



Carátula para entrega de prácticas

Facultad de Ingeniería

Laboratorio de docencia

Laboratorios de computación salas A y B

Profesor: M.I TISTA GARCIA EDGAR

Asignatura: Programación Orientada a Objetos

Grupo: 03

No de Práctica(s): 11

Integrante(s): Hernández Diaz Sebastián

*No. de Equipo de
cómputo empleado:* 24

No. de Lista o Brigada: 19

Semestre: 2023-1

Fecha de entrega: 1 de Diciembre del 2022

Observaciones:

CALIFICACIÓN: _____

OBJETIVOS

OBJETIVO: Implementar el intercambio de datos (lectura y escritura) entre fuentes externas (archivos y/o entrada y salida estándar y un programa (en un lenguaje orientado a objetos).

OBJETIVO DE CLASE: Conocer las clases básicas para el manejo de archivos que implementa el lenguaje de programación utilizado en el curso

EJERCICIOS PROPUESTOS

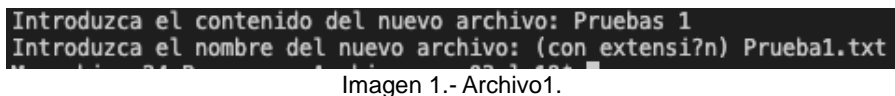
Ejercicio 1.- Pruebas con archivos

Lo primero que se pide es contestar las siguientes preguntas:

a.- ¿Qué hace el programa?

En este ejercicio se nos dieron 6 programas distintos, en el primer ejercicio se utilizan 2 variables para el scanner, además de que se crea una variable de tipo "PrintWriter", después de esto le pide al usuario que ingrese contenido a un archivo mediante un bloque try-catch se le pide el nombre de un archivo junto con su extensión la cual se obtendrá con el scanner pero se usara para crear un objeto de tipo PrintWriter, se imprimirá el texto que ingresemos y después se cerrara el archivo, en caso de obtener la excepción de "FileNotFoundException" se mostrara un mensaje de error.

Este es el resultado de la ejecución del programa:



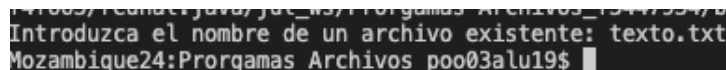
```
Introduzca el contenido del nuevo archivo: Pruebas 1
Introduzca el nombre del nuevo archivo: (con extensi?n) Prueba1.txt
```

Imagen 1.- Archivo1.

Donde no se imprime nada en pantalla, pero se crea un nuevo archivo con el texto ingresado.

El segundo archivo "Archivos2" de igual forma se crea un scanner y un tipo de dato PrintWriter, en una variable String se almacena un texto y mediante un bloque try-catch se pedirá el nombre de un archivo que debe de existir si este archivo existe se agregara el contenido al archivo se imprimirá y después se cerrara el archivo, si hay una excepción de tipo "IOException" se imprimirá el mensaje de error.

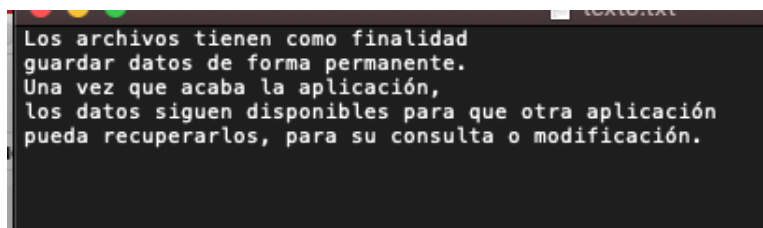
Este es el resultado de la ejecución del programa:



```
Introduzca el nombre de un archivo existente: texto.txt
```

Imagen 2.-Archivos 2.

Al ejecutar este programa con un archivo ya existente se agrega el texto almacenado en la variable al final del archivo, de la siguiente manera:



```
Los archivos tienen como finalidad
guardar datos de forma permanente.
Una vez que acaba la aplicación,
los datos siguen disponibles para que otra aplicación
pueda recuperarlos, para su consulta o modificación.
```

Imagen 3.- Agregar.

Este es el archivo previo a la modificación, y este es el resultado del archivo después de la modificación:

```

Los archivos tienen como finalidad
guardar datos de forma permanente.
Una vez que acaba la aplicación,
los datos siguen disponibles para que otra aplicación
pueda recuperarlos, para su consulta o modificación.
Append para el archivo

```

Imagen 4.- Append.

El tercer archivo "Archivos3" creará 3 variables, una para el scanner de entrada de datos, otra para un string y una última de tipo Scanner, mediante un bloque try-catch se pedirá el nombre del archivo para que se almacene en una variable y mientras esta aun tenga líneas para imprimir entonces se imprimirá en pantalla su contenido, al terminar de imprimir se cerrará el archivo y si ocurre una excepción "FileNotFoundException" entonces se mostrará un mensaje de error.

