CSci 3501 Lab 8
32 points (See canvas for due date)
Work in pairs

- All lab submissions should be done by canvas. Please be sure to include your group members.
- When working on the lab, please comment your work so that it is clear what the contributions from each person are.
- At the end of the lab each group should submit the results of their in-class work. Please indicate if this is your final submission. Don't forget to answer all the questions below.
- If your submission at the end of the lab time was not final, please submit a final copy before the due time.

Lab overview and goals

The goal of the lab is to practice with JFLAP (a tool for experimenting with finite automata and other computability topics) and to design and test finite automata (DFAs and NFAs).

Using JFLAP and naming your files

Please save your automata files as .jff files and your data as .txt files. Files names must be as follows: names of those in the group followed by question name, e.g. SmithAdams3.jff (where 3 refers to the question number). This will help me in running test data. To load test data from a file, go to Input → Multiple Runs → Load Inputs

When adding multiple transitions between the same two states, add them one by one. Typing ”0, 1” in a label for an arrow will give you a wrong result since the automaton will try to match this input exactly, including the comma.
When writing DFA, check that every state has a transition on every symbol. JFLAP does not check it.
Use Convert → Combine Automata to copy one automaton into a file for

another one.
Do not use "convert to DFA" or "minimize DFA" options.
Consult the JFLAP tutorial as needed.

Lab task 1: Some NFA practice (12 points: 4 points for each problem)

Unless specified otherwise, the alphabet is the set of 0 and 1.

- Construct an NFA for the language of all strings with at least two 0s or at least two 1s.
- Construct an NFA for the language of all strings that have at least two occurrences of a pattern 01.
- Construct an automaton (DFA or NFA) for a language of all strings of length at least 3 that have a pattern 01.

Lab task 2: Some more FA (12 points: 4 points for each problem)

Unless specified otherwise, the alphabet is the set of 0 and 1.

- Construct an automaton (DFA or NFA) for a concatenation of the following languages: a language of all strings of an odd length and a language of strings that can be represented as a non-empty sequence of 1s followed by a non-empty sequence of 0s. Test the automaton carefully.
- Construct an automaton (DFA or NFA) for A* (* stands for the "star" regular operation), where A is the language of strings of length at least two with no occurrences of the pattern 00. Think carefully of what the language is. Use the * construction for NFA. Test the automaton thoroughly.
- Use the "convert to DFA" function of JFLAP to convert an NFA for the language of all strings with at least two 0s to a DFA.

Lab task 3: Convert a regular expression to an NFA (8 points)
Use the tutorial Regular Expressions and Converting to an NFA to convert

the regular expressions given below to NFAs. Note that JFLAP uses + as a symbol for union and * for the star operation. Also make sure to set parentheses correctly in your expression.

JFLAP gives you hints on the steps and provides a button "(D)e-expressionify Transition" to break down regular expressions into smaller subexpressions. You need to add the empty transitions to complete each step.

As you are converting a regular expression to an NFA, **write down the intermediate expressions** as a note on the JFLAP screen: use the selection tool (the one that you use to mark initial and final states), right-click anywhere on the JFLAP screen, and choose "add note". For instance, if you are breaking down an expression 00+11 into 00 and 11, write 00+11 → 00, 11.

Below are two regular expressions to convert:

- $((0 + 1)0)^*$
- $(00)^* + (10)^*$

What to submit
Submit your files to canvas.

- Make certain your group members and **contributions** of each member are clear.
- Make sure to submit your DFA files (as .jff)
- and your input data (as .txt) (test files are **required**).

Make sure to follow the naming requirements! Make it clear which data refers to which automaton.