MÄLARDALENS HÖGSKOLA
ESKILSTUNA VÄSTERÅS

JANUARY, 2018

LABORATION 1: SOFTWARE DESIGN

DVA 422: SOFTWARE ENGINEERING 3

EMINA SMAJLOVIĆ & AMINA KREKIĆ
esc17001, akc17003

The Key Word in Context index system

The Key Word in Context (KWIC) index system is described in [Parnas72] and [GS93] as follows:
The KWIC index system accepts an ordered set of lines, each line is an ordered set of words, and each word is an ordered set of characters. Any line may be “circularly shifted” by repeatedly removing the first word and appending it at the end of the line. The KWIC index system outputs a listing of all circular shifts of all lines in alphabetical order.

Your task is to create an object-oriented design for the KWIC system and a prototype implementation that realizes your design.

Requirements and assumptions for the design

Use the principle of "information hiding" and include (at least) classes that encapsulate the following design decisions:

1. The input media and data format
2. The format for storing the lines in the input
3. The algorithm for computing the circularly shifted lines and the format for storing the result
4. The algorithm for sorting the circularly shifted lines and the format for storing the result
5. The output media and data format

In other words, changing any of these decisions should only require updating the corresponding class.
Document your design in a file (Word or PDF) containing UML diagrams. Create a model with (at least) a class diagram showing classes, public methods, and associations. In the internal design of your classes, efficient use of execution time and memory is not a primary concern. It is permitted to prioritize efficient use of your own working time.

Requirements and assumptions for the prototype implementation

Implement your prototype on the .NET platform using Visual Studio 2010 or 2013. (For example, you can use the C# language). Your program must be able to read the input data from a text file and write the output to another text file. Make a console application that accepts the names of the input and output files as arguments. An example project that illustrates how to do this in C# is attached. It is only necessary to handle input files consisting of lines of space-separated words correctly. It is not necessary to invent solutions for handling punctuation characters, multiple spaces, empty lines, etc.

Reporting the assignment

Create a subfolder called "Lab 1" in your Subversion repository containing the following:

- A file (Word or PDF) containing UML diagrams
- Your complete Visual Studio project

Hint: you may use the Clean Project function in Visual Studio to remove binary files from the project before adding it to your repository.

Solution

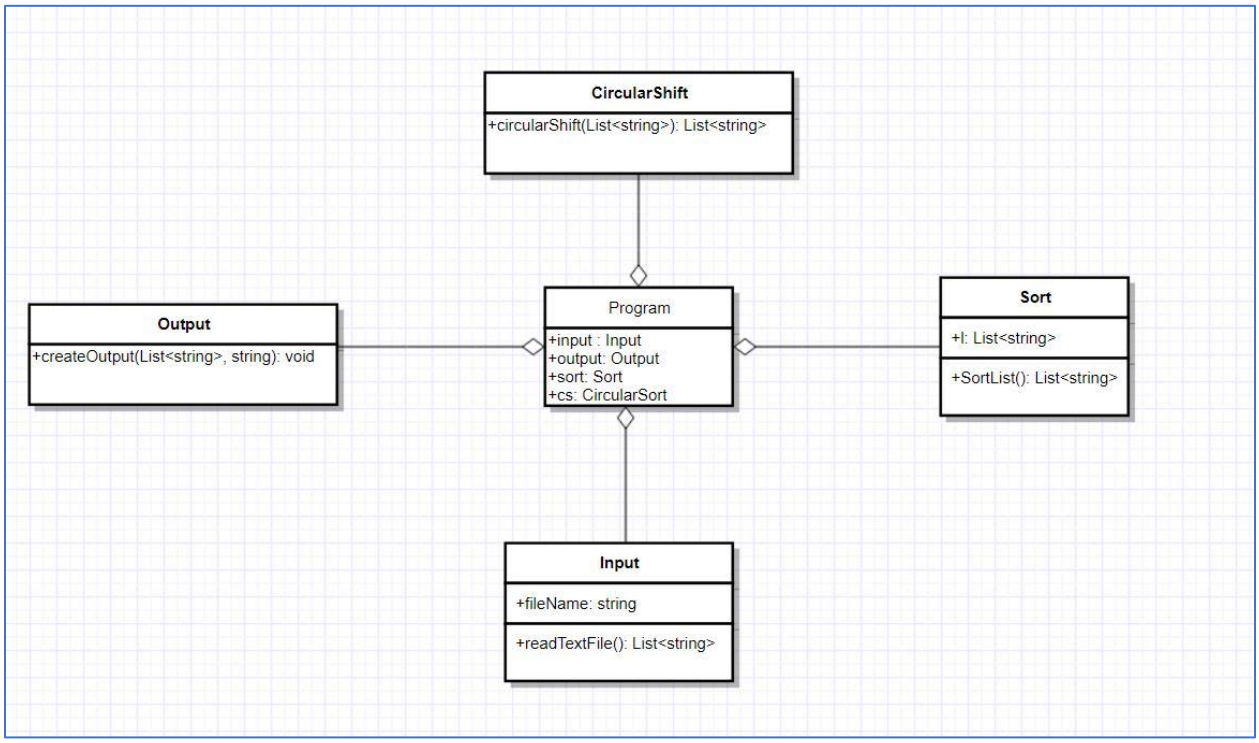

Figure 1 – UML diagram