Este es el resultado de la ejecución del código:

```

Introduzca el nombre del archivo: texto.txt
Los archivos tienen como finalidad
guardar datos de forma permanente.
Una vez que acaba la aplicaci??n,
los datos siguen disponibles para que otra aplicaci??n
pueda recuperarlos, para su consulta o modificaci??n.
Append para el archivo

```

Imagen 5.- Archivos 3.

Se puede ver que recibe el nombre de un archivo y después imprime todo el contenido.

El cuarto archivo "Archivos4" crea una variable para obtener datos del usuario, otra variable de tipo ObjectOutputStream, además de otras 2 de tipo "TestObject" en los cuales se mandarán como parámetros unas variables, la primera será un "id" de tipo entero, la segunda un String y la tercera una variable de tipo double, la última variable que se crea es una de tipo String, se le pedirá al usuario el nombre del archivo y mediante un bloque try-catch se creará un archivo y se le agregarán los objetos creados con el tipo de dato "TestObject" finalmente se cerrará el archivo, en caso de ocurrir una excepción se imprimirá el mensaje correspondiente.

Esta es la ejecución del programa:

```

Introduzca el nombre del archivo: texto.txt
Mozambique24:Prorgamas Archivos poo03alu19$

```

Imagen 6.- Archivos 4.

En este caso simplemente pide el nombre del archivo, para después mediante el código agregar los TestObject al archivo, al abrir el archivo se vería el siguiente contenido:

```

Isr
TestObjectZ"/öçIidDnumberLtexttLjava/lang/String;xp@tprueba 1sq~@fffffft
Cadena texta

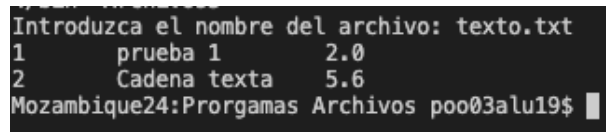
```

Imagen 7.- TestObject.

Es algo que no se puede leer por el tipo de codificación que posee.

El quinto archivo “Archivos5” crea una variable para la obtención de datos por parte del usuario, otra variable de tipo `ObjectInputStream` y 2 variables de tipo “`TestObject`”, se le pedirá al usuario el nombre del archivo, dentro de un bloque try-catch se accederá al archivo mencionado, después se leerán los objetos, al terminar estas operaciones el archivo se cerrara si encuentra una excepción de tipo “`IOException`” entonces mostrara un mensaje de error y pasara lo mismo si ocurre una excepción de tipo “`FileNotFoundException`”.

Esta es la ejecución del código:



```
Introduzca el nombre del archivo: texto.txt
1      prueba 1      2.0
2      Cadena texta  5.6
Mozambique24:Prorgamas Archivos poo03alu19$
```

Imagen 8.- Archivos 5.

En este caso tambien se agregan variables `TestObject`, pero se imprimen mediante el método `display` que se vio con anterioridad.

En este ejercicio se requiere de un archivo ya existente por lo que se usó el archivo “texto.txt” el cual fue modificado con estos objetos agregados los cuales se agregaron de manera correcta pues al usar el método `display` se imprimen de manera correcta, pero si se quiere abrir este archivo no se podrá leer por el tipo de codificación.

Por último, tenemos la clase “`TestObject`” la cual implementa la interfaz `Serializable`, dentro de esta hay 3 atributos 2 de ellos private las cuales son `id` de tipo `int` y `text` de tipo `String`, el tercer atributo es público, `number` de tipo `double`, dentro del constructor se reciben estos 3 parámetros y se almacenan, tiene un método “`display`” el cual no retornara nada, dentro de este método se imprimirán los atributos que se mencionaron con anterioridad.

b.- ¿Qué tipo de archivos se utilizan? (texto, binario, objetos)

En los programas que se vieron se usaron archivos de texto, al agregar cadenas como se vio en los primeros ejercicios, además de que estas ya tenían texto con anterioridad y tambien se usaron archivos de objetos pues se agregó contenido al archivo de un tipo de objeto el cual era `TestObject` con sus atributos y métodos, dentro de la consola estos datos almacenados en el archivo se pueden leer de manera correcta pero no se pueden leer al abrir el archivo pues no es una codificación que se pueda leer.

c.- ¿Qué clases de la jerarquía de archivos utiliza?

Se uso la clase `PrintWriter` la cual hereda de la clase `Writer` con la cual se pudo escribir de manera sencilla en los archivos de texto.

Se uso la clase `FileWriter` la cual tambien hereda de la clase `Writer`, esta nos permite escribir en un archivo de texto plano.

Se uso la clase FileReader la cual hereda de la clase Reader, con esta clase se pudo obtener los caracteres ingresados para la obtención de cierto archivo en la clase "Archivos3".

Se uso la clase FileOutputStream la cual hereda de la clase OutputStream la cual crea y escribe un flujo de bytes en un archivo de texto plano.

Se uso la clase FileInputStream la cual hereda de la clase InputStream, esta permite leer flujos de bytes desde un archivo de texto plano.

