Some Answers on: Decidability and Recursive Languages

F1: To run a program for a standard TM on this variation, wherever the code says to change the character and move left, say going to state B, allow the head to move right but transition the machine into a new state where it moves the head left two cells and then goes to state B. Similarly with code that says to change the symbol and not move the head.

F2: (a) Assume boolean function F accepts L. To build boolean function for first-halfs: given an input string, generate all possible strings from the alphabet of that length, concatentate each to the input string in turn and see if F accepts the concatenation. Accept if and only if F says yes at least once. Since the number of possibilities is finite, this will terminate.
(b) Same proof except need to run the possibilities through F in parallel.

F3: Let q be the number of states of the TM. If the head stays on the input or within q cells of the input, then there is a finite number of configurations and so if it computes too long we can stop it. If the head goes more than q cells away from the input, then I claim the TM is stuck in an infinite loop: for, the last $q+1$ steps must have had a repeated state, and every time the TM was reading a blank. So then too we can stop it.

F4: Take M, create M_S, test for equivalence of NFAs.

F5: One idea: Discard any state from which an accept state is not reachable. Convert to having a unique accept state. Now, define a set S of states as reachable if there is a sequence of input chars such that S is exactly the set of states the NFA could be in. (These are the states in the DFA built by the subset construction.) Then the answer is yes if and only if there is a reachable set that has two identically-labeled transitions to the same state.

F6: Since the input is read-only, the current configuration of the 2FA can be given by specifying the state and the positions of the two heads. Suppose the input has length n and the number of states is q. Then there are at most qn^2 configurations. In particular, if the machine runs for more than qn^2 steps, then some configuration must have recurred, and so the deterministic machine is stuck in an infinite loop. So we can decide if the machine will halt or not by running it for this long.

F7: One idea: Let k be constant of the Pumping Lemma. We claim: If there is a string in the language of length at least k, then there is such a string of length less than $2k$. For, let z be a shortest string in the language that has length at least k, assuming there is such a string. Then by the Pumping Lemma, we can write $z = uvwxy$ where uwy is in the language as well. Since we chose z to be the shortest string with length at least k, it follows that $|uwy| < k$ and so $|z| = |uwy| + |vx| < 2k$.

Now to answer the question: test all strings with length k through $2k-1$. If there is such a string, the language is infinite and the answer is yes. If there is no such string, then test all strings with length less than k and answer yes if and only if you find 100 strings.