C01: Assignment 3
Due date: 7th August @ 11:59pm

1 What is expected?

On many occasions, when you go and work in industry, you may be given an unfortunate task of taking some previously poorly written code, and to refactor it so that it is better designed and has the flexibility to easily accommodate new requirements in the future. What you are expected to do in your final assignment, is to take some poorly written code, and refactor it as per the techniques that we learnt in the last few lectures of CSCC01. This poorly written code and design that you will soon inherit was written as per the specs provided in `C01aux.pdf`, this file is available on Blackboard for you to read and understand what the the original requirements were. You will notice, that the code that you inherit does meet the customer requirements i.e, it compiles and runs correctly on the provided `input1.txt` and `input2.txt`, however, it is poorly designed.

You will want to consider the following items, before starting to refactor your code.

- Single responsibility principle
- Liskov substitution principle
- Interfaces/Abstract classes
- Design patterns that we learnt in this term
- Unit testing
- Exceptions and Generics
- Limit methods to 30 lines (this does not include comments).

I do not expect you to make use of every single design pattern; however, some of the design patterns may be applicable, and you must use them. You will, however, are expected to make use of the `single responsibility principle` and use of `interfaces` and perform extensive unit testing (mock objects including). You may want to explore the possibility of incorporating the `iterator` design pattern, such that it allows the traversal of the `user user` matrix.

You are free to create as many new classes as you see fit, however, make sure that these classes reside under their appropriate package names, and that the test code resides under the `test` folder. If you create any inner classes, you are not responsible for testing these inner classes. You can use `Mockito` and `JUnit` to write your unit test.

You are also expected to submit a report `A3.pdf`, that must reside in the assignment three folder, and this report must document what are the design patterns that you made use of, and

wherein the code would we find them? And what refactoring did you carry out to clean up the previously written code?

2 Few more items to keep in mind

- On your local computer, `cd` into the directory where you first performed `git clone` for Assignment1. This is the directory that contains the `.git` file
- Now, type the following: Comments are indicated by //

```
git remote add instructor https://mcs2@bitbucket.org/mcs2/cscc01summer2018master.git
git fetch instructor

//make sure you are on master branch at this point.
git merge instructor/master

//after the merge, you should have assignment3 folder on your local master branch.
//This folder will also contain two java files CfilteringDriver.java and
//Cfiltering.java
```

- The above git commands create an alias for `instructor`, that you have read permissions (but not write permissions). However, when you commit, make sure to commit on `origin` and not `instructor`.

1. After running the above commands, you can compile your textttassignment3 by running `mvn install` and then finally executing your assignment3, by running `mvn -e exec:java -Dexec.mainClass=csc.summer2018.cscc01.CfilteringDriver`.
2. When prompted on the terminal, you must enter either `input1.txt` or `input2.txt`.
3. The code that is provided to you will compile and run fine with the above two test files.