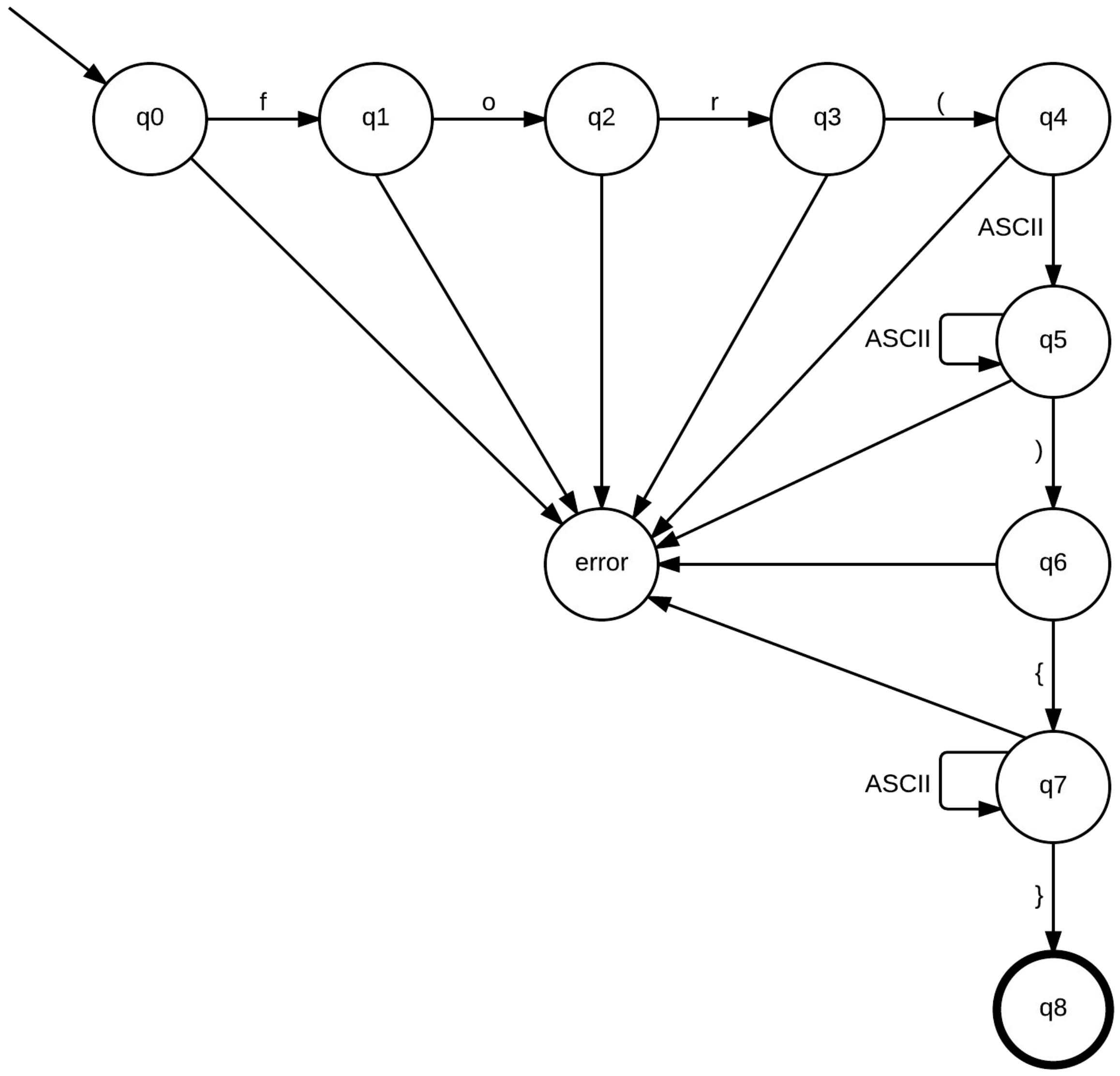
Tyler Cavera
CMPT 440
Professor Rivas
2/1/16

DFA:



Some problems that I encountered during this assignment was figuring out a way to handle the strings of input that were to be received. I wasn't sure how to properly represent the initialization, termination, and increment parts of the for loop, as well as the possible strings in the statement part of the loop. With the vast amount of possible combinations of characters, I felt that the set of characters in the ASCII set would suffice as a coverage of acceptable characters. Uppercase and lowercase letters, numbers, equality characters, and punctuation would all be acceptable inputs into the for loop. Instead of creating a much larger diagram with more states being used for the input, the ASCII loops on states q5 and q7 (hopefully) allow the validation of the parts of the for loop. Another issue I had with the diagram was the choice of either having separate, more specific errors for the different parts of the loop or just grouping everything together into one error. For the sake of space and time, I chose to only use one error state. A more general issue was that I don't know if I should be more specific in these diagrams to either reduce confusion, increase efficiency, or just be correct if I am currently not.

To implement this DFA in Java, I would start by reading whatever file the string is in with a scanner or reader. An array or linked list could be used to hold whatever alphabet is to be referenced. States could also be held in an array or preferably a linked list that has each state point to the next state. At each state, whatever character is present can be checked against the accepted alphabet, and if the state is valid, the program can move on to the next state, otherwise an error can be thrown. I suppose if-then-else statements can be used in the validation process, and for or while loops can be used to traverse the arrays or linked lists. I'm not entirely sure how large of a scale of the implementation would be, but I would assume that it would be relatively lengthy. I don't have much experience with readers or scanners in Java so it would definitely be ideal to research the implementation of these two tools. Looking at our textbook, the author provides a small scale implementation of a program that determines if numbers are divisible by 3, so I would reference and build upon this example.