1. (a) Let the language of all reversed strings be denoted by $R(L(A))$. We claim that $R(L(A))$ is indeed regular.
To show this, consider the new automaton A' obtained from A by making every final state as initial and making the initial state final. Moreover, we reverse all the arrows of A.
For formally, if A was equal to $(Q, \Sigma, \delta, q_0, F)$, then $A' = (Q, \Sigma, \delta', F, \{q_0\})$ where $\delta' : Q \times \Sigma \to 2^Q$ is defined as:

$$\delta'(q, a) = Q_a \text{ where } Q_a = \{q' \in Q : \delta(q', a) = q\}.$$

Thus, A' is an NFA and the language accepted by A' is precisely $R(L(A))$. As languages accepted by NFAs are regular, we are done.
We don't need to prove that $L(A') = R(L(A))$ in this course but let us see a proof anyway.

Proof. Suppose w is a word that has a run in A from the state q_0 to q_f. We now show that the reverse of w has a run in A' from any set of states containing q_f to a set of states containing q_0 such that this run accepted the reverse of w.
We shall show this by induction on the length of the word w.

Base case: $|w| = 0$. In this case, we get that $q_0 = q_f$. It is easy to conclude the result as any set containing q_f does contain q_0. Moreover, the only word of length 0 is ϵ which is indeed its own reverse.

Inductive hypothesis: Whenever w is a word with $|w| \leq n$ ($n \geq 0$) such that w has a run from the state q_0 to q_f in A, then reverse of w has a run in A' from any set of states containing q_f to a set of states containing q_0 such that this run accepted the reverse of w.

Inductive step: Suppose w is a word such that $|w| = n + 1$. Let $q_f = \hat{\delta}(q_0, w)$ be the final set of w in A. By definition of $\hat{\delta}$, this means that there is a run from q_0 to q_f that gives w. Now we wish to show that there exists a run in A' from any set of states containing q_f to a set of states containing q_0 such that the run accepts the reverse of w.
Now, consider the prefix of w of length n. Let this be w'. Let the last letter of w be denoted by α. Then, note that we have $\delta(\hat{\delta}(q_0, w'), \alpha) = q_f$.
Then, by our definition of δ', we get that $\hat{\delta}(q_0, w') \in \delta'(q_f, \alpha)$.
Now, consider the state $\hat{\delta}(q_0, w')$. w' has a run in A from q_0 to $\hat{\delta}(q_0, w')$, by definition of $\hat{\delta}$. Thus, by our inductive hypothesis, there is a run from $\delta'(q_f, \alpha)$ to a set containing q_0 such that this run accepts the reverse of w'. Also, observe that reverse of w' is just α concatenated with reverse of w'.
Thus, we have shown that there is a run from q_f to a set containing $\hat{\delta}(q_0, w')$ (accepting α) from which there is a run to q_0 accepted w'. This completes our induction.

Now, let w be any word accepted by A. Then w has a run starting at q_0 and ending at some state $q_f \in F$. Then, by our construction of A', we get that $\{q_0\}$ is the set of final states and q_f is in the initial set of states. By our previous result, we get that reverse of w is accepted by A'.

Now, we must show that any word accepted by A' is indeed in $R(L(A))$. I leave this to the reader.

(b) We shall assume that a language accepted by an $\epsilon-$NFA is regular.
L is given to be a regular language. Thus, there exists a DFA A that accept L as its language.
Suppose $A = (Q, \Sigma, \delta, q_0, F)$.
Consider the $\epsilon-$NFA given by $A' = (Q, \Sigma', \delta', \{q_0\}, F)$ where $\Sigma' = (\Sigma \setminus \{a\}) \cup \{\epsilon\}$ and $\delta' : Q \times \Sigma' \to 2^Q$

is defined as

$$\delta'(q,\alpha) := \begin{cases} \{\delta(q,\alpha)\} & \text{if } \alpha \neq \epsilon \\ \{\delta(q,a)\} & \text{if } \alpha = \epsilon \end{cases}$$

Thus, A' is the automaton obtained from A by replacing every a edge by an ϵ edge.
It is clear by construction that the language accepted by A' is $L \downarrow \{b,c\}$ and hence, it is regular.

(c) The idea is to construct a new automaton by connecting all the final states of the original automaton to a new state via an ϵ and making that as the only final state.
Formally, suppose $A = (Q, \Sigma, \delta, Q_0, F)$ is our original NFA.
Consider the new $\epsilon-$NFA $A' = (Q \cup \{q_f\}, \Sigma', \delta', Q_0, \{q_f\})$ where $\Sigma' = \Sigma \cup \{\epsilon\}$ and $\delta' : Q \cup \{q_f\} \times \Sigma' \to 2^{Q \cup \{q_f\}}$ is defined as:

$$\delta'(q,\alpha) := \begin{cases} \delta(q,\alpha) & \text{if } \alpha \neq \epsilon \\ \varnothing & \text{if } \alpha = \epsilon \text{ and } q \notin F \\ \{q_f\} & \text{if } \alpha = \epsilon \text{ and } q \in F \end{cases}$$

(Note that the second line of definition is basically saying that I don't want any new ϵ edges from non-final states. It is still important to put that from a mathematical point of view.)
It is easy to see that this $\epsilon-$NFA does accept the same language as the original automaton. (Proof?)

