Problem 1 Explain how you would design a Turing machine deciding the language $L = \{1^{n^2} \mid n \geq 0\}$. You do not need to give the state diagram for your machine, but you should give sufficient detail that (given enough patience!) one could write down the state diagram based on your description. (For examples of the right level of detail, see the descriptions of the Turing machines in Examples 3.11 and 3.12 in Sipser.)

Problem 2 Let $L$ be a language. Prove that if both $L$ and its complement $\bar{L}$ are R.E. (i.e., Turing-recognizable), then $L$ is recursive (i.e., decidable).

Problem 3 The input to a Turing machine is always a string, but we will want to use Turing machines to reason not just about strings but about objects such as graphs, automata, grammars, and even other Turing machines. In order to do this, we will encode each object $O$ that is the input to the Turing machine as a string $\langle O \rangle$ over the machine’s input alphabet $\Sigma$; the input to the Turing machine will consist of the object in encoded (i.e., string) form.

Explain how you would encode a DFA $A = (\Sigma_A, Q_A, \delta_A, q_0A, F_A)$ as a string $\langle A \rangle$ suitable for input to a Turing machine. You can choose anything convenient for the Turing machine’s alphabet $\Sigma$.

Problem 4 A Turing machine with a doubly infinite tape is one in which the machine’s read head can move left off the beginning of the input string as well as right off the end of the input string. All tape cells not part of the input string, in both directions, initially contain the blank character “\_”.

a. Turing machines with doubly infinite tapes are described using the same 7-tuple as the Turing machines (with singly infinite tapes) discussed in class and in Sipser; start configurations and accepting configurations are also the same. However, the rules for how a configuration $C_i$ yields a configuration $C_{i+1}$ are different. Give the rules for how $C_i$ yields $C_{i+1}$ for a Turing machine with doubly infinite tape, including the special-case rules that apply when the machine’s read head is close to either edge of the area it has used so far on the tape.

b. Explain how to simulate a Turing machine with a singly infinite tape on a Turing machine with a doubly infinite tape. (Be sure to handle the “can’t move left” special case.)

c. Explain how to simulate a Turing machine with doubly infinite tape on a Turing machine with singly infinite tape.