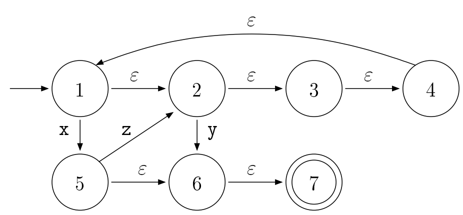
There are 7 problems in this assignment. Write your solutions clearly on white sheets of paper and turn them in at the end of the class on the due date.

1. Consider the comma-separated floating-point numbers, defined as follows. Each number consists of an integer part followed by a decimal point and an optional rational part. The integer part is either 0 or a sequence of digits separated by commas. A comma occurs after each group of three digits, starting from the rightmost digit. If the integer part is non-zero, it cannot have leading zeros and cannot start with a comma. If the rational part is present, it has at most two digits and cannot end with a zero. For instance, the number 148,576.2 is a comma-separated floating-point number. Write a regular expression which describes comma-separated numbers and draw the DFA which accepts such numbers.

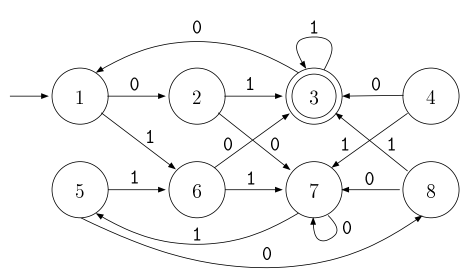
2. A comment in the C language begins with the two-character sequence `/*`, followed by the body of the comment, and then the sequence `*/`. The body of the comment may not contain the sequence `*/`, although it may contain the sequence `/*`, or the characters `*` and `/`.
   (a) Draw the DFA which accepts C comments and then use the constructed DFA to write the regular expression for C comments. Use the notation `\*` for the individual character `*`, to distinguish it from the Kleene star operator.
   (b) Write the DFA and the regular expression for C comments extended such that the sequence `*/` is permitted, as long as it appears inside quotes. For example, `/* ab "cd */" ef */` would be a valid comment.

3. Write the DFAs for each of the following:
   (a) Binary numbers with an even number of 0's and odd number of 1's;
   (b) Numbers in base 3 (consisting of digits 0, 1, and 2) which are multiples of 4.

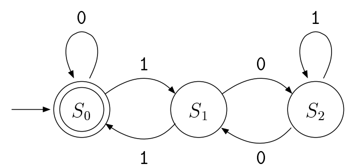
4. Convert the following NFA to a DFA. For each DFA state, indicate the set of NFA states to which it corresponds. Make sure you show the initial state and the final states in the constructed DFA.



5. Minimize the states of the following DFA. Label each state in the minimized DFA with the set of states from the original DFA to which it corresponds.



6. Write the regular expression for the language accepted by the following DFA. Hint: successively remove non-final states, replacing them with edges labeled with regular expressions.



7. Consider the following lexical analysis specification:

```
(aba)+    { return Tok1; }
(a(b*)a)  { return Tok2; }
(a|b)     { return Tok3; }
```

In case of tokens with the same length, the token whose pattern occurs first in the above list is returned.

(a) Show the NFA which accepts strings matching one of the above three patterns.

(b) Transform the above NFA to a DFA. Label the DFA states with the set of NFA states to which they correspond. Indicate the final states in the DFA and label each of these states with the (unique) token being returned in that state.

(c) Show the steps in the functioning of the lexer for the input string `abaabbaba`. Indicate what tokens does the lexer return for successive calls to `getToken()`. For each of these calls indicate the DFA states being traversed in the automaton.