Assignment 1: KWIC CS3219 SEM1 2016/17

Code Repository URL: https://github.com/hanchiang/CS-3219/tree/master/KWIC

Student Name	Yap Han Chiang	Yang Jung Kai
Matriculation Number	A0125168E	A0121593J

1. Introduction

KWIC is a way of searching for words in a sentence. Given a list of lines and a list of words to ignore, KWIC system should generate a KWIC index of the input lines. In a KWIC-index, a line is listed once for each keyword that occurs in the line. The keyword cannot be in words to ignore . Also, KWIC-index is alphabetized by keyword.

2. Requirements

The system must take in a list of sentences, and the words to ignore, and cycle through each word and populate the lines in alphabetical order. It must be able to handle a decent number of lines as input and produce a reasonably fast response time.

2.1 Functional

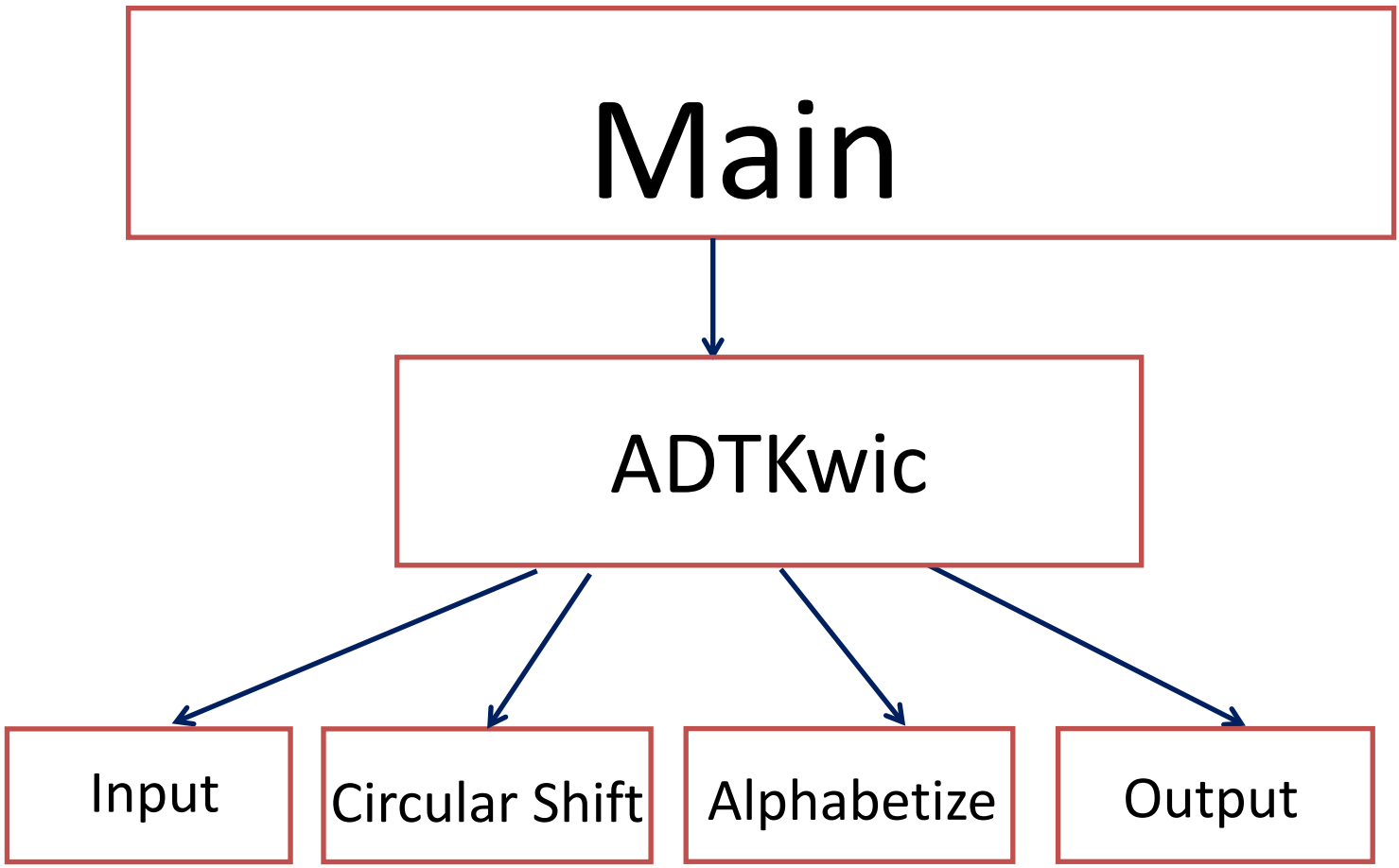
- User need to provide movie titles and words to be ignored
- The program need to circular shift all the words in the list and output the first 20 results to console and the rest to output.txt

2.2 Non-functional

- The program should be able to handle test cases up to 1000 lines and generate the result in a reasonable time frame

