CS350 – Lab 7
Due Date: March 26

Purpose: A finite automaton is a simple special case of a Turing machine in which the tape is not edited and the head always moves right. This simple computational model uses limited resources and is extremely useful in the computer science discipline. This model is also equivalent to a very useful model called a regular expression that is ubiquitous in computer science. The purpose of this lab is for you to use finite automata and regular expressions to recognize languages and to start to explore the bounds of what this model can accomplish.

Knowledge: This lab will help you become familiar with the following content knowledge:

- How to use finite automata to recognize languages.
- How to analyze the equivalence of deterministic and nondeterministic finite automata.

Task: Follow the steps in this lab carefully to complete the assignments.

Assignment 1:
Create deterministic finite automata in JFLAP to recognize each of the following languages using the alphabet $\Sigma = \{a,b\}$:

1. all strings with exactly one a.
2. all strings with no more than two a's.
3. all strings with at least one b and exactly two a's.

Criteria for Success: Test each of your dfa's and make sure that each accepts **exactly** the languages provided.

Assignment 2:
Any language that can be accepted by a DFA is called a **regular language**. Show that if L is a regular language then $L - \{\lambda\}$ is regular as well. Note that λ represents the empty string.

Criteria for Success: You can do this by describing how you would modify a DFA which accepts L so that you get another DFA which accepts $L - \{\lambda\}$. You need to have a general enough explanation so that it will apply to **any** language but it might help you to first examine a couple of particular languages that accept λ to see how they may be modified.

Assignment 3:
Complete exercise 9.11 on p190 in the text.

Criteria for Success: You have a regex for the language and have tested its correctness using JFLAP. (You will have to use the facility to convert the regex to an nfa, export this machine and then run your test inputs).

The program `egrep` is an acronym for "extended global regular expressions print" and may be used to search for a string or a more complex pattern in a file. Regular expressions provide a convenient, compact way of expressing patterns. The internal workings of egrep are based on finite automata.

The command format is:

```
        egrep 'regexp' filename
```

Egrep returns all lines in the file which contain a match for the regular expression.
The format of a regular expression in egrep is as follows:

regular expression	egrep notation
r^*	`r*`
r^+	`r+`
$r + \lambda$	`r?`
r + s	`r|s`
rs	`rs`
(r)	`(r)`
char c	c
special char	`\c`
any symbol	`.`
beginning-of-line	`^`
end-of-line	`$`
any character listed	`[   ]`
any character not listed	`[^  ]`

Assignment 4:
In a terminal window on phoenix try the following commands and then clearly and succinctly describe the pattern expressed by the regular expression:

1. `egrep 'depend' /usr/share/dict/words`
2. `egrep '^y.*y$' /usr/share/dict/words`
3. `egrep '^s..u.t..e$' /usr/share/dict/words`
4. `egrep '[qQ][^u]' /usr/share/dict/words`

Criteria for Success: You have a clear succinct description of each of the four languages.

Assignment 5:
Write `egrep` commands for the following:

1. All lines that contain the letter `a` and the letter `b` (either upper or lower case) appearing in any order.
2. All lines that contain the word `a` (either upper or lower case) followed by a word starting with a vowel

Criteria for Success: Test your commands with the file `~jillz/cs350/lab7/testfile1` and on any other testfile of your choosing and verify your results.

Assignment 6:
Try the regex `[0-9]?[0-9]:[0-9][0-9](am|pm)` on the file
`~jillz/cs350/lab7/testfile2`
Correct this regex so that it does not match illegal times like 99:99pm

Criteria for Success: Your regex works for legal times but fails for times where the hour is greater than 12 or the minutes are greater than 59.

Submit your jflap files in Canvas. Written answers may be submitted in Canvas or on paper for grading.