3. A.)

   use an enumerator to find every possible permutation of nodes then test whether it contains all edges

   Since we know if there's an enumerator there's a decider we will use an enumerator to find all the different permutations of nodes that we could find.

   N = "on input ⟨G⟩, where G is a graph.

   1. for i 1...|V|
      2. simulate enumerator E to find every subset c of i nodes of G.
         3. test each subset whether G contains all edges connecting nodes in c.
         4. if yes, output c.

   Running time: This construction is correct because we will test every possible permutation of nodes and output all of the accepting sets.

   step 1: runs for how ever many nodes there are in the graph (will always be finite)

   step 2: finds every permutation of i nodes. This is large but still a finite number

   step 3. This check if each node is connected to every other node therefore this will be O(n^2)

   step 4. O(1).

   step 3 is based on step 2 and step 2 is based on step 1. However since these are all finite steps based on i they will not grow exponentially.

   Therefore the worst case running time is still polynomial O(n^k)

   Validation:

   B.) Since we can pass anything in for k for regular clique, we can pass something in that makes it grow exponentially fast. For example we could pass n in for k. Then it would no longer be polynomial on a deterministic machine. However for clique$_k$ since our user cannot pass in a k, we know that k cannot grow larger than the number of nodes, therefore it will always be finite, making the running time of clique$_k$ polynomial.