The more challenging problems are marked with $*$. Be concise when describing a Turing machine. It is like writing pseudocodes. It suffices to present the most important ideas behind your Turing machines. You do not need to give all the details, e.g., the set of states and the transition function. Check the course website for more info about homeworks. CC: Computational Complexity. TC: Introduction to the Theory of Computation. MA: Computational Complexity: A Modern Approach.

1. Problem 3.14 on page 149 of TC (8 points): Show that the collection of decidable languages is closed under the operations of **a).** union. **b).** concatenation. Given two languages L and L', their concatenation is the following language L^*: a string $\mathbf{w} = w_1 \dots w_n$ is in L^* if there exists an i such that $w_1 \dots w_i \in L$ and $w_{i+1} \dots w_n \in L'$. **c).** complementation. **d).** intersection. (Xi: I removed the operation of "star" from the problem but its solution is similar to that of "concatenation".)

2. Problem 3.11 on page 148 of TC (10 points): A Turing machine with doubly infinite tape is similar to an ordinary Turing machine except that its tape is infinite to the left as well as to the right. The tape is initially filled with blanks except for the portion that contains the input. Computation is defined as usual except that the head never encounters an end to the tape as it moves leftward. Show that this type of Turing machines decides (Xi: I changed the word "recognizes" in the original problem to "decides".) the same class of languages as ordinary Turing machines.

3. Problem 3.10 on page 148 of TC (10 points)*: Say that a write-once Turing machine is a single-tape TM that can alter each tape cell at most once (including the input portion of the tape). Show that this variant Turing machine model decides the same class of languages as ordinary Turing machines. (Hint: As a first step consider the case whereby the Turing machine may alter each tape cell at most twice. Use lots of tape. Xi: You will get 7 points for write-twice Turing machines. The key for write-once is how to mark cells without altering them when making a copy of a string.)

4. Problem 3.17 on page 149 of TC (10 points)*: Show that single-tape TMs that cannot write on the portion of the tape containing the input string can only decide regular languages. (Xi: A regular language is a language accepted by a finite-state automata. Given such a TM, what are the states of your automata? You will receive 7 points by solving Problem 3.13, an easier version of 3.17.)

5. Problem 3.4.1 on page 66 of CC (12 points): For each of the following problems involving Turing machines, determine whether or not it is decidable: **a).** Given a Turing machine M, does it halt on the empty string? **b).** Given a Turing machine M, is there a string for which it halts? **c).** Given a Turing machine M, is $L(M)$ finite? Here $L(M)$ denote the set of strings $\mathbf{w}$ such that $M(\mathbf{w}) =$ "yes". **d).** Given two Turing machines M and M', is $L(M) = L(M')$? (Xi: To prove a problem is undecidable, show that if there is a TM that decides it, then one can construct a TM to decide the halting problem or *a problem that you have already shown to be undecidable.*)

6. Problem 3.4.5 on page 67 of CC (10 points): Show that L is recursive if and only if there is a Turing machine M that enumerates L, and such that the strings in L are output by M in length-increasing fashion. (For the if direction, if L is finite there is nothing to prove. So suppose it is infinite ...) (Xi: Read pages 61-62 of CC or pages 140-141 of TC for the definition of a Turing machine enumerating a language L.)