Problem 1

Design a Turing Machine that recognizes all palindromes over the alphabet $\Sigma = \{a, b\}$.

Problem 2

Answer the following questions about Turing machines, with explanations: (Answer all 3 correctly for credit.)

a) Can a Turing machine ever write the blank symbol $\sqcup$?

b) Can a tape alphabet Γ be the same as the unput alphabet Σ?

c) Can a Turing machine contain just a single state?

Problem 3

Let a k-PDA be a pushdown automaton that has k stacks. Thus, a 0-PDA is an NFA and a 1-PDA is a conventional PDA. You already know that 1-PDAs are more powerful (recognize a larger class of languages) than 0-PDAs. [Sip96]

Show that 2-PDAs are more powerful than 1-PDAs. (Pick a language that cannot be recognized by a 1-PDA, and give a high-level description of how a 2-PDA could be designed to recognize it.)

Problem 4

Here are some examples of decision problems involving Turing machines. Is it decidable whether a given Turing machine:

a) has at least 481 states?

b) takes more than 481 steps on input ϵ?

c) takes more than 481 steps on *some* input?

d) takes more than 481 steps on *all* inputs?

If it *is* decidable, give a high-level description of an algorithm to decide it.
If not, explain (at a high level) why. [Koz07]

Problem 5

Explain (at a high level) why it is undecidable whether a given Turing machine M accepts a specific string w.

References

[Koz07] Dexter C Kozen. *Automata and computability*. Springer Science & Business Media, 2007.

[Sip96] Michael Sipser. Introduction to the theory of computation. *ACM Sigact News*, 27(1):27–29, 1996.