CSCI 431 Project 3

Create software that takes 5 command-line arguments: the first two are the names of input files. These two files are FSA description files (formatted in the same way as in project 1).

The last three files are output files. Each of them is an FSA description file.

The program must meet these specifications:

1. There should be an argument guard.
2. If either or both of the input files are incorrectly formatted, provide an appropriate error message and quit. Don't output files.
3. If the input files have different alphabets, provide an appropriate error message and quit. Don't output files.

   Note: `abc` and `bac` are the same alphabet, even though the order of the symbols are in a different order.
4. If all is in order, then create 3 new FSAs and output the FSA file for each.
   - The first one should recognize the union of the two languages from the input FSAs
   - The second one should recognize the intersection of the two languages from the input FSAs.
   - The last one should recognize the language that is the set difference of the language from the first input FSA minus the language from the second FSA.
5. States who have their number changed. Each of these is a separate action. Each description should give the initial number of the state and the number it is changed to.

Alternate Approach

If you prefer to create three programs (a different one for each output) you are free to do that. In this case, each program would take three arguments. The first two are the input FSAs and the last one the output FSA.

FSA definition file

The file is formatted as follows:

Line 1: This holds a string. The string represents the alphabet, so every character in the string (including a blank) is part of the alphabet except for the newline character at the end. A newline character is never part of the string.

Only printable characters with ASCII codes in the range 32-126 can be in the alphabet. In the rest of this description we use α as the the number of characters in the alphabet.

Line 2: This line holds a single number. The number is the number of states in the machine. In the rest of this description, η is used to represent the number of states.

State 0 is always taken to be the starting state.

Line 3: This line holds one or more numbers in the range $0 - (\eta - 1)$. These are the numbers of the accepting states. Adjacent numbers are separated by a single space.

The next η lines: Each of these lines represents a row in the table for the transition function (δ). The first line represents the row for state 0, the next line the row for state 1, etc.

Each line has α numbers, with adjacent numbers separated by a single space. The first number is the state to transition to if the character read is the first character of the string in line 1. The second number is the state to transition to if the second character in the string on line 1 is the character read, and so on.

Teams

You may work in pairs or individually. You may pick your own partner if you work with someone else.