Problem 1: Basics

Checking to see if an arbitrary string of size 5 is a palindrome can be calculated with a FSM.  
Checking to see if an arbitrary string of any size is a palindrome can be calculated with a FSM.  
FSMs can represent regular languages  
Every regular expression has exactly one corresponding DFA.  
On average, compared to a DFA, checking acceptance with an NFA is more computationally expensive  
NFAs have exactly one path during a graph traversal for any given input  
All DFAs are NFAs.  
A DFA can have a only one start state and final state

Problem 2: Finite State Machine Analysis

Which strings would the above Finite State Machine accept? Select all that apply.

A bb  B cab  C caccab  D baccccb  E cb
F c  G cabb  H the empty string  I cbcb  J cbb

Write a regular expression that is equivalent to the above Finite State Machine:
Problem 3: NFA to DFA

Consider the NFA and fill in the blanks of the equivalent DFA. Use the subset construction (on-demand) algorithm we gave in lecture/discussion. We will only be checking state names for partial credit.

What state(s) are final states? Select all that apply:

A S1  B S2  C S3  D S4

Scratch Space: