Exercise sheet 3: Decidability

1. An *enumerator* is a Turing Machine that outputs strings one at a time. For instance, a TM that outputs 1, then 11, then 111, et cetera, is said to be an *enumerator* for the language 11^*. You may assume that the TM has a state that triggers it to output, and it has a second tape specifically to write its output down.

   The language of an enumerator is the set of all strings it will output after any finite amount of time.

   Prove that if a language has an enumerator that produces the strings in that language in lexographic order, then that language is decidable.

2. Prove that if a language is decidable, there is an enumerator for that language that outputs the strings in lexographic order.

3. Let $NE_{TM} = \{\langle M \rangle \mid M \text{ accepts a string}\}$. Prove that NE_{TM} is undecidable.

4. Here is an attempt to solve the halting problem. Explain the flaw in the reasoning, and give an example of a machine and a string that the proposed method would fail on. (Note: there is actually a flaw here, not merely a lack of detail. You should find something that is false in this paragraph and explain why, or identify something implicitly assumed and explain why that is false.)

   "In order to determine if a Turing machine is going to loop, we just have to simulate it while keeping track of every configuration it enters. If the simulated machine ever enters the same configuration twice, we know it will loop, and thus, we reject. To keep track of the configurations, we use a second tape. When we enter a new configuration, we scan this new tape to see if this new configuration had been seen before; if so, we reject, and if not, we write this configuration at the end of the second tape. Whenever a TM enters the same configuration twice, it will loop, thus, when we reject, the machine does not halt. If the Turing machine does halt it will never enter the same configuration twice, so we will accept. Thus, this method shows that the halting problem is decidable."