Solutions to the midterm

Problem 1.
1.
a) You start with generating possible states for the automaton:
States with 0, 1, 2, 3, more 0's
States for not having seen any of 101, seen 1, seen 10, seen 101

The total number of states is 5 X 4 = 20. Certainly, not all states are obtainable.

The regular expression is:
1* | 01* | 001* | 0001* | 001*0 | 01*0 | 01*00 | 1*00 | 1*0 | 1*000 | 1*001* | 01*001* | 1*0001*

b) How can we create strings like there? We can generate 0^n 1^n and then 1^m 0^m. This grammar is not regular:

S -> A B
A -> 0 A 1 |
B -> 1 B 0 |

2.
Rev(R) is a regular language: if R is a regular expression, then there is a finite automaton generating it. If we reverse the order of all transitions and switch starting and final states (given that an automaton can have multiple final states, in Rev(R) we can add another state with transitions to the final states of R)

Problem 2.
1.
a)
The proper order of answering questions is:
LR(0), SLR(1), LALR(1), LR(1)

This grammar is not LR(0) (there is a shift/reduce conflict in state S3)

The grammar is SLR(1) - first sets for the non-terminals are sufficient to disambiguate the parsing. Therefore, the grammar is LALR(1) and LR(1).

b)

2. To do that, we need to give one example of such a grammar. If a grammar is not LR(k), the decision cannot be made based on the first k characters (for example, they are all the same), but can be using k+1-character lookahead.