1. \{0100100001\ldots0^{2n}1 \mid n \geq 0\}
   This is very similar to the set from last week's problem #1, except this language can have an even or odd number of blocks.

2. Any language with the correct transition diagram receives full credit.

3. Many different solutions are possible for the regular expressions. Answers that are not easy to understand should be accompanied by a short explanation.
   a) \((1*(01*0)*\)^*
   b) \(0*1*
   c) \((00 + 01 + 10 + 11)*(0 + 1)

4. a) \(0*(1*00)*1*
   There will never be an odd number of 0’s between any two 1’s because then one of those two 1’s would have an odd number of 0’s following it.
   For example: 101___ There is no way for both 1’s to have even 0’s after.
   b) \(0*1* + 0*1(0 + 1)^*
   Any string with 10 in it must start with a 0, to have 01 before all substrings of 10.
   c) \(0*(1*(00))*0^*

5. a) \(\varepsilon + 1(10 + 01)^*(\varepsilon + 0 + 1)

b) \((00 + 01(11)*0 + (01(11)*0 +1)(0(11)*0)(0(11)*10 + 1))^*01(11)*0 + 1)(0(11)*0)^*
   One way of doing this problem is converting from the dfa to the regular expression. The yellow highlighted area are all paths that lead from the start state back to itself. The green highlighted area are all paths that lead from the state state to the final state. Finally, the blue highlighted area refers to paths from the final state to itself.
c) \((00 + 1)^*\)