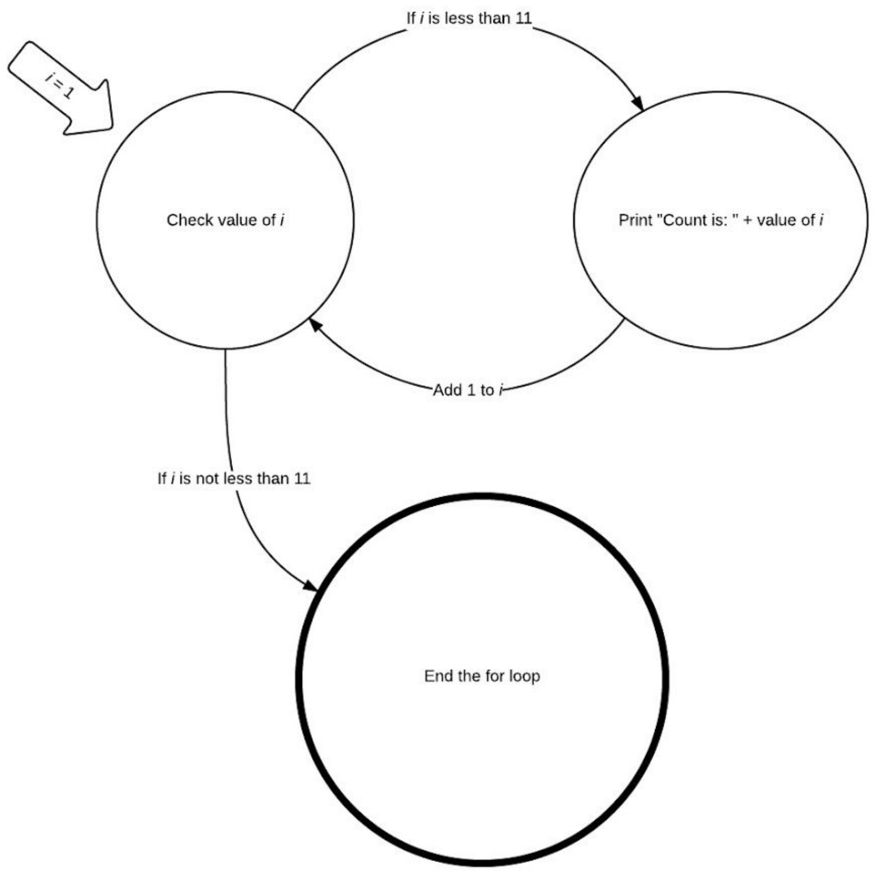
Charlie Hill
Professor Rivas
CMPT 440
1 February 2016

Lab 1: Create a DFA

The main problem that comes with creating a DFA to represent code is that there may be multiple interpretations. There are probably a few ways to represent a for loop with a DFA. Since it is a for loop, that means there are a set amount of times the loop can run. This means someone could have a new state for every time the loop restarts. However, that inefficient and contradicts the purpose of having a loop. The way I set up my diagram has the starting state checking the value of *i*. If *i* happens to be less than 11, the state changes to a state executes the for loop. Once the for loop is execute the *i* is incremented and the state changes back to first state. This process repeats until *i* is greater than 11. Once this happens the loop stops and goes to state outside the for loop that is a stop state. I have learned that representing code visually can be much more useful when trying to understand the code. This could be useful for me personally because using a visual diagram could help me plan out my program before I start writing it. It could also be useful for when trying to explain the program to others, such as a boss who may not be very tech savvy.

If you wanted to validate a string with a DFA you could take multiple routes. At first you could have each state bubble check each letter individually. However, this process does not scale, it would get very big very quick because of how many states would be necessary. This process can be simplified with a for loop. The for loop could check the string and see if it is valid. If the string is valid the for loop will end and the program can proceed. Otherwise the for loop could restart and keep checking. Depending of the program the for loop could check for letters individually or just check the entire string at once. Either way it allows us the make the program much more scalable than having to have the program error out as soon the string is deemed invalid. Validating a string in Java program could be useful for numerous reasons. Validating a password is an obvious example. The inputted password could be checked with a database and matching username. A search function would also be useful because you are searching for matches of a string that a user inputs. Without a for loop, trying to validate a string is Java would be an inefficient task, the DFA diagram helps visualize this process.


If *i* is less than 11
i = 1
Check value of *i*
Print "Count is: " + value of *i*
Add 1 to *i*
If *i* is not less than 11
End the for loop