Exercise 1. An independent set of a graph $G = (V, E)$ is a subset of vertices $S \subseteq V$ such that no two vertices of $S$ are adjacent in $G$. Consider the Independent Set and the Maximum Independent Set problems.

**Independent Set**
- **Input:** Graph $G$, integer $k$
- **Question:** Does $G$ have an independent set of size at least $k$?

**Maximum Independent Set**
- **Input:** Graph $G$
- **Output:** A largest independent set of $G$

1. Given a $O(1.2^n)$-time algorithm for Maximum Independent Set, design an algorithm for Independent Set with running time $O(1.2^n \cdot \text{poly}(n))$.
2. Given a $O(1.2^n)$-time algorithm for Independent Set, design an algorithm for Maximum Independent Set with running time $O(1.2^n \cdot \text{poly}(n))$.

**Solution sketch.**
1. Run the algorithm for Maximum Independent Set on $G$. If it returns a set of at most $k$ vertices, return Yes, otherwise return No.
2. First, determine the size OPT of a largest independent set (sequentially by trying $k = 0, 1, \ldots$ or by binary search). Then, select an arbitrary vertex $v$. Check whether there is a largest independent set that contains this vertex by running Independent Set($G - v$, OPT). If so, recurse on $G - v$; otherwise recursively find a largest independent set of $G - N[v]$ (of size OPT-1) and add $v$.

Exercise 2. Discussion topic.
Are Nondeterministic Turing Machines realistic computation models?
- Is this a good representation of how our computing devices work?
- What is different?
What about Deterministic Turing Machines?

**Exercise 3.** Design a Deterministic Turing Machine \((Q, \Gamma, \Sigma = \{0, 1\}, q_0, A, \delta)\) that accepts palindromes. A palindrome is a word that is equal to its reverse; e.g., 011010110.

**Solution sketch.**

- Overwrite first symbol by a blank, and remember (state) if it was a 0 or a 1.
- Move the head all the way to the right until hitting a blank symbol.
- Move one spot to the left and check whether the symbol corresponds to what we remembered before, otherwise move to a non-accepting state.
- Overwrite with a blank. Move all the way to the left and start over until the tape is blank, which is when we move to an accepting state.

**Exercise 4.** A vertex cover in a graph \(G = (V, E)\) is a subset of vertices \(S \subseteq V\) such that every edge of \(G\) has an endpoint in \(S\).

<table>
<thead>
<tr>
<th>Vertex Cover</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input:</strong></td>
</tr>
<tr>
<td><strong>Question:</strong> Does (G) have a vertex cover of size (k)?</td>
</tr>
</tbody>
</table>

- Prove that **Vertex Cover** is NP-complete.

**Hints.**

- Reduce from Clique.
- The *complement* of \(G = (V, E)\) is the graph \(\overline{G} = (V, \overline{E})\), where \(\overline{E} = \{\{u, v\} : u, v \in V\text{ and } \{u, v\} \notin E\}\).