1.

P2:

Let T be a TM that decides P2:

“On input w:

1. If w is not of type <M> where M is a DFA over {a,b}, reject.
2. Construct a TM TEDFA where L(TEDFA) = EDFA
3. Run TEDFA on M.
4. If TEDFA accepts, reject.
5. Determine if the accept state of M has only one string that leads to it, by beginning to find strings in M. If at any point we find more than one string, reject.
6. Otherwise, accept.

All our steps are finite and for every TM x, if x is in P2, T accepts. If x is not in P2, T rejects. We are able to determine if |L(M)| = 1 in step 2 because DFAs have a finite number of states, and finite number. Notice that if M was a regular expression, and there is a \* or U, T will reject finitely. Otherwise, we find the only string in M.

Thus, P2 is decidable.

P4:

Let T be a TM that decides P4:

“On input w:

1. If w is not of type <M> where M is a DFA over {a, b}, reject.
2. Construct a TM TADFA where L(TADFA­) = ADFA
3. Run TADFA on <M, ab>
4. If TADFA accepts, accepts. Otherwise reject.”

All our steps are finite and ADFA is decidable, and for every TM x, if x is in P4, T accepts. If x is not in P4, T rejects.

Thus, P4 is decidable.

P5:

Let T be a TM that decides P5:

“On input w:

1. If w is not of type <M, M’> where M and M’ are both DFA over {a, b}, reject.
2. Construct a TM TEQDFA where L(TEQDFA) = EQDFA
3. Run TEQDFA on w.
4. If TEQDFA accepts, reject. Otherwise accept. “

All our steps are finite and EQDFA is decidable, and for every TM x, if x is in P5, T accepts. If x is not in P5, T rejects.

Thus, P5 is decidable.

P6:

Let T be a TM that decides P6:

“On input w:

1. If w is not of type <M, M’> where M and M’ are both DFA over {a, b}, reject.
2. In parallel, find the list of reachable states M and M’s transition functions, and if they are accept or reject states. At any point, if M has an accept state and M’ has a reject state, reject.
3. After all the states are ran, accept.

All our steps are finite and for every TM x, if x is in P6, T accepts. If x is not in P6, T rejects. Notice a DFA has a finite amount of states, so step 2 is finite.

Thus, P6 is decidable.

2.

a) No such language exists. Proof by contradiction:

Suppose there is an unrecognizable language whose complement is finite. Its complement must be a decidable language. A language is decidable iff both it and its complement are recognizable, which means the original unrecognizable language is recognizable, which is a contradiction.

b) No such language exists. Proof by contradiction:

Suppose there is a context-free language that is undecidable. However, we know that every context-free language is decidable by Theorem 4.9 of the textbook. It cannot be undecidable and decidable.

c) Σ\*

We know Σ\* is recognizable, as we can construct a TM that accepts every string. We also know every language is a subset of Σ\*, and it has unrecognizable subsets, such as ETM.

d) ATM

ATM is recognizable as we can construct a TM for it. It is undecidable as defined in class, as we cannot compute D on D via the diagonalization proof.

e) ATMC

We know a language is decidable if and only it and its complement is recognizable. As ATM as seen above is undecidable, ATMC must be unrecognizable. If it is unrecognizable, it is also undecidable.

3.

a)

Let T be a TM that recognizes 4stTM: (also decides)

“On input w:

1. If w is not of type <M> where M is a TM, reject.
2. Read the number of states in M.
3. If |Q| of M is 4, accept. Otherwise reject.”

We know there are Turing Machines that can be called as a subroutine to decode string representations of objects and interact with them, so we are able to count the number of states in M. Turing Machines have a finite set of states, so this is possible. Also, every step in this definition is finite.

Thus, 4stTM is recognizable.

b)

Let T be a TM that recognizes NonETM:

“On input w:

1. If w is not of type <M> where M is a TM, reject.
2. Run ETMC on w.
3. If ETMC accepts, accept. Otherwise reject.”

Notice that ETMCis recognizable. For every TM x, if x is in NonETM. T accepts. If x is not, T rejects or does not halt.

Thus, NonETM is recognizable.