Una de las partes que más me interesaron de esta práctica fue la forma en la que se compilaron y ejecutaron los archivos de la práctica. Empezaré con la compilación ya que al inicio me costó trabajo poder comprender el funcionamiento de todas las opciones que contiene. La parte que más me gustó fue el poder tener un directorio específicamente para los archivos compilados. Este tipo de orden ya lo había visto en otros proyectos pero jamás lo había utilizado. La mayoría de las implementaciones para los métodos de las interfaces ***ArrayList*** y ***LinkedList*** ya las habíamos visto en clase por lo que no tuve problemas para poder crear las implementaciones faltantes. Sin embargo, la implementación que más me gustó crear la que hice para el método ***limpiarLista*** para la interfaz ***LinkedList*** la cuál se puede ver en la siguiente imagen.

```
@Override
public void limpiarLista() {

  if(esVacia()){
    throw new NullPointerException("Lista está vacía");
  }

  while(!esVacia()){
    eliminarElementoInicio();
  }

  this.primero = null;
  this.ultimo = null;
  tam=0;


}
```

Figura 1. Implementación del método limpiarLista

Al inicio del método se verifica si la lista contiene algún elemento ya que en caso que esté vacía se lanza una excepción que indica que la lista está vacía. En caso que contenga algún elemento se repite las veces que sean necesarias el método que elimina el primer elemento de la lista. Decidí usar este método ya que no realiza ninguna operación de repetición, todo el proceso de eliminación lo hace por medio la redirección de apuntadores. Una vez que la lista está vacía se toman los atributos ***primero*** y ***ultimo*** de la clase y se apuntan a nulo, así mismo el tamaño de la lista asigna a 0. Personalmente esta solución me pareció la más elegante y eficiente para poder limpiar los elementos de una lista.