and further splits it to words and stores in an array called words. This array is then useful for performing the circular shift. After the circular shift is done once, it calls the append function.
* private void append(String words[]) – The aim of this function is to join the word array and append if the first word is not in the list of words to be ignored. The capitalization of the first word is performed here. After that, sortInAlphabeticOrder function is invoked
* private void sortInAlphabeticOrder() – The appended sentences are then ordered in alphabetical order.
* public void print() – displays the final outcome as mentioned under the basic requirements.

**3. Limitation & Benefits of Selected Design**

Benefits of Data Abstractions and Object-Oriented Organisation

* Software maintenance - Programming using this architecture is not disposable. Maintaining and improving on an object-oriented program is much easier than a non-object oriented program.
* Design benefits - Large programs are very difficult to write. Object Oriented Programs force designers to go through an extensive planning phase, which makes for better designs with fewer flaws.
* Encapsulation - Once an Object is created, knowledge of its implementation is not necessary for its use. Objects have the ability to hide certain parts of themselves from programmers. This prevents programmers from tampering with values they shouldn’t. Additionally, the object controls how one interacts with it, preventing other kinds of errors.

Limitations of Data Abstractions and Object-Oriented Organisation

* Since the functions are invoked by another functions, a small change in one of the functions can mean that these changes will have to be taken care by other functions too.