Worth: 9% **Due:** Thursday October 1 (at tutorial)

For each question, please write up detailed answers carefully. Make sure that you use notation and terminology correctly, and that you explain and justify what you are doing. Marks **will** be deducted for incorrect or ambiguous use of notation and terminology, and for making incorrect, unjustified, ambiguous, or vague claims in your solutions.

1. In *dyadic* notation, integers are represented in base 2 using the digits 1 and 2 (instead of 0 and 1). For example, 10 is represented by the dyadic string "122" because $10 = 1 \cdot 4 + 2 \cdot 2 + 2 \cdot 1$. Note that each integer has a *unique* dyadic representation, because there cannot be any leading digits. This is unlike binary notation, where each integer has infinitely many representations (*e.g.*, 10 is represented by each of the infinitely many binary strings "1010", "01010", "001010", ...). In particular, the unique dyadic representation of 0 is the empty string (ε). For any dyadic string $s \in \{1,2\}^*$, val(s) represents the decimal value of s (*e.g.*, val("122") = 10 and val("") = 0).

   Design a TM that multiplies two dyadic integers. More precisely, when started in initial configuration "$\sqcup\ q_0\ w$" (for input string $w \in \{1,2,\times\}^*$), your TM must have the following behaviour:

   - if w does not contain the character $\times$, or w contains $\times$ more than once, then your TM eventually halts in configuration "$\sqcup\ q_R\ w$", where q_R is the rejecting state;
   - if w contains exactly one occurrence of $\times$, *i.e.*, $w = x \times y$ for some $x, y \in \{1,2\}^*$, then your TM eventually halts in configuration "$\sqcup\ q_A\ z$", where q_A is the accepting state and z is the dyadic representation of val(x) $\times$ val(y).

   Give a high-level description (the main idea in one paragraph), an implementation-level description (a numbered list describing details of head movements and tape contents without mentioning states), and a formal-level description (a full transition table or diagram, as well as an explicit list of the tape alphabet) of your TM. Make sure to relate your formal-level description to the equivalent stages in your implementation-level description.

   To help you, let M_+ be another TM with the following behaviour: when started in initial configuration "$w\#\ q_+\ x{+}y$", where $x, y \in \{1,2\}^*$ and $w \in \Gamma^*$, M_+ eventually reaches configuration "$w\#\ q_D\ z$", where z is the dyadic representation of val(x) + val(y). The behaviour of M_+ is not defined for other initial configurations, M_+ does not define any transition out of state q_D (this state only marks the end of M_+'s computation), and M_+ never moves its head left of the square that contains #.

   You may make use of M_+ within your own TM, by having transitions into state q_+ to add two dyadic integers (as long as the tape contents and head position are set up appropriately), and by defining appropriate transitions out of state q_D to continue with the computation once the addition is done. Do *not* provide any transition out of state q_+ or into state q_D—that is what M_+ does. You can simply *use* M_+ without concerning yourself with its implementation.

2. String $x \in \Sigma^*$ is a *prefix* of string $y \in \Sigma^*$ if $y = xz$ for some string $z \in \Sigma^*$. For example, the prefixes of 0101 are: ε (the empty string), 0, 01, 010, 0101.

   For any language $L \subseteq \Sigma^*$, define another language $L' \subseteq \Sigma^*$ as follows:

   $$L' = \{w \in \Sigma^* : L \text{ contains every prefix of } w\}$$

   (a) Prove that if L is decidable, then L' is decidable.

   (b) If L is recognizable but not necessarily decidable, what can be concluded about L'? Prove your claim.

3. Consider the following model for Turing Machines: The tape is doubly infinite, and the read/write head moves i steps (left or write) after i'th stage (transition) of the computation. In other words, after the first step the head moves one-step left or right, after the second step it moves two steps left or right, and so on. Other than than the model is the same as the standard one.

   Show that this model is equivalent with the standard model of TM. The description should be in the implementation level.

   Remark: You may use the facts shown in class and tutorial that doubly-infinite tape TM as well as MMultiTape TM are equivalent with standard TM.