CS 341, Fall 2015
Solutions for Midterm 2

1. (a) False. A TM M may loop on input w.
   (b) True, by Theorem 4.9.
   (c) True, by slide 4-38.
   (d) False, by Theorem 4.8.
   (e) False, by Theorem 4.11.
   (f) False, e.g., $\overline{A_{\text{TM}}}$ is not Turing-recognizable.
   (g) 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 $(\overline{A} \cap B) \cup (A \cap \overline{B}) = \emptyset$.
   (h) True, by Theorem 4.5.
   (i) False, by Homework 9, problem 1.
   (j) False, by Theorems 3.13 and 3.16.

2. (a) No, because $f(x) = f(y) = 1$.
   (b) No, because nothing in A maps to $3 \in B$.
   (c) No, because f is not one-to-one nor onto.
   (d) A language L_1 that is Turing-recognizable has a Turing machine M_1 that may loop forever on a string $w \notin L_1$. A language L_2 that is Turing-decidable has a Turing machine M_2 that always halts.
   (e) An algorithm is a Turing machine that always halts.

3. $q_1 010\#1 \quad xq_2 10\#1 \quad x1q_2 0\#1 \quad x10q_2\#1 \quad x10\#q_4 1 \quad x10\#1q_{\text{reject}}$

4. (From slides 4-39 and 4-40). Let $\mathcal{L}$ be the collection of languages over an alphabet Σ, and let $\mathcal{B}$ be the set of infinite binary strings, which we know is uncountable (by a diagonalization argument). We will show that there is a correspondence between $\mathcal{L}$ and $\mathcal{B}$. Let $s_1, s_2, s_3, \ldots$ be an enumeration of the strings in Σ^*, e.g., the enumeration can list the strings in string order. Define mapping $\chi : \mathcal{L} \to \mathcal{B}$ such that for a language $A \in \mathcal{L}$, the nth bit of $\chi(A)$ is 1 if and only if the nth string $s_n \in A$. We now show χ is a correspondence.

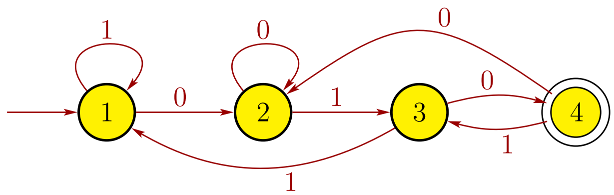
   - To show that χ is one-to-one, suppose that $A_1, A_2 \in \mathcal{L}$ with $A_1 \neq A_2$. Then there is some string s_i such that s_i is in one of the languages but not the other. Then $\chi(A_1)$ and $\chi(A_2)$ differ in the ith bit, so χ is one-to-one.
   - To show that χ is onto, consider any infinite binary sequence $b = b_1 b_2 b_3 \ldots \in \mathcal{B}$. Consider the language A that includes all strings s_i for which $b_i = 1$ and does not include any string b_j for which $b_j = 0$. Then $\chi(A) = b$, so χ is onto.

   Since χ is one-to-one and onto, it is a correspondence. Thus, $\mathcal{L}$ and $\mathcal{B}$ have the same size, so $\mathcal{L}$ is uncountable because $\mathcal{B}$ is uncountable.

5. This is a slight modification of HW 8, problem 3. Let $\Sigma = \{0, 1\}$, and the language of the decision problem is

$$A = \{ \langle R \rangle \mid R \text{ is a regular expression describing a language over } \Sigma \text{ containing at least one string } w \text{ that ends in } 010 \text{ (i.e., } w = x010 \text{ for some } x \in \Sigma^*) \}.$$

Define the language $C = \{ w \in \Sigma^* \mid w \text{ ends in } 010 \}$. Note that C is a regular language with regular expression $(0 \cup 1)^*010$ and is recognized by the following DFA D_C:



Now consider any regular expression R with alphabet Σ. If $L(R) \cap C \neq \emptyset$, then R generates a string ending in 010, so $\langle R \rangle \in A$. Conversely, if $L(R) \cap C = \emptyset$, then R does not generate any string ending in 010, so $\langle R \rangle \notin A$. By Kleene's Theorem, since $L(R)$ is described by regular expression R, the language $L(R)$ must be a regular language. Since C and $L(R)$ are regular languages, $C \cap L(R)$ is regular since the class of regular languages is closed under intersection, as was shown in Chapter 1. Thus, $C \cap L(R)$ has some DFA $D_{C \cap L(R)}$. Theorem 4.4 shows that $E_{\text{DFA}} = \{ \langle B \rangle \mid B \text{ is a DFA with } L(B) = \emptyset \}$ is decidable, so there is a Turing machine H that decides E_{DFA}. We apply TM H to $\langle D_{C \cap L(R)} \rangle$ to determine if $C \cap L(R) = \emptyset$. Putting this all together gives us the following Turing machine T to decide A:

T = "On input $\langle R \rangle$, where R is a regular expression:

1. Convert R into a DFA D_R using the algorithm in the proof of Kleene's Theorem.

2. Construct a DFA $D_{C \cap L(R)}$ for language $C \cap L(R)$ from the DFAs D_C and D_R.

3. Run TM H that decides E_{DFA} on input $\langle D_{C \cap L(R)} \rangle$.

4. If H accepts, *reject*. If H rejects, *accept*."

6. This is Theorem 5.1, whose proof is given on slide 5-8. Specifically, suppose that $HALT_{\text{TM}}$ is decidable, and let R be a TM that decides $HALT_{\text{TM}}$. Thus, for any $\langle M, w \rangle$, which is an (encoded) pair of a TM M and string w, if $\langle M, w \rangle \in HALT_{\text{TM}}$ is the input to R, then R halts and accepts; if $\langle M, w \rangle \notin HALT_{\text{TM}}$ is the input to R, then R halts and rejects. Now we build a TM S that decides A_{TM} using R as

a subroutine.

S = "On input $\langle M, w\rangle$, where M is a TM and w a string:

1. Run TM R on input $\langle M, w\rangle$.
2. If R rejects, then *reject*.
3. If R accepts, then run M on input w.
4. If M accepts, then *accept*. If M rejects, *reject*."

Note that if M accepts w, then S accepts $\langle M, w\rangle$. If M does rejects w, then S rejects $\langle M, w\rangle$. If M loops on w, then S rejects $\langle M, w\rangle$ in stage 2. Thus, S decides A_{TM}, which is impossible because A_{TM} is undecidable. Therefore, $HALT_{\text{TM}}$ is also undecidable.