1. Crear archivo y verificar existencia:
   * Crea un programa en Java que cree un archivo llamado miArchivo.txt en el directorio C:/prueba y verifique si ya existe antes de crearlo.
2. Eliminar archivo existente:
   * Implementa un programa que elimine un archivo llamado archivo\_a\_borrar.txt y verifique si la eliminación fue exitosa.
3. Crear directorio nuevo:
   * Crea un directorio llamado nuevaCarpeta dentro del directorio C:/prueba y muestra un mensaje en la consola indicando si fue creado correctamente.
4. Escribir en archivo de texto:
   * Escribe un programa que permita al usuario escribir una cadena en un archivo textoSalida.txt utilizando FileWriter.
5. Leer archivo línea por línea:
   * Implementa un programa que lea el contenido del archivo textoEntrada.txt línea por línea usando BufferedReader y lo muestre en la consola.
6. Copiar archivo:
   * Desarrolla un programa que copie el contenido de un archivo entrada.txt a salida.txt usando FileReader y FileWriter.
7. Verificar si una ruta es absoluta o relativa:
   * Crea un programa que determine si una ruta proporcionada es absoluta o relativa utilizando el método isAbsolute() de la clase File.
8. Listar archivos en un directorio:
   * Escribe un programa que liste todos los archivos y subdirectorios dentro de C:/prueba y los muestre en consola.
9. Renombrar archivo:
   * Desarrolla un programa que renombre un archivo llamado archivo.txt a archivoRenombrado.txt en el mismo directorio.
10. Acceso secuencial:
    * Crea un programa que lea el contenido de un archivo de texto de manera secuencial y lo imprima en la consola.
11. Escribir en una posición específica con RandomAccessFile:
    * Usa RandomAccessFile para escribir una cadena en una posición específica de un archivo llamado datos.txt.
12. Leer desde una posición específica con RandomAccessFile:
    * Implementa un programa que lea desde una posición específica en un archivo utilizando RandomAccessFile y muestre el resultado en consola.
13. Listar archivos .txt en un directorio:
    * Escribe un programa que liste solo los archivos con extensión .txt en el directorio C:/prueba, utilizando un filtro con FilenameFilter.
14. Listar solo directorios:
    * Crea un programa que liste solo los directorios dentro de C:/prueba utilizando el método listFiles() y verificando si es un directorio.
15. Ordenar archivos por nombre:
    * Implementa un programa que liste todos los archivos en un directorio y los ordene alfabéticamente antes de mostrarlos.
16. Escribir bytes en archivo binario:
    * Desarrolla un programa que escriba un array de bytes en un archivo binario llamado datos.bin usando FileOutputStream.
17. Leer archivo binario:
    * Crea un programa que lea el contenido de un archivo datos.bin utilizando FileInputStream y muestre los datos en consola.
18. Modificar archivo binario con RandomAccessFile:
    * Usa RandomAccessFile para modificar un byte en una posición específica dentro de datos.bin.
19. Copiar archivo con Files.copy():
    * Implementa un programa que copie un archivo desde origen.txt a destino.txt utilizando el método Files.copy().
20. Mover archivo con Files.move():
    * Crea un programa que mueva un archivo mover.txt de un directorio a otro utilizando Files.move().
21. Eliminar archivo con Files.delete():
    * Desarrolla un programa que elimine un archivo llamado borrar.txt usando el método Files.delete() y maneje posibles excepciones.
22. Escritura en un archivo binario:
    * Desarrolla un programa en Java que escriba datos numéricos (como enteros y floats) en un archivo binario llamado datos.bin usando DataOutputStream.
23. Lectura de un archivo binario:
    * Crea un programa que lea el archivo datos.bin utilizando DataInputStream y muestra los datos leídos en consola.
24. Copiar un archivo binario:
    * Implementa un programa que copie el contenido de un archivo binario llamado original.bin a un archivo nuevo llamado copia.bin utilizando FileInputStream y FileOutputStream.
25. Leer archivo línea por línea:
    * Crea un programa que lea un archivo de texto lectura.txt línea por línea utilizando BufferedReader y muestra el contenido en consola.
