Lab 2: Automating NFA to DFA Conversion

Due: March 21, 2025 EOD

Groups of 2

Objective

Students will work in groups of two to write a C++ or Python program that automates the conversion of an NFA to a DFA. The program should:

1. Read input from a text file containing an NFA transition table.
2. Convert the NFA to an equivalent DFA using the subset construction algorithm.
3. Write the resulting DFA to an output text file, ensuring the format matches the input format for automated checking.
4. Include the student IDs in the output file, since the input file will not have student details.
5. Process multiple input files at once, provided as command-line arguments.

Program Requirements

- The program should accept one or more input files as command-line arguments.
- Each input file contains an NFA transition table but does not include student details.
- The output DFA must be written to a single text file in the same folder.
- The output format must match the input format, with the addition of student IDs for tracking.
- Each group must include both student IDs at the top of the output file.

Input File Format (NFA Definition)

Each input file follows this structure without student names or IDs:

```
States: q0 q1 q2
Alphabet: 0 1
Start: q0
Final: q2

Transitions:
q0 0 q0 q1
q0 1 q0
q1 0 q2
q1 1 –
q2 0 q2
q2 1 q1
```

Output File Format (DFA Definition)

The output file must match the input format but represent the equivalent DFA with student IDs at the top in the exact format

```
# Student IDs: 20251234, 20259876

States: {q0} {q0,q1} {q0,q1,q2} {q0,q2}
Alphabet: 0 1
Start: {q0}
Final: {q0,q1,q2}

Transitions:
{q0} 0 {q0,q1}
{q0} 1 {q0}
{q0,q1} 0 {q0,q1,q2}
{q0,q1} 1 {q0}
{q0,q1,q2} 0 {q0,q1,q2}
{q0,q1,q2} 1 {q0,q1}
{q0,q2} 0 {q0,q2}
{q0,q2} 1 {q0,q1}
```

Implementation Details

Command-Line Arguments

The program should accept one or more file names as arguments:

```
python convert_nfa.py input1.txt input2.txt
./convert_nfa input1.txt input2.txt
```

- The program should read each NFA file, process it, and generate a DFA file with the same name but prefixed as 'DFA_output_'.

Automated Checking

- All submissions will be collected into a .zip file.
- An automated script will unzip the submissions, execute each group's program, and compare the DFA output to the expected output.
- If the format does not match, the submission will be marked incorrect.

Submission Guidelines

- Each group must submit a zip file containing:
  - Source code (.cpp or .py file).
- The zip file name should follow:
  - Group_StudentID1_StudentID2.zip