**CS 5313 Formal Language Theory – fall 2024**

Programming assignment #1a

4 points

Due date and time: September 3, 2024, 12:00 PM

Programming language restriction: Must program in C, C++, or Java; otherwise the assignment will receive 0 credit.

This is part of programming assignment focusing on FSM and applications. Other assignments will use this one. For this assignment you are required to design, implement, and test a DFSM simulator in the ‘csx’ environment as described below:

**DFSM Simulator**: Design and implement a program to simulate a DFSM (a DFSM interpreter). This program must take two arguments and output ‘yes/no’. The first argument is the DFSM (the name of a file containing the specification of a DFSM) and the second argument is a string (the name of the file containing the input string to the DFSM). Both arguments must be command line arguments. The output of the program is ‘yes’ if the DFSM accepts the string and ‘no’ otherwise. Result should be printed on the terminal screen.

The DFSM specification is structured as follows (first argument is the file name):

The input file should have three sections – the input alphabet, the transition table, and the accepting states. Use an empty line to separate the sections; that should be the only way empty lines used. First section contains the alphabet – symbols separated by one or more spaces. Each alphabet is a single character. The order of the alphabet listing will be the same as the order of the columns of the transition table. The second section provides the transition function represented as a table, each line of the section beginning the first line of the section represents a row of the transition table. First row represents the start state. Elements of the row are next states separated by one or more spaces. The columns of the table are indexed by the ordering of the alphabet as it appears in the first section. States are denoted by integers starting from 1, which the start state. The rows of the transition table constitute the lines in the second section. The third section lists the accepting states separated by spaces. If the third section is empty, then there are no accepting states.

Input sting should be in the input file as a stream of characters. The second argument is the file name containing the input string.

Requirements:

1. You need to upload the source code and typescript file onto the course directory in csx. 50% penalty will be applied if typescript file is missing.
2. Your program must compile and run correctly in ‘csx’ (penalty 100%) according to specifications.
3. The programs must be well documented (documentation must include your name, course, and assignment number) (penalty 10%)
4. The program must take two strings (file names) as command line arguments. The first argument is the name of the file that contains the DFSM specification. The second argument is file name containing string (penalty 10% applied to otherwise correct program).
5. Must not use any function other than the primitive ones of the programming language (penalty 90%).
6. Your program must check for boundary conditions and output appropriate error messages (penalty 20 points).
7. Check syllabus for plagiarism related penalties. (Full penalty will apply).

You need to submit your program using the handin program.

Example program submission command (from your account in csx) is “handin cs5313 program1a prog1.c”. prog1.c is the file being submitted.