26. Escribir y agregar texto a un archivo:
    * Desarrolla un programa que escriba texto en un archivo escribir.txt y luego agregue más contenido al final del archivo usando FileWriter en modo *append*.
27. Filtrar y listar archivos por tamaño:
    * Escribe un programa que filtre y liste solo los archivos de más de 1MB en un directorio específico usando la clase File.
28. Leer archivo de acceso aleatorio:
    * Utiliza RandomAccessFile para leer y mostrar el contenido de un archivo datos.txt desde una posición específica.
29. Modificar un archivo usando RandomAccessFile:
    * Crea un programa que modifique el contenido de un archivo de texto en una posición específica utilizando RandomAccessFile.
30. Copiar contenido de un archivo secuencialmente:
    * Desarrolla un programa que copie un archivo de texto origen.txt a destino.txt línea por línea utilizando BufferedReader y BufferedWriter.
31. Listar archivos con extensión específica:
    * Escribe un programa que liste solo los archivos .java en un directorio usando un filtro con FilenameFilter.
32. Listar archivos ordenados por tamaño:
    * Desarrolla un programa que liste los archivos en un directorio ordenados por su tamaño, de mayor a menor.
33. Listar solo archivos ocultos:
    * Crea un programa que muestre únicamente los archivos ocultos de un directorio usando el método isHidden().
34. Copiar archivo con Files.copy():
    * Implementa un programa que copie un archivo documento.txt de un directorio a otro utilizando el método Files.copy() de java.nio.file.
35. Mover un archivo con Files.move():
    * Crea un programa que mueva un archivo movible.txt de un directorio a otro utilizando el método Files.move() de java.nio.file.
36. Eliminar un archivo con Files.delete():
    * Desarrolla un programa que elimine un archivo llamado eliminar.txt de un directorio utilizando el método Files.delete().
37. Lectura de un archivo XML usando DOM:
    * Crea un programa en Java que lea un archivo XML llamado libros.xml, cargue su contenido en un árbol DOM, y muestre todos los títulos de libros que contiene.
38. Modificación de un archivo XML usando DOM:
    * Implementa un programa que modifique el contenido de un archivo XML, cambiando el precio de un libro específico dentro del archivo libros.xml.
39. Validación de un archivo XML:
    * Desarrolla un programa que verifique si un archivo libros.xml cumple con un esquema XSD proporcionado, utilizando DocumentBuilderFactory y un validador.
40. Generar un archivo XML desde Java:
    * Escribe un programa que genere un archivo XML llamado clientes.xml, que contenga una lista de clientes con nombre, dirección y teléfono, utilizando Transformer y DocumentBuilder.
41. Conversión de un archivo XML a otro formato:
    * Crea un programa en Java que convierta un archivo XML llamado productos.xml a formato HTML utilizando XSLT.
42. Lectura y escritura de atributos en XML:
    * Implementa un programa que lea los atributos de un archivo XML, como el idioma (lang) de un elemento <libro>, y luego modifique estos atributos y guarde los cambios en el archivo.
43. Acceder a nodos específicos en un archivo XML:
    * Escribe un programa que acceda a los nodos de un archivo XML libros.xml y muestre todos los valores de los nodos <autor>.
44. Modificar nodos en un XML:
    * Desarrolla un programa que cambie el contenido de un nodo <precio> en un archivo XML y guarde los cambios en un archivo libros\_modificado.xml.
45. Eliminar nodos de un XML:
    * Implementa un programa que elimine un nodo <libro> del archivo XML libros.xml si el precio del libro es superior a un valor determinado.
46. Procesar un XML utilizando StAX:
    * Crea un programa que procese un archivo XML llamado empleados.xml utilizando StAX, mostrando el nombre y el cargo de cada empleado.
47. Escribir en un XML utilizando StAX:
    * Desarrolla un programa que escriba un archivo XML empleados\_stax.xml con una lista de empleados utilizando la API StAX.
48. Filtrar elementos en un XML con StAX:
    * Implementa un programa que filtre y muestre solo los libros con precio superior a 30 en un archivo libros.xml utilizando StAX.
49. Manejo de eventos XML con StAX:
    * Crea un programa que procese un archivo XML llamado eventos.xml utilizando StAX y capture los eventos de inicio y fin de cada elemento, mostrando su nombre.
