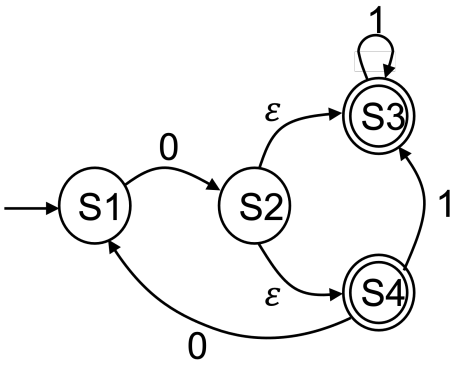
CSCI 338
Project 2

Assigned 10/4/2022, due by start of class (3:05 pm) on 10/27/2022. Please submit this assignment to the appropriate dropbox on D2L. You must follow the collaboration policy detailed on the course website.

In this project you are going to build a regular expression engine based on NFAs. This software will be able to tell you whether or not a provided string is described by a provided regular expression. You are being provided three classes to get you started:

1. *NFA.java* is a class that represents a single NFA. The representation of an NFA is already done for you. Look through the code and understand how an NFA is represented. You should not have to change this code. Understanding how the accepts() and extension() methods work will probably be beneficial to completing the rest of the assignment.

2. *Proj2.java* is the Driver. The first three lines show you how my test program will call your code. The rest of this class demonstrates how to create an NFA, just as an example to help you understand it. I will not be using your Driver when I grade your submission, so do whatever you want here, though you should probably test fairly extensively. The sample NFA created in this class is shown below:



3. *RegularExpression.java* is the class that will process and represent a regular expression. I took care of parsing the regular expression and calling the methods that you need to complete. Those methods will build NFA's for the conditions that construct a regular

expression (single characters, unions, concatenations, stars, plusses). Read the comments carefully and don't change things you are not allowed to, or shouldn't. Descriptions of how to union, concatenate, and star NFAs can be found here: `https://ecat.montana.edu/d2l/le/content/712007/viewContent/6781652/View`. You should be able to figure out how to plus them and make single character NFAs.

This code can be found here: `https://ecat.montana.edu/d2l/le/content/712007/viewContent/6781692/View`

Complete the code needed to evaluate regular expressions as indicated in the Driver. Submit the RegularExpression.java source code. If for some reason you modified NFA.java and I need your version to run it, provide that too.

Here are some hints/tips/assumptions:

- The regular expressions will be well formulated and you do not need to do error checking.
- The regular expressions will not have epsilons or the empty set in them.
- The string being tested could be the empty string: ""
- The regular expressions will only have the operations indicated (concatenation, single characters, union, star, plus) in them.
- Test test test.
- Be careful adding things to HashMaps. If there is something already associated with that key, adding (map.put()) something else will overwrite what is already there. If you want to add to what is already there, you would first need to get the entity in the map and then add to it.
- It would probably be best for each state you create for each instantiation of RegularExpression to be uniquely named. I accomplished that by having a global counter in the RegularExpression class and every time I created a state, I do so as: "S" + stateCounter++.