Se puede incluir el for en actualizar para evitar duplicarlo pero quizá se pierde un poco de claridad y la verdad es que la duplicación limitada no es trascendente en este caso

```java
import java.util.HashMap;
import java.util.Map;
import java.util.Scanner;

public class App {
   static Scanner sc = new Scanner(System.in);
   static Map<String, Integer> ultimoDinastia ;

   static void ActualizarMapa(String nombreRey) {
       Integer ultimo = ultimoDinastia.get(nombreRey); // si no hay rey get() devuelve null
       ultimoDinastia.put(nombreRey, (ultimo == null) ? 1 : ultimo + 1);

   }

   public static void main(String[] args) {

       int reyes = sc.nextInt();
       while (reyes != 0) {
           ultimoDinastia = new HashMap<>();
           String nombreRey = "";
           // bucle para procesar dinastia inicial

           for (int i = 0; i < reyes; i++) {
               nombreRey = sc.next();
               ActualizarMapa(nombreRey);
           }

           // bucle para procesar sucesores y generar salida
           int reyesSucesores = sc.nextInt();
           for (int i = 0; i < reyesSucesores; i++) {
               nombreRey = sc.next();
               ActualizarMapa(nombreRey);
               System.out.println(ultimoDinastia.get(nombreRey));
           }
           System.out.println("");

           reyes = sc.nextInt();
       }

    }
}
```