(d) No, it is not. We shall prove this with a minimal counterexample. Consider $Q = \{q_0\},\ \Sigma = \{a\},\ \delta : Q \times \Sigma \to Q, F = \varnothing$. δ is defined as $\delta(q_0, a) = q_0$. (Obviously. (Why obviously?))
Consider the automaton $N = (Q, \Sigma, \delta, q_0, F)$. It is clear that $L(N) = \varnothing$ as there is no final state. (Note that is **not** $\{\epsilon\}$.)
However, if we consider N_1 as defined in the question, then we see that $L(N_1) = a^*$.
Thus, $L(N_1) \neq (L(N))^* = \varnothing$.

(e) Since L is given to be a regular language, there exists an automaton A accepting it.
Let $A = (Q, \Sigma, \delta, q_0, F)$ be the formal description.
Let $Q' = Q \times Q,\ \Sigma' = \Sigma,\ Q_0 = \{q_0\} \times F$.
Moreover, define $F' = \{(q,\ q) : q \in Q\}$ and $\delta' : Q' \times \Sigma' \to 2^{Q'}$ as follows:

$$\delta'((q_1,q_2),a) := \{\delta(q_1,a)\} \times Q_a$$

$$\text{where } Q_a = \{q \in Q : \exists b \in \Sigma \text{ such that } \delta(q,b) = q_2\}$$

Now, consider the automaton $A' = (Q', \Sigma', \delta', Q_0, F')$.
The language accepted by A' is precisely $L_{\frac{1}{2}}$.

The reason this works is because intuitively - we are approaching the middle of a word from the beginning and the end (that's why our Q_0 was $\{q_0\} \times F$) and we accept a word once we reach a common state. (That's why our F' is the set of all (q,q) as this means that we've reached a common point from both ends.)

2. Let us first note a few things.
For integers a and b, let $a \bmod b$ denote the (positive) remainder left when a is divided by b.
As an example, $16 \bmod 9 = 7$.
Suppose an integer $n \geq 1$ is given. We claim that the sequence

$$2^0 \bmod n,\ 2^1 \bmod n,\ 2^2 \bmod n,\ \ldots$$

is eventually periodic.
For example, if $n = 3$, then the above sequence is $1, 2, 1, 2 \ldots$. If $n = 4$, then it is $1, 2, 0, 0, \ldots$. If $n = 5$, then it is $1, 2, 4, 3, 1, 2, 4, 3, \ldots$.
The reason that this happens is that there are at most n distinct remainders, thus a value must occur twice. If that happens, then the values after that are also fixed.

To form a binary string, our language must be $\{0, 1\}$. It makes sense to keep a track of the remainder in order to determine whether the number formed is a multiple of n.
To achieve this, let us think of states as tuples of numbers of the form $\begin{pmatrix} a \\ b \end{pmatrix}$ where a will keep track of the remainder of the number formed so far and b will keep track of the remainder of power of 2 at that stage.
Thus, we have n^2 states as a and b can possibly take all integer values in $[0, n]$.
Given a state $\begin{pmatrix} a \\ b \end{pmatrix}$, we need two outgoing edges from it, one for 0 and one for 1. By our construction, it is easy to see that it the 0 edge will take us to $\begin{pmatrix} a \\ 2b \bmod n \end{pmatrix}$ and the 1 edge will take us to $\begin{pmatrix} (a+b) \bmod n \\ b \bmod n \end{pmatrix}$.

Now, for our initial state, we define a new dummy state such that the 0 edge from that state takes us to $\begin{pmatrix}0\\0\end{pmatrix}$ and 1 edge takes us to $\begin{pmatrix}1 \bmod n\\1 \bmod n\end{pmatrix}$.

Thus, we now have a finite set of states with the transition function defined and hence, we have a DFA. This gives us that L_n is regular.
Note that this DFA will accept multiples of n interpreted in reverse. That is, for $n = 4$, it will accept 001 which is the reverse of 4 written in binary but it won't accept 100.
But even if we do want to accept it the right way forward, part (a) tells us that the language is still regular.

Draw this for $n = 3$ to see how it's actually working.

3. It is clear that if a language L is regular, then there is an NFA with the devilish acceptance that does accept it. Namely, any DFA accepting L is also (essentially) an NFA with the devilish acceptance and we know that *a* DFA must exist by definition of being regular.
The reason we write "essentially" is because formally, there is a slight difference.
Namely, if $A = (Q, \Sigma, \delta, q_0, F)$ is the DFA, then the corresponding NFA is $A' = (Q, \Sigma, \delta', \{q_0\}, F)$ where $\delta' : Q \times \Sigma \to 2^Q$ is defined as $\delta'(q, a) = \{\delta(q, a)\}$.