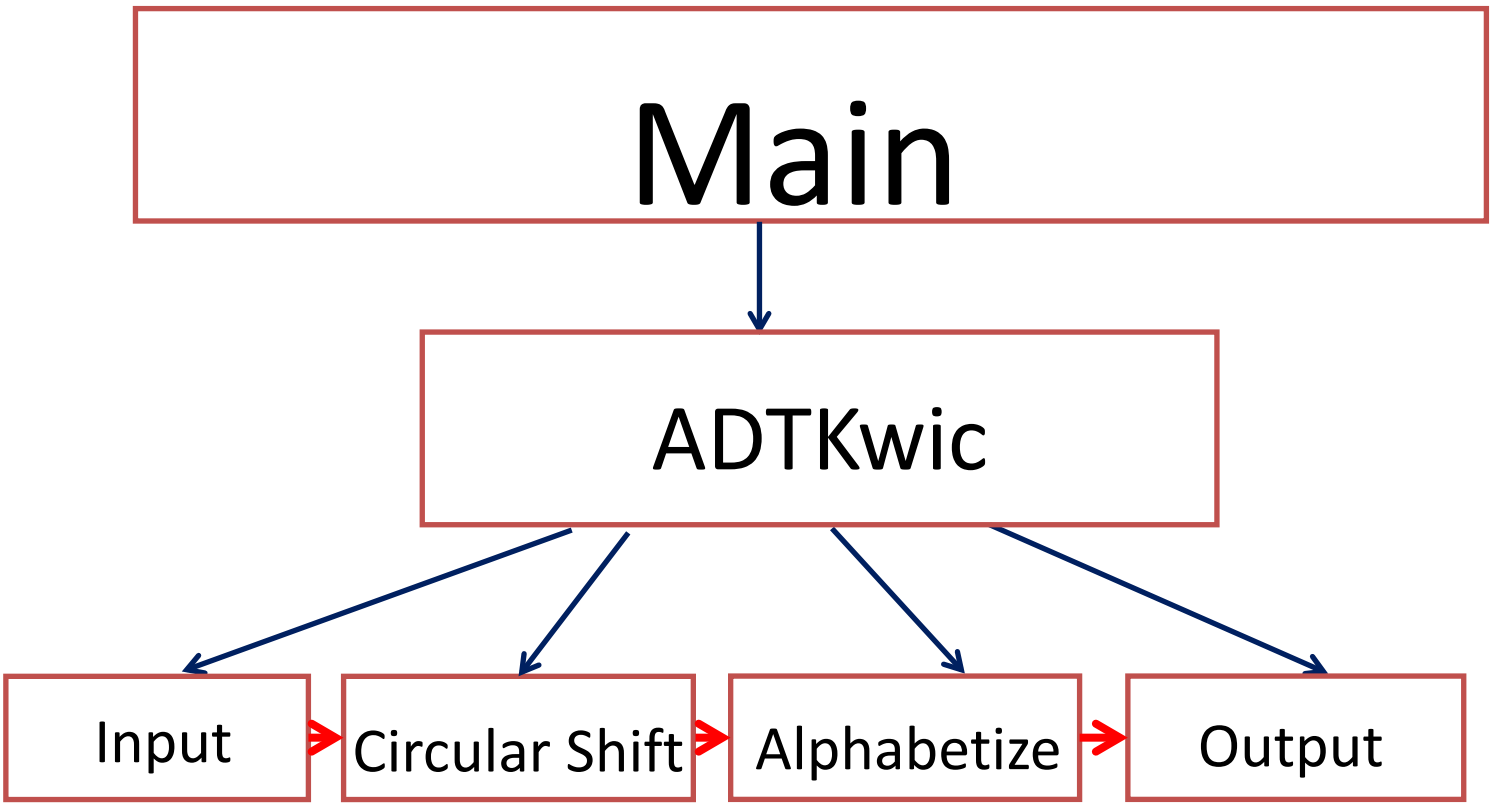
3. Architectural Designs

Solution 1:



In this design, the main program calls the 4 functions, and data is shared between them.

Solution 2:



In this design, the 4 functions are encapsulated as their own abstract data type. The red arrow represents the flow of data from each type to the other.

4. Limitation & Benefits of Selected Designs

Solution 1

Benefits: The code is shorter, and is ideal for small programs, as we can see all the data attributes and functions that are used in this program.

Limitation: As the data is shared between the functions, if the program becomes bigger, errors will be harder to trace as the same data attribute can be use and modified in several places.

Solution 2

Benefits: Each module is encapsulated in its own data type, which results in a clear and logical work flow. The code is easier to understand when it is shared with other people. Each

modules do not affect one another, which results in a less tedious task of debugging.

Limitation: Adding new functions to modules requires modification to the modules, which can result in a more complicated code. Using ADTs results in a longer code than the traditional way.

5. User Guide

Shared Data Solution(SDS) & Abstract Data Type (ADT)

1. Run the program on either EcIipse IDE or command prompt.
2. To run on command prompt, click export and export as runnable jar and name it as KWIC.jar and choose the destination to be your local directory (e.g. C:\Users\USER\Deskstop).
3. After you run the program, message that prompt the user to choose either shared data KWIC or ADT KWIC will show up. Enter the corresponding index number to choose the type of KWIC to run.
4. Prompt will then show up to allow user to choose either manual input or input through file reading.
5. If user choose to enter manually, user is able to enter movie titles by typing movie titles and press enter. To move on to enter ignore word, user need to press enter twice.
6. Prompt will then show up to guide user to enter ignore words. the way of entering the input will be similar to the way of entering input for the movie titles. To skip the input for ignore word and display the result, user need to press enter twice.
7. If user choose to input through read file, user has to make sure that the file is saved in the same directory as where the program is stored. Prompt will be shown for user to enter the name of the file which contains movie titles and name of file which contain the ignore words.
8. If inputs for both movie titles and ignore words are entered without any errors, first 20 items in alphabetical order for result of KWIC will be shown and the result will then be saved in output.txt in the directory containing the program. User will be able to view the result by opening output.txt.