A. Let INF_{TM} be the set of all Turing machine encodings $\langle M \rangle$ such that $L(M)$ is infinite. Some Turing machines have infinite languages (such as the TM that accepts all inputs) and some Turing machines have finite languages (such as the TM that rejects all inputs). So being finite vs. infinite is a non-trivial property of recognizable languages. So INF_{TM} is undecidable by Rice's theorem.

B. I will argue that Joshutrons are equivalent to NTMs in power. Therefore they are more powerful than PDAs or NFAs.

First, it is easy to see that any Joshutron J has an equivalent NTM N: The NTM tape has all of the capabilities of a queue, and more. Conversely, let N be an NTM. We will define a Joshutron J that has roughly the same states as N, but possibly some extra states to implement certain manipulations of the queue. The queue of J will store the tape of N as follows. If the contents of the tape are

$$a_1 \cdots a_{i-1} a_i a_{i+1} \cdots a_m$$

with the tape head over the ith cell, then the queue contains

$$a_i a_{i+1} \cdots a_m \# a_1 \cdots a_{i-1},$$

where # is a special marker for the left end of the tape. On input $w = w_1 \cdots w_n$, J does these steps:

1. Enqueue the input string. All of J's remaining transitions will be ϵ-transitions, which consume no input but rather just enqueue or dequeue. Enqueue #. At this point, the queue contains $w_1 \cdots w_n \#$.
2. For each transition of the NTM:
   (a) Deqeueue. Let a be the dequeued symbol and q the current state of J.
   (b) If N has a transition $\delta(q, a) = (r, b, R)$, then:
      i. Enqueue b.
      ii. Move to state r.
   (c) If N has a transition $\delta(q, a) = (r, b, L)$, then:
      i. Enqueue b with a special mark.
      ii. Mimic our Turing-machine-with-left-reset homework problem. Repeatedly scan the queue, manipulating marks, until the cell at the front of the queue is the cell just to the left of where b was just written. Along the way, erase the mark over b.
      iii. Move to state r.

For the sake of clarity, but at the expense of completeness, this algorithm omits the handling of two special cases: When the tape head moves right off the end of the tape or tries to move left off the end of the tape. Both cases can be detected by the presence of # at the front of the queue. Extra subroutines can be added to detect and handle these cases. [I omit the details.]

C. Let DEC be the set of all Turing machine encodings $\langle M \rangle$ such that M is a decider. Is DEC decidable? Prove your answer.

Suppose, for the sake of contradiction, that D decides DEC. We will use D to build a decider H for $HALT_{TM}$. The decider H, on input $\langle M, w \rangle$, does:

1. Build a Turing machine N that, on input x, ignores x and instead runs M on w, outputting whatever M outputs.
2. Run D on N, outputting whatever D outputs.

Neither of H's steps can fail to terminate, so H is a decider. To see that H decides $HALT_{TM}$, notice that each of the following statements is logically equivalent with its immediate neighbors.

1. $\langle M, w \rangle \in HALT_{TM}$.
2. M halts on w.
3. N halts on all inputs x.
4. N is a decider.
5. D accepts $\langle N \rangle$.
6. H accepts $\langle M, w \rangle$.

But $HALT_{TM}$ is not decidable. This contradiction implies that DEC is also not decidable.

D. In class we have proved that every context-free language is an element of P. Because every regular language is context-free, we immediately conclude that every regular language is an element of P.