CS5230: Tutorial 1

1. Consider the following assembly language Z.
   (a) There is one input variable \( X \), and one output variable \( Y \), and any fixed number of
       other temporary variables. Each variable can hold a non-negative integer.
   (b) The instructions may or may not have labels.
   (c) The instructions in the assembly languages are of the form:
       i. \( V \leftarrow V + 1 \). Meaning: Add one to variable \( V \).
       ii. \( V \leftarrow V - 1 \). Meaning: Subtract one from variable \( V \). If \( V = 0 \), then subtraction
           doesn’t have any effect (i.e. \( V \) remains 0).
       iii. IF \( V \neq 0 \), then GOTO L. Meaning: If \( V \neq 0 \), then go to instruction number \( L \).
           Otherwise go to the next instruction.
   (d) A program is any finite sequence of instructions of above form.
       In execution of any program in the above language, initially all temporary variables
       and output variable have value 0 (input variable has value as given in input). The
       execution of program ends (if ever) when there is no next instruction to execute (in
       case of instructions of type (i) and (ii) above, the next instruction is the instruction
       immediately after the current instruction in the list of finite instructions). When the
       program halts, output is the value of the output variable \( Y \).

   Does above assembly language have the same power (as far as function computation goes)
   as a Turing Machine? Give justification for your answer.

2. Consider the problem of sorting \( m \) numbers that you studied in your algorithms class.
   (a) Model how input for above problem can be given to the Turing Machine.
   (b) Consider any algorithm for sorting (you may want to choose a simple algorithm).
       Show how you might implement it on the Turing Machine. You may just give a sketch of
       how the Turing Machine does important steps of the algorithm. You need not give the
       state table of Turing Machine itself.
   (c) How much time does your algorithm take? Giving the time complexity in \( O \) notation
       is fine.

3. (a) Give details of how one can simulate a multi-tape TM using one tape TM.
     (b) Consider the simulation of multi-tape Turing Machine done in part (a). If the multi-
         tape TM takes time \( t \) before it halts on input \( x \), can you provide a reasonable bound on
         the time taken by the simulation?
     For the following two questions, \( n \) is the length of the input.

4. (a) Suppose a Turing Machine \( M \) is \( n \)-time bounded. Can we construct another Turing
     Machine which accepts the same language as \( M \), but is \( n/2 \) time bounded?
   (b) Suppose a Turing Machine \( M \) is \( n \)-space bounded. Can we construct another Turing
       Machine which accepts the same language as \( M \), but is \( n/2 \) space bounded?

5. Can you give a recursive language, which cannot be accepted in time \( O(2^n) \)? \( O(2^{2^{2^n}}) \)?