COMP 6651: Project

Fall 2022

Submission through Moodle is due by December 5th at 23:55

In this project you will write plagiarism detection tool in one of the following programming languages: C++, Java, or Python. It is important that you follow the guidelines on the input/output and structure of your submission **exactly**, since submissions that do not adhere to these guidelines receive a grade of 0.

Your submission will be a "student_id.zip" file with a folder named with your student id and containing the following files only:

```
<student_id>_detector.[cpp, java, py]
<student_id>_README.txt
Makefile
```

Thus, if the student id is 12345678 and the project is done in C++ then the 12345678.zip file should have a folder named 12345678, and inside the folder there should be the following files only (and no other files):

```
12345678_detector.cpp
12345678_README.txt
Makefile
```

The detector tool accepts two command line arguments: the first argument is the path to FILE1, and the second argument is the path to FILE2. The input files FILE1 and FILE2 are plaintext files. The tool should analyze these files in reasonable time (not more than 5 seconds, and file sizes will be no more than 500 kilobytes) and print 1 if it detects plagiarism and print 0 otherwise. The printing is done to standard input, and no other output should be produced by the program.

Makefile must support the rule "run" which is executed as follows:

```
make FILE1=<path_to_file1> FILE2=<path_to_file2> run
```

This command should compile (if necessary) and execute your plagiarism detector tool with inputs FILE1 and FILE2.

Your program must use only built-in libraries that come with the language and must not rely on external libraries.

Your README file should be a plaintext file that has between 400 and 500 words describing how your algorithm detects plagiarism.

Your submissions will be tested and executed on the following server:

```
computation.encs.concordia.ca
```

You need to make sure that your program compiles and runs on this server as is. That is, I should be able to upload your folder to the above server, upload test dataset, and execute "make FILE1 FILE2 run" for every pair of files in the dataset. If your program does not compile, or if "make" command returns a mistake, or the program takes too long to terminate, or the program prints something other than 0 and

1 to standard output then the grading process is aborted and grade 0 is assigned. Make sure to test your program on the above server extensively and submit it only when you are sure it works without any issues.

On Moodle you can find a file called “sample_data_and_submission.zip” which contains some datasets with pairs of non-plagiarised, as well as pairs of plagiarised texts. I also included a short Makefile, a stub of a sample submission in C++, and a dummy README.txt which indicates approximate length of the description you should write about your plagiarism detector.

Some suggestions:

1. You are free to implement many different approaches to devise the best plagiarism detection program. Some initial ideas include such algorithms covered in this course as Edit Distance and LCS. You can also try to extend these algorithms to weighted versions, where weights of certain characters are smaller than others in Edit Distance. You may also play around with preprocessing of the input, such as removing whitespace, or converting the input text to a bag of words representation and computing some similarity metric between two bags of words. You may also investigate other string processing algorithms such as Rabin-Karp algorithm and its variations. Or, perhaps, you discover that a combination of many simple techniques gives the best results. Be creative!

2. Your program (as can be seen from sample input) can be executed both on regular text as well as on source code of programs in various languages. Perhaps, you can make your plagiarism detector first detect whether the input text is source code or regular text and then run different plagiarism detection algorithms: one for source code plagiarism and one for regular text plagiarism.

3. Your program will be tested on a larger dataset than what you are given and the grade will be proportional to the quality of your detection algorithm. Thus, you may wish to find or generate more samples of plagiarised and non-plagiarised text to check the performance of your algorithm.

4. Your programs will also be tested on submissions of other students (including your own) in this course. Thus, if your algorithm detects plagiarism on this plagiarism detection project then it may be used as a basis for starting academic violation hearing (upon closer manual examination). In particular, your plagiarism detection tool may detect that your own code is plagiarised with another submission.