1.15 The construction described in this problem is similar to the construction in the proof of Theorem 1.49. The difference is that the construction in the problem uses the original start state and makes it into an accept state. The proof of Theorem 1.49 adds a new start state which is an accept state and there is an ε jump from the new start state to the original start state. Thus, this problem is designed to illustrate that it is necessary to introduce a new start state rather than just changing the original start state to an accept state. The reason for this is that there are some DFA’s in which (i) the start state is not an accept state, and (ii) it is possible to leave the start state and come back to it. If you
change the start state into an accept state under these conditions, you will accept all strings that will leave the accept state and finish at the accept state – these strings should not be in $A_1^*$. Below is an example:

This DFA accepts all strings of alternating 0’s and 1’s that begin and end with a 0. Below are the Kleene star construction described in the proof of Theorem 1.49 and the one described in the problem. First the one from the proof of the theorem

In this NFA, you must still begin and end with a 0, though you are no longer restricted to having alternating 0’s and 1’s (you can have consecutive 0’s but not consecutive 1’s).

Now the construction from the problem

In this NFA it is possible to accept a string ending in a 1. Since no string accepted by the original NFA can end in a 1, it is impossible that applying the Kleene star would result in accepting a string ending in a 1.
1.17b The diagram below labels the states with the subscripts of the original states as opposed to the set of original states. Also, I have not included the ∅ state because it makes the resulting DFA unreadable. (For example, state 4 has no edge corresponding to input 0. There is an “implied” edge there to the ∅ state.)

1.31 Since a language is regular iff it is recognized by a DFA, it suffices to show that for any DFA, M, there is a DFA, M₁, such that L(M₁) = L(M)^	ext{R}. Also, since any NFA has a corresponding DFA that accepts the same language, M₁ may be an NFA. Let M = (Q, Σ, δ, q₀, F). Intuitively, we want M₁ to follow the paths of M in reverse. Therefore, (i) we will have a new start state that has an ε jump to each of the original accept states, (ii) all transitions will reverse direction, and (iii) the original start state will be the new accept state. More formally, define M₁ = (Q₁, Σ, δ₁, q₀₁, F₁), where Q₁ = Q ∪ {qₛ}, q₀₁ = qₛ, F₁ = {q₀}. Finally, define δ₁ as follows: (i) δ₁(q,a) = r whenever δ(r,a) = q, and (ii) δ₁(qₛ,ε) = r for every r in F.

Claim: w is recognized by M iff w^R is recognized by M₁.

Proof: Assume w = w₁w₂...wₙ is recognized by M. Then there is some sequence of states q₁, q₂, ..., qₙ₊₁ in M such that (i) q₁ = q₀ (the start state of M), (ii) for each i = 1, 2, ..., n, qᵢ₊₁ = δ(qᵢ,wᵢ), and (iii) qₙ₊₁ is in F. Now consider what happens when M₁ reads w^R = wₙwₙ₋₁...w₁. Since qₙ₊₁ is in F, there is an ε jump from qₛ to qₙ₊₁ in M₁. Furthermore, by definition, for every i = 1, 2, ..., n δ₁(qᵢ₊₁,wᵢ) = qᵢ. Therefore, when M₁ reads w^R, it is possible to follow the sequence of states qₛ, qₙ₊₁, qₙ, ..., q₂.
q₁. I.e., M₁ recognizes wᴿ since q₁ is an accept state of M₁. This shows that if M reaccepts w, then M₁ accepts wᴿ.

The proof that M₁ accepts wᴿ implies that M accepts w uses the same argument.

1.32 By the previous problem, we can show B is regular by showing Bᴿ is regular. We can show Bᴿ is regular by constructing a DFA that accepts B. The DFA will have three states – N (for non-carry), C (for carry), and I (for impossible). The transition function is below

<table>
<thead>
<tr>
<th></th>
<th>N</th>
<th>N</th>
<th>I</th>
<th>I</th>
<th>N</th>
<th>I</th>
<th>N</th>
<th>C</th>
<th>I</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>I</td>
<td>N</td>
<td>C</td>
<td>I</td>
<td>C</td>
<td>I</td>
<td>I</td>
<td>C</td>
<td></td>
</tr>
<tr>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td>I</td>
<td></td>
</tr>
</tbody>
</table>

The start state of this machine is N and the accepting set is {N}. The diagram for this machine is shown below.

By design, this machine accepts Bᴿ. Thus, Bᴿ is a regular language. By problem 1.31 B is also a regular language.