10. Given that the polynomial has a root at x_0, we can say the following.

$$
\begin{aligned}
c_1{x_0}^n + c_2{x_0}^{n-1} + \cdots + c_n x_0 + c_{n+1} &= 0 \\
c_1{x_0}^n &= -(c_2{x_0}^{n-1} + \cdots + c_n x_0 + c_{n+1}) \\
|c_1{x_0}^n| &= |c_2{x_0}^{n-1} + \cdots + c_n x_0 + c_{n+1}| \\
|c_1{x_0}^n| &<= |c_2{x_0}^{n-1}| + \cdots + |c_n x_0| + |c_{n+1}| \\
|c_1{x_0}^n| &<= |c_2||{x_0}^{n-1}| + \cdots + |c_n||x_0| + |c_{n+1}| \\
|c_1{x_0}^n| &<= c_{max}(|{x_0}^{n-1}| + \cdots + |x_0| + 1)
\end{aligned}
$$

Now, if $c_{n+1} = 0$, then $x = 0$ is also a root of the polynomial. In that case, the given inequality holds true.

Else, we must have $|x_0| >= 1$. Therefore, we get

$$
\begin{aligned}
|c_1{x_0}^n| &<= c_{max} n |{x_0}^{n-1}| \\
|x_0| &<= n c_{max}/|c_1| \\
|x_0| &< (n c_{max}/|c_1|) + 1 \\
&<= (n c_{max}/|c_1|) + (c_{max}/|c_1|) \\
&<= (n+1) c_{max}/|c_1|
\end{aligned}
$$

This concludes the proof.

13. There are two sides to this proof.

For the first direction, say we are given an enumerator E that enumerates the language L in standard string order. Then we can use this to build a decider D for L. On a given input w, D runs the enumerator E until either it prints w or some string w' larger than w in standard string order. This way, D terminates on any given input, and accepts w iff $w \in L$.

For the second direction, say we are given a decider D that decides L. Then, we can build an enumerator E for L as follows. E steps through all strings in the alphabet in standard string order, and for each invokes D to determine whether it belongs to L or not. E prints w iff $w \in L$.

This concludes the proof.

14. Since B is Turing recognizable, some enumerator E enumerates it. We'll

build another enumerator E' that does the following. E' runs E, and for each TM description printed by E, checks if it is larger in terms of standard string order than the previous printed TM description. If yes, then E' prints it as is. If no, then E' modifies the TM description somehow to make it larger in terms of standard string order than the previous printed TM description, without altering the effective behavior of the TM, and then prints it. This can be done for example by adding extra (unused) states to the TM.

We can clearly see that E' enumerates C, a language consisting of TM descriptions equivalent to those in B, in standard string order. Just like we did in problem 13, we can use this enumerator to build a decider for C. This concludes the proof.

20. We'll demonstrate that the stay-put Turing machine recognizes the class of regular languages. It is easy to see that a stay-put Turing machine that always moves the tape head to the right on each symbol and finally enters either the accept or reject state without looping forever is equivalent to a DFA. Even though the stay-put Turing machine may write onto the tape, the written contents will always be to the left of the tape head and never influence the execution of the machine.

Now, we'll take a general stay-put Turing machine and transform it one step at a time into the above special form, which will conclude the proof.

Say we are given some general stay-put Turing machine $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$. We'll build $M' = (Q, \Sigma, \Gamma, \delta', q_0, q_{accept}, q_{reject})$ another stay-put Turing machine with the special form described above, as follows.

Firstly, for each transition of the form $\delta(q, a) \to (q_{special}, b, d)$ in M where $q_{special} \in \{q_{accept}, q_{reject}\}$ and $d \in \{S, R\}$, we will add a transition of the form $\delta'(q, a) \to (q_{special}, b, R)$ in M'. This captures the idea that during the transition into an accept or reject state, it is irrelevant whether the machine keeps the tape head stay put or moves it to the right.

Secondly, starting at every state and symbol, we'll search for all instances of the below sequence of transitions in M

$$\begin{aligned}
\delta(q_1, t_1) &\to (q_2, t_2, S) \\
\delta(q_2, t_2) &\to (q_3, t_3, S) \\
&\dots \\
\delta(q_{k-1}, t_{k-1}) &\to (q_k, t_k, S) \\
\delta(q_k, t_k) &\to (q_{k+1}, t_{k+1}, R)
\end{aligned}$$

And correspondingly add only the following (effective) transition to M'. This

ensures that we get rid of pointless stay-put transitions.

$$\delta'(q_1, t_1) \to (q_{k+1}, t_{k+1}, R)$$

Thirdly, starting at every state and symbol, we'll search for instances of the below sequence of transitions in M

$$\begin{aligned}
&\delta(q_1, t_1) \to (q_2, t_2, S) \\
&\delta(q_2, t_2) \to (q_3, t_3, S) \\
&\quad \cdots \\
&\delta(q_k, t_k) \to (q_i, t_i, S) | i \in \{1, 2, \cdots, k\}
\end{aligned}$$

And correspondingly add the following (effective) transition to M'. This ensures that the machine won't get stuck in an endless loop due to staying put.

$$\delta'(q_1, t_1) \to (q_{reject}, \sqcup, R)$$

These efforts will have got rid of all transitions that cause the tape head to stay put. Finally, starting at every state and symbol, we'll search for instances of the below sequence of transitions in M

$$\begin{aligned}
&\delta(q_1, t_1) \to (q_2, t_2, R) \\
&\delta(q_2, t_2) \to (q_3, t_3, R) \\
&\quad \cdots \\
&\delta(q_k, t_k) \to (q_i, t_i, R) | i \in \{1, 2, \cdots, k\}
\end{aligned}$$

And correspondingly add the following (effective) transition to M'. This ensures that the machine won't get stuck in an endless loop moving the tape head rightwards.

$$\delta'(q_1, t_1) \to (q_{reject}, \sqcup, R)$$

Now, for any transition in M not already discarded through the above methods, add them into M'.

M' is a stay-put Turing machine that never stays put (i.e moves the tape head right each step), and never gets stuck in a loop. Therefore, we have completed our desired construction, thus concluding the proof.