50. Capturar atributos en eventos XML:
    * Desarrolla un programa que capture y muestre los atributos de elementos específicos en un archivo XML productos.xml usando StAX.
51. Modificar XML utilizando StAX y eventos:
    * Implementa un programa que modifique los valores de los elementos <precio> en un archivo XML productos.xml utilizando la API de eventos de StAX.
52. Crear un archivo temporal:
    * Crea un programa en Java que utilice la clase Files para crear un archivo temporal en el sistema y verifique si fue creado correctamente. Usa el método Files.createTempFile().
53. Eliminar un archivo temporal:
    * Desarrolla un programa que elimine un archivo temporal creado previamente usando el método Files.deleteIfExists() y muestra si la operación fue exitosa.
54. Comprobar si un archivo temporal existe:
    * Escribe un programa que verifique si un archivo temporal existe en el directorio temporal del sistema utilizando la clase File y el método exists().
55. Crear y listar archivos de un directorio:
    * Desarrolla un programa que cree varios archivos en un directorio y luego liste todos los archivos de dicho directorio usando File.listFiles().
56. Renombrar un archivo:
    * Crea un programa que renombre un archivo existente antiguo.txt a nuevo.txt usando la clase File y el método renameTo().
57. Eliminar un directorio:
    * Implementa un programa que elimine un directorio completo, incluyendo todos los archivos que contiene, utilizando un método recursivo con la clase File.
58. Crear múltiples directorios:
    * Crea un programa que cree una estructura de directorios anidados (por ejemplo, C:/prueba/carpeta1/carpeta2) utilizando el método mkdirs() de la clase File.
59. Verificar permisos de un archivo:
    * Desarrolla un programa que verifique si un archivo tiene permisos de lectura, escritura y ejecución utilizando los métodos canRead(), canWrite(), y canExecute() de la clase File.
60. Obtener propiedades de un archivo:
    * Implementa un programa que obtenga y muestre las propiedades de un archivo, como su tamaño y la fecha de última modificación, utilizando los métodos length() y lastModified().
61. Obtener ruta absoluta de un archivo:
    * Escribe un programa que convierta una ruta relativa a una ruta absoluta para un archivo documento.txt utilizando el método getAbsolutePath() de la clase File.
62. Comparar rutas de archivos:
    * Desarrolla un programa que compare dos rutas de archivos y determine si son iguales, utilizando los métodos getCanonicalPath() y equals() de la clase File.
63. Comprobar si una ruta es un directorio o un archivo:
    * Crea un programa que verifique si una ruta dada es un archivo o un directorio utilizando los métodos isFile() y isDirectory() de la clase File.
64. Filtrar archivos por extensión:
    * Crea un programa que filtre y liste solo los archivos con la extensión .txt en un directorio dado usando FilenameFilter.
65. Filtrar archivos ocultos:
    * Desarrolla un programa que filtre y liste solo los archivos ocultos en un directorio utilizando el método isHidden() de la clase File.
66. Crear un filtro personalizado:
    * Implementa un filtro que liste solo los archivos cuyo nombre contenga una palabra específica, utilizando FilenameFilter y un criterio personalizado de búsqueda.
67. Serialización de un objeto:
    * Crea un programa que serialice un objeto de la clase Persona con atributos como nombre, edad y dirección. Guarda el objeto en un archivo persona.ser utilizando ObjectOutputStream.
68. Deserialización de un objeto:
    * Desarrolla un programa que lea un archivo persona.ser, deserialice el objeto Persona y muestre sus atributos en la consola utilizando ObjectInputStream.
69. Modificar un objeto serializado:
    * Crea un programa que deserialice un objeto Persona, cambie uno de sus atributos y vuelva a serializar el objeto en un archivo llamado persona\_modificado.ser.
70. Escribir en un archivo binario:
    * Escribe un programa que guarde un array de enteros en un archivo binario numeros.bin utilizando DataOutputStream.
71. Leer desde un archivo binario:
    * Desarrolla un programa que lea un archivo numeros.bin e imprima los números almacenados en consola utilizando `Data