Math 311
Project 6 Handout

Instructions: This handout is designed to describe a project on one of the "Additional Topics" that may to choose to complete in order to satisfy the Project component of your grade in the course. As noted in the Course Policies Handout, you can earn 25 points (or more) based on the progress you make on these projects. To help make sure that you have time to work on these, you will be given time in class to work on these rather than our standard Daily Group Work assignments during the next three class periods (starting on Thursday, November 29th). You are encouraged to work in groups on these projects. You should turn in **three** projects for grading. One should be completed by Tuesday, December 4th. Your second is due on Thursday, December 6th, and your third project must be completed by Tuesday, December 11th. You will have the opportunity to earn up to 10-12 points on each of these projects. Any points beyond 25 points will count as extra credit.

Project 6: Finite State Automata and Formal Languages
This project will investigate using labeled directed graphs as a means of defining "accepted languages" by defining legal words as those that label paths in the graph that lead to and from specified vertices in the graph.

Definitions: Here are a few important definitions that we will need in order to define finite state automata.

- Let A be a set of letters. Then A^* denotes the set of all finite length **strings** or **words** over the alphabet A.
- A **language** $\mathcal{L}$ is a subset of A^* (this makes linguistic sense, since a language is a specified set of words).
- Given a word $w = a_1a_2...a_n$ with $a_i \in A$ for each i, the length of w, denoted by $|w|$, is n, the number of **letters** in the word w.
- The symbol ϵ represents a word called the **empty word** (the unique word with no letters in it). By convention, $|\epsilon| = 0$.

Examples:

- Let $A = \{a, b, c\}$. Then A^* is the collection of all words that can be formed using the letters a, b, and c.
- Let $u = ab$, $v = baba$ and $w = abbccac$. Then $|u| = 2$, $|v| = 4$, and $|w| = 7$.
- Let $\mathcal{L}$ be the set of all words of length less then or equal to 2. Then $\mathcal{L} = \{\epsilon, a, b, c, aa, ab, ac, ba, bb, bc, ca, cb, cc\}$
- Note that $u \in \mathcal{L}$ while $v, w \notin \mathcal{L}$.

Note:
• Our previous example of a language is not particularly interesting since it is only a finite collection or words. In order to "recognize" a language, one must be able to distinguish between strings of letters that form words from those that do not form words. It is important to note that if we allow words of arbitrary finite length, there are infinitely many potential words.

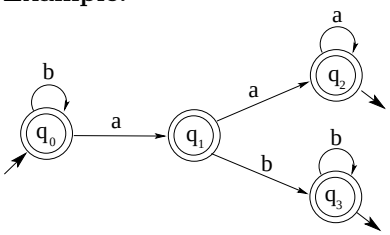
• To specify the words in a finite language, one can create a dictionary by simply listing the words in the language (it could be a very long list). If we wish to consider languages that could have infinitely many words, we need to be more creative in how we specify the words in the language. Do to this, we consider some simple computational machines called a Finite State Automata.

Definition: A **finite state automaton** (FSA) $\mathcal{A}$ is a 5-tuple (Q, A, δ, q_0, F). Here, Q is a finite set of *states*, A is a finite input alphabet, $q_0 \in Q$ is the *initial state*, $F \subseteq Q$ is the set of *final states*, and δ is a *transition function* mapping $Q \times A$ into Q. That is, $\delta(q, a)$ is a state for each state $q \in Q$ and each input symbol a.

Notes:

- We visualize a FSA by drawing it as a directed, labeled graph. The set Q is the set of *vertices* in the graph, generally called **states**.
- The vertex $q_0 \in Q$ is the **initial vertex** and the subset $F \subseteq Q$ is a set of **accept states**. Any state in $Q - F$ is called a **fail** state.
- The function $\delta : Q \times A \to Q$ defines the edges in the FSA and assigns labels to each directed edge. If $\delta(q_1, a) = q_2$, then there is a directed edge from vertex q_1 to vertex q_2 labeled by the letter a.
- A word in $w \in A^*$ is **accepted** by the FSA if there is a directed path in the FSA labeled by the word w that starts at the initial state q_0 and ends at an accept state.

Example:



Notes:

- The arrow pointing *into* the state q_0 identifies it as the initial state.
- The arrow pointing *out* from a state identifies it as an accepting state. In this automaton, q_2 and q_3 are accept states.
- States without outward pointing arrows (q_0 and q_1) are non-accepting or Fail states. Note that it **is** possible for the initial state to also be an accept state.

1. (2 points) Give examples of three words that are accepted by the FSA shown above and three words are **not** accepted by this FSA.
2. (2 points) Describe the language that is accepted by this FSA (the set of all words that are accepted) as clearly and precisely as you can.
3. (2 points) Create your own example of a FSA. Your example should use the alphabet $B = \{0, 1\}$. It should have at least 7 total states and should have more than one accept state. Finally, your FSA should accept an infinite number of words (and please try to make the resulting language at least somewhat interesting).
4. (2 points) Clearly describe the language that is accepted by your FSA.

   The Pumping Lemma: Let $\mathcal{L}$ be a regular language. There there is a constant n such that if z is any word in $\mathcal{L}$, and $|z| \geq n$ we may write $z = uvw$ in such a way that $|uv| \leq n$, $|v| \geq 1$, and for all $i \geq 0$, uv^iw is in $\mathcal{L}$. Furthermore, n is no greater than the number of states in the smallest FSA recognizing $\mathcal{L}$.

5. (2 points) Using the example of an FSA that you created above, find an n value and a specific word $z = uvw$ such that the Pumping Lemma is satisfied (if possible, make u, v, and w all non-empty words). That is, $uv^iw \in \mathcal{L}$ for all positive integers i.
6. Given an intuitive explanation for why the Pumping Lemma should be true for any language accepted by a FSA.