   Now, to prove the converse, let A be a NFA. We shall now construct a DFA A' that accepts the same language accepted by the NFA with the devilish acceptance.
   Let $A = (Q, \Sigma, \delta, Q_0, F)$ be the formal definition of A. Consider $Q' = 2^Q$ and $F' = 2^F \setminus \{\varnothing\}$. Thus, we now have that $Q_0 \in Q'$ and $F' \subset Q'$.
   Also, let us define $\delta' : Q' \times \Sigma \to Q'$ as:

$$\delta'(X, a) := \bigcup_{q \in X} \delta(q, a).$$

   (Check that this actually makes sense.) Thus, the automaton A' given by the description $(Q', \Sigma, \delta', Q_0, F')$ is a DFA. Moreover, by construction, the final states only consist of those subsets of Q in which every state is a final state. Moreover, by removing $\varnothing$, we have also ensured that there indeed is *a* state.
   (The construction is almost identical to what we did in class when we converted an angelic NFA to a DFA. Only the final states have been slightly modified to account for devilish condition.)
   Thus, we have now shown that any language accepted by a DFA is indeed regular.

4. Since L is given to be a regular language, there exists an automaton A accepting it.
   Let $A = (Q, \Sigma, \delta, q_0, F)$ be the formal description.
   We shall now construct an automaton accepting L' by simply removing the outgoing edges from any final state. We can do this by either creating a new DFA with a trap state or make our life simpler by simply considering an NFA.
   However, one mustn't always go for shortcuts and hence, we shall construct a new DFA.
   Define $Q' = Q \cup \{q'\}$ and $\delta' : Q \times \Sigma \to Q$ as

$$\delta'(q, \alpha) := \begin{cases} \delta(q, \alpha) & \text{if } q \notin F \cup \{q'\} \\ q' & \text{if } q \in F \cup \{q'\} \end{cases}$$

   Then, our required DFA is simply $A' = (Q', \Sigma, \delta', q_0, F)$.
   Now, let that argue that $L(A') = L'$.
   Suppose $w \in L(A')$. Then, at no point during ***the*** run of w in A' were we at a final state. This is because the only outgoing edges from any final state are to the trap state. As the final states of A and A' coincide, we get that no proper prefix of w is accepted by A. Thus, $w \in L'$. Hence, we have shown that $L(A') \subset L'$.
   Conversely, suppose that $w \in L'$. Then, the only stage at which we come across a final state in the run of w in A is at the final stage. Thus, we never traverse any outgoing edge from any final state. As all such edges are preserved in A' as well, we get that $w \in L(A')$ and hence $L' \subset L(A')$.
   Thus, we have shown that $L(A') = L$.

   At this point I should mention that it probably was easier to make a DFA rather the NFA as formally writing an NFA is actually more troublesome. Moreover, the argument was actually easier to write as we could talk about "***the***" run of a word in A' since it was deterministic which gave us a unique run.

5. Let S_{n-1} denote the prefixes of length $n-1$ of the words that are in L_n. That is, $S_{n-1} = \{w \in \Sigma^* : |w| = n-1 \text{ and } wy \in L_n \text{ for some } y \in \Sigma^*\}$.
   Let A be a DFA accepting L_n. [1]

[1] It is easy to see that *a* DFA does exist as it's easy to construct an NFA accepting L_n.

We claim that if $x, y \in S_{n-1}$ such that $x \neq y$, then $\hat{\delta}(q_0, x) \neq \hat{\delta}(q_0, y)$, where $\hat{\delta}$ and q_0 have their usual meanings.
That is, we are claiming that the runs of x and y will land us at different states.

Proof. Let $x, y \in S_{n-1}$ be given such that $x \neq y$.
Then, x and y differ at some position i where $1 \leq i \leq n-1$. Let i be the biggest such integer. (That is, x and y must agree at every position from i onwards.)
Without loss of generality, assume that x has a 0 at position i and y has a 1.
Consider the words $\bar{x} = x0^i$ and $\bar{y} = y0^i$. That is, the words obtained by appending i 0s at the end of x and y. Thence, $\bar{x}$ and $\bar{y}$ are of lengths $n + i - 1$.
Now, if we consider the digit at the n^{th} position from the right in $\bar{x}$, we get a 0. On the other hand, we get a 1 for $\bar{y}$. Thus, $\bar{y}$ is accepted by A while $\bar{x}$ is not A. Or more simply, $\bar{x}$ and $\bar{y}$ must end in different states.
Now, by our assumption, we had that x and y ended at the same state. As $\bar{x}$ and $\bar{y}$ were obtained by adding i extra zeros at the end, we get that $\bar{x}$ and $\bar{y}$ end up at the same state since A is deterministic, a contradiction.

□

Thus, we get that the number of states is at least $|S_{n-1}|$. Now, we claim that $|S_{n-1}| = 2^{n-1}$. If we are able to show this, then our proof will be complete.
This is easy to show as there are 2^{n-1} possible words of length $n-1$. Moreover, all of these will actually be the prefix of a word in L_n. Just consider the word obtained by appending a 1 followed by $n-1$ 0s.