Consider the Oozinoz carousel door. This carousel accepts and stores material that is passed through the doorway. The door operates with a single button. Generally we think of a door as either open, closed or transitioning between being open and closed.

If the door is closed (*B*), clicking the button makes the door start opening. If you click again before the door opens fully, the door will begin closing. If you let the door open all the way, it will automatically begin closing after a 2-second timeout. You can prevent this by clicking again when the door is open (within that 2-second timeout), and then the door will remain open (*A*) until you click the button.

If the door is open (point *A* above), clicking the button makes the door start closing. If you click again before the door is closed completely, the door will begin opening. If you let the door close all the way, it will remain closed (point *B* above).

a) Show with a statechart diagram the operation of the carousel door. Show states, transitions and guards.
b) Show the class diagram that results when you represent the operation of the carousel door using the State design pattern.
c) Use a sequence diagram to show the messages sent if someone approaches a closed carousel door, clicks the button, and while the door is opening the person clicks again.

A parser is a collection of classes that recognize tokens according to specific rules. CSV is an acronym for comma-separated-values. A CSV parser is a parser that detects each value in a CSV string. For example if the string to be parsed is

ABC,"1,456.33",12345,,,,"Smith, Jim",c c c

then the parser recognizes the following values:

ABC
1,456.33
12345
Smith, Jim
c c c

Note that this string would yield the same tokens:

"ABC","1,456.33",12345,"Smith, Jim","c c c"

Tokens (values) in the string are delimited by commas, and optionally may be enclosed in a pair of double quotes. You could say there are two types of values in a CSV string: unquoted values and quoted values. The first value is either an unquoted value beginning in the first position, or it is a quoted value because the first character is a double quote. From techniques covered in courses like ACS-1903 and ACS-1904 one could code this parser using various character methods, loops, and if-else statements, but that is not the approach we want to take … instead we are interested in designing a parser based on the State design pattern.

a) Construct a state chart diagram for this CSV parser. The parser will be in a certain state depending on characters encountered in the string. When the parser starts, it begins with the following code:

```
// the variable s contains the string to be parsed
for (int i = 0; i<s.length(); i++) {
    // process the current character according to the current state
    state.processChar(s.charAt(i));
    }
```

b) Given your states in part a) show the corresponding class diagram.

c) Given your classes from part b) illustrate the sequence diagram when the CSV string to be parsed is

A,"BB"