CPSC 351
Problem Set 8
Building a DFA Simulator

Submission Details:

1. I encourage you to work in groups of up to three people.
2. The assignment is a program. Submit the code using the Black Board submit facility. Submit a hard copy of the code to me during my office hours or in class during the week of 11/29,
3. Both pieces of the assignment are due Friday 12/3 by 5 P.M.

The Problem

1. In Problem Set 3, you completed a state diagram for a DFA that recognized L2 = {*w* | every odd position of *w* is a 1}. Call that DFA, D. Write a python program that simulates D. The program is invoked from the Linux command line.

The program must have a function that accepts a formal definition of D as a tuple (Q, Σ, **δ**, q0, F) along with its input, w. This could either be a simple function, called, say, "simulate," or the constructor for a class called, say, "DFA." The DFA will accept continuous input, printing 'accept' for each string if it is an element of L2, 'reject' if it is not. The DFA also rejects if a symbol is not an element of sigma or if a state is not an element of Q.

The operative part of the DFA is a collection of delta transitions. One technique is to model delta as a python dictionary that accepts a tuple (state,input) and returns a new state. This is known as the Kilfoyle Model, after its inventor, Jeb Kilfoyle.

Here's a pseudo-code sketch of a solution:

```
while (there are more strings to be tested)
        enter a string, w or CTRL-C to quit
        define a DFA: (Q, Σ, δ, q0, F)
        simulate(DFA,w)
                set the current state to q0
                for symbol in w
                        if current state not in Q or symbol not in Σ, reject
                        current state = δ(current state, symbol)
                accept if current state is in F, else reject
```