CS 341, Spring 2020
Solutions for Midterm 2

1. (a) True, by Theorem 3.13.
   (b) True, by slide 4-25.
   (c) False, e.g., if $A = \{00, 11, 111\}$ and $B = \{00, 11\}$, then $\overline{A} \cap B = \emptyset$ but $A \neq B$. For A and B to be equal, we instead need $(\overline{A} \cap B) \cup (A \cap \overline{B}) = \emptyset$.
   (d) False. The set $\mathcal{N} = \{1, 2, 3, \ldots\}$ is infinite and countable.
   (e) True, because every context-free language is decidable by Theorem 4.9, and every decidable language is Turing-recognizable because the definition of Turing-recognizable is less restrictive than the definition of decidable.
   (f) True, by slide 4-38.
   (g) False, by Theorem 3.16.
   (h) False. A TM M may loop on input w.
   (i) False. $\overline{A_{\text{TM}}}$ is not Turing-recognizable by Corollary 4.23.
   (j) False. The set $A = \Re$ is uncountable, but the set $B = \{1, 2, 3\}$ is countable and $B \subseteq A$.

2. (a) No, because $f(a) = f(c) = 1$.
   (b) No, because nothing in D maps to 2 or to 4, which are both in R.
   (c) No, because f is not one-to-one and onto.
   (d) A language L_1 that is Turing-recognizable is recognized by a Turing machine M_1 that may loop forever on a string $w \notin L_1$. A language L_2 that is Turing-decidable is recognized by a Turing machine M_2 that always halts.
   (e) An algorithm is a Turing machine that always halts.

3. $q_1 aabb\#ba \quad xq_2abb\#ba \quad xaq_2bb\#ba \quad xabq_2b\#ba \quad xabbq_2\#ba \quad xabb\#q_4ba \quad xabb\#bq_ra$

4. This is HW 9, problem 1. Each element in $\mathcal{B}$ is an infinite sequence $(b_1, b_2, b_3, \ldots)$, where each $b_i \in \{0, 1\}$. We prove that $\mathcal{B}$ is uncountable by contradiction using a diagonalization argument. Suppose $\mathcal{B}$ is countable. Then we can define a correspondence f between $\mathcal{N} = \{1, 2, 3, \ldots\}$ and $\mathcal{B}$. Specifically, for $n \in \mathcal{N}$, let $f(n) = (b_{n1}, b_{n2}, b_{n3}, \ldots)$, where b_{ni} is the ith bit in the nth sequence, i.e.,

n	$f(n)$
1	$(b_{11}, b_{12}, b_{13}, b_{14}, b_{15}, \ldots)$
2	$(b_{21}, b_{22}, b_{23}, b_{24}, b_{25}, \ldots)$
3	$(b_{31}, b_{32}, b_{33}, b_{34}, b_{35}, \ldots)$
4	$(b_{41}, b_{42}, b_{43}, b_{44}, b_{45}, \ldots)$
$\vdots$	$\vdots$

Now define an infinite binary sequence $c = (c_1, c_2, c_3, c_4, c_5, \ldots) \in \mathcal{B}$, where $c_i = 1 - b_{ii}$ for each $i \in \mathcal{N}$. In other words, the ith bit in c is the opposite of the ith bit in the ith sequence. For example, if

n	$f(n)$
1	$(0, 1, 1, 0, 0, \ldots)$
2	$(1, 0, 1, 0, 1, \ldots)$
3	$(1, 1, 1, 1, 1, \ldots)$
4	$(1, 0, 0, 1, 0, \ldots)$
$\vdots$	$\vdots$

then we would define $c = (1, 1, 0, 0, \ldots)$. Thus, for each $n = 1, 2, 3, \ldots$, note that $c \in \mathcal{B}$ differs from the nth sequence in the nth bit, so c does not equal $f(n)$ for any $n \in \mathcal{N}$, which is a contradiction because the enumeration was supposed to contain every infinite binary sequence. Hence, $\mathcal{B}$ is uncountable.

5. This is HW 8, problem 4. We need to show there is a Turing machine that recognizes $\overline{E_{\text{TM}}}$, the complement of E_{TM}. Let $s_1, s_2, s_3, \ldots$ be a list of all strings in Σ^*, e.g., in string order. For a given Turing machine M, we want to determine if any of the strings $s_1, s_2, s_3, \ldots$ is accepted by M; i.e., if $\langle M \rangle \in \overline{E_{\text{TM}}}$. If M accepts at least one string s_i, then $L(M) \neq \emptyset$, so $\langle M \rangle \in \overline{E_{\text{TM}}}$; if M accepts none of the strings, then $L(M) = \emptyset$, so $\langle M \rangle \notin \overline{E_{\text{TM}}}$. However, we cannot just run M sequentially on the strings $s_1, s_2, s_3, \ldots$. For example, suppose M accepts s_2 but loops on s_1. Because M accepts s_2, we have that $\langle M \rangle \in \overline{E_{\text{TM}}}$. But if we run M sequentially on $s_1, s_2, s_3, \ldots$, we never get past the first string. The following Turing machine avoids this problem and recognizes $\overline{E_{\text{TM}}}$:

   R = "On input $\langle M \rangle$, where M is a Turing machine:
   1. Repeat the following for $i = 1, 2, 3, \ldots$.
   2. Run M for i steps on each input $s_1, s_2, \ldots, s_i$.
   3. If any computation accepts, *accept*.

6. Define the language as

$$E_{\text{NFA}} = \{ \langle N \rangle \mid N \text{ is an NFA with } L(N) = \emptyset \}.$$

   Recall that the proof of Theorem 4.4 defines a Turing machine R that decides the language $E_{\text{DFA}} = \{ \langle B \rangle \mid B \text{ is a DFA with } L(B) = \emptyset \}$. Then the following Turing machine S decides E_{NFA}:

   S = "On input $\langle N \rangle$, where N is an NFA:
   1. Convert N into an equivalent DFA D
   using the algorithm in the proof of Kleene's Theorem.
   2. Run TM R for E_{DFA} on input $\langle D \rangle$.
   3. If R accepts, *accept*. If R rejects, *reject*."