Shreyank Patel
08/31/2021
CS 461
PA1

For this programming assignment, I had to create a corresponding DFA given a NFA. My approach to solving this problem was first concatenate all of the non-epsilon transitions and then writing a recursive function to loop through all the episolen transitions for a given state and it's child states, and then concatenate the parent episolen transitions and child episolen transitions together for the given state. Once I had the episolen transitions for a given state, I concatenated it to the non-epsilon transition string and inserted the string into a set data structure to record the new DFA state. Once i had all of the new DFA states, i looped through each of them and checked which ones contained the end state from the NFA, and recorded them in a vector; these were the new DFA final states; Finally i looped through each of the DFA states their transitions, and looked at the corresponding NFA states with the same transition and for any non-empty list, i made a string of the transition states for the NFA state and then tried to find that string my DFA set data structure. The index value + 1 of any found string was the new DFA state transition for that transition input. Some data structures I used were sets, maps and vectors. If given more time, I would have created a new data structure state and linked them using pointers to each of the transitions. Below is a pseudo code for what the data structure might look like:

```
i.   Struct state: {
       int visited;
       char stateName;
       map <transitionInput, vector<*state>> transitions
     }
```

I used gdb to ensure my recursion was correct. I also used the output.txt files provided to check if my program was output correctly. The debugging for my recursive function took longer than expected hence I ran out of time to fix my program for input3.txt. Also relearning NFA to DFA translation took longer than I had anticipated.