Exercise 1

Although much of our discussion of Turing machines will take place at a high level, it’s still instructive to try to design Turing machines at the level of individual states.

a. Every context-free language can also be decided by a Turing machine. For instance, consider $L_a = \{ w \in \Sigma^* \mid w$ is a palindrome $\}$, where $\Sigma = \{0, 1\}$. Draw the state-transition diagram for a Turing machine whose language is $L_a$.

I recommend you test that your design works for at least two inputs – one that should be accepted and one that should be rejected.

You can trace the operation of your Turing machine by hand, or you can use the online simulator at turingmachine.vassar.edu. (Use # for a blank.)
b. We’ve shown that the language $L_b = \{ a^n b^n c^n \mid n \in \mathbb{N}_0 \}$ is not context-free and therefore cannot be recognized by a pushdown automaton (nor by a finite automaton). However, it can be recognized by a Turing machine. Draw a state-transition diagram for a Turing machine whose language is $L_b$. 
Exercise 2

Design a Turing machine that decides the language \( \{ w \mid w \text{ contains twice as many } 0s \text{ as } 1s \} \) over the alphabet \( \{0, 1\} \).

Give an implementation-level description in the style of Example 3.7 on page 171 (not a formal description or a state diagram).
Exercise 3

What does it mean to solve a problem? If $L$ is a language over $\Sigma$ and $M$ is a Turing machine with input alphabet $\Sigma$, any of these properties may hold:

1. $M$ is a decider (i.e., halts on all inputs).
2. For any string $w \in \Sigma^*$, if $M$ accepts $w$, then $w \in L$.
3. For any string $w \in \Sigma^*$, if $M$ rejects $w$, then $w \notin L$.

To claim that a Turing machine solves a problem, it seems like it should have at least some of these properties. But we can show that having just two of these properties doesn't say much.

a. Prove that if $L$ is any language over $\Sigma$, then there is a Turing machine $M$ that satisfies properties 1 and 2 with respect to $L$.

b. Prove that if $L$ is any language over $\Sigma$, then there is a Turing machine $M$ that satisfies properties 1 and 3 with respect to $L$.

c. Prove that if $L$ is any language over $\Sigma$, then there is a Turing machine $M$ that satisfies properties 2 and 3 with respect to $L$.

d. Suppose $L$ is a language over $\Sigma$ for which there is a Turing machine $M$ that satisfies properties 1, 2, and 3. What can you say about $L$?
Exercise 4

Give a high-level description of a Turing machine $M$ such that $L(M) \in \mathbb{R}$, but $M$ is not a decider.

(This shows that just because a Turing machine's language is decidable, it's not necessarily the case that the TM itself must be a decider.)

This doesn't need to be a specific language. How could you modify a decider for some language $L$ to be only a recognizer?