Aim: In this assignment, you will implement Burrows-Wheeler Transformation (BWT) and Ferragina-Manzini (FM) index. Your implementation will have two parts:

- Given a long text from the DNA alphabet ($\Sigma = \{A, C, G, T\}$) it will generate and save the BWT string, the FM-index, and the suffix array. The input file format will be FASTA as in the Homework 1. There will be two output index files. First, it will generate a file with extension ".bwt", which will be a text file that contains the BWT string. The second file with extension ".fm" will include the FM-index tables and the suffix array. The format will be your own design.You may assume that the input text is at most 1 million characters-long.
- Given the BWT file (.bwt) and a pattern to search in FASTA format, find all exact match locations of the pattern in the text.

Sample run:
Part 1 - Index generation:

```
bwtfm --index mytext.fa
```

This should generate two files with the filenames `mytext.fa.bwt` and `mytext.fa.fm`.
Part 2 - Search:

```
bwtfm --search mytext.fa --pattern query.fa
```

This should automatically detect `mytext.fa.bwt` and `mytext.fa.fm`. Example output:

```
Pattern P found in T 4 times at positions:
    pos 1: 23655
    pos 2: 243141
    pos 3: 141113
    pos 4: 327128
Search completed in 4.23 seconds.
```

Note: read the text and pattern sequence names directly from the input FASTA files. Do not just output "P" and "T".

Notes:

- You must write your code yourself. Sufficient evidence of plagiarism will be treated the same as for plagiarism or cheating.
- Non-compiling submissions will not be evaluated.
- Your code must be complete. You may divide your code into multiple files, but all should be compiled into a **single binary** using the Makefile. If scripting languages are used, a single wrapper script should be provided.
- Do not submit the program binary. You must submit the following items:
  - All of the source files.
  - A script to compile the source code and produce the binary (Makefile), if required.
  - A README.txt file that describes how the compilation progress works, if required.
- **Submit your answers through the Moodle page. DO NOT EMAIL.**
- Zip your files and send them in only one zipped file. File name format='surname_name_hw2.zip'
- C / C++, Python, Java will be used as programming language. STL is allowed.
- All submissions must be made by 23:59, October 30, 2019. Late submissions will lose 20 points for each additional day. Three or more days of delay will result in a zero grade.
- The overall fastest implementation wins. **Bonus** will be given for the code that runs the fastest for both index generation and search.