1.2. Explica las diferencias importantes entre Archivos1, Archivos3 y Archivos5

En la implementación del ejercicio de Archivos1 se solicita agregar contenido a un archivo, en caso de que el archivo no exista entonces se creara 1 con ese nombre y con el contenido propuestos, en caso de que el archivo exista se eliminara lo que haya en el documento y se escribirá lo mencionado en el inicio del programa.

En el ejercicio de Archivos3 en este caso el archivo debe de existir previamente porque lo que hará este código es leer la información que contiene el archivo, no agrega ni modifica nada.

En el ejercicio de Archivos5 en este caso tambien se pide un archivo existente pues si se ingresa uno que no exista ocurrirá una IOException, y no solo debe de existir, sino que su contenido debe de ser objetos de la clase TestObject pues este método lo que hace es tomar un archivo con esos objetos y mediante un método poder imprimir sus valores.

En conclusión, la clase de Archivos1 pide un archivo para sobrescribir contenido y si el archivo no existe lo crea, en la clase Archivos3 solo lee e imprime lo de un archivo ya existente, y en la clase Archivos5 se necesita de un archivo que tenga almacenados objetos de TestObject para poder leerlos y mostrarlos en pantalla.

Ejercicio 2. Trabajando archivos de texto

En este ejercicio se nos pide que mediante un menú el usuario pueda crear, abrir o editar un archivo de texto plano para esto lo primero que se hace es mostrar este menú mediante un bucle do-while, en el caso de crear archivo se le pedirá la ruta en la cual desea crear este archivo, y se usara la ruta para crear un objeto de tipo file, mediante un bloque try-catch se creara este archivo con la variable PrintWriter y se cerrara el archivo.

En la opción de abrir el archivo se le pedirá de igual forma la ruta del archivo, en un bloque try-catch se usará la variable BufferedReader para poder acceder a los datos del archivo, en una variable String se almacenará lo que se lea del archivo con el método "readLine" y se imprimirá en pantalla, finalmente se cierra el archivo.

En la opción de editar se le pedirá la ruta del archivo, tambien se le solicitará el contenido a agregar, para que mediante un bloque try-catch y la variable PrintWriter se agregue contenido al archivo asegurándonos de que en los parámetros del método FileWriter tenga un true para agregar el contenido.

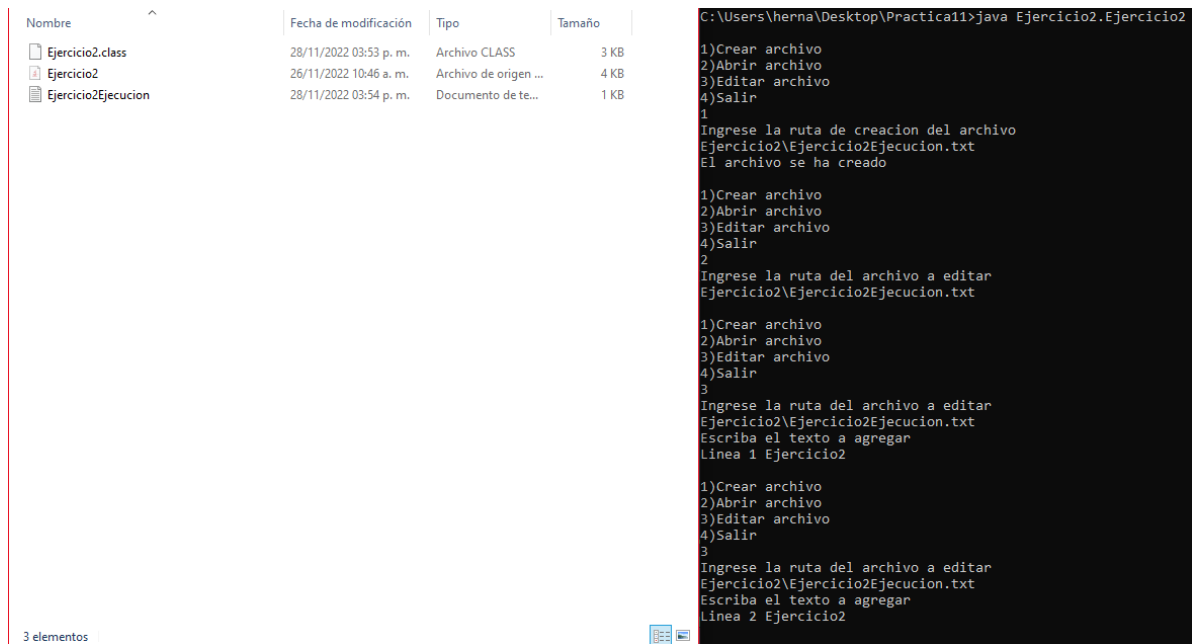


Imagen 9.- Creación de archivo.

En esta parte de la ejecución primero se ejecuta el código, aparece el menú y se crea el archivo dando la ruta y el nombre de este, después se vuelve a presentar el menú y se selecciona la opción de abrir, pero aún no hay nada en el documento por lo cual no se imprime nada, pero después se agrega contenido mediante la opción 3, por lo que al volver a abrir el documento se muestra lo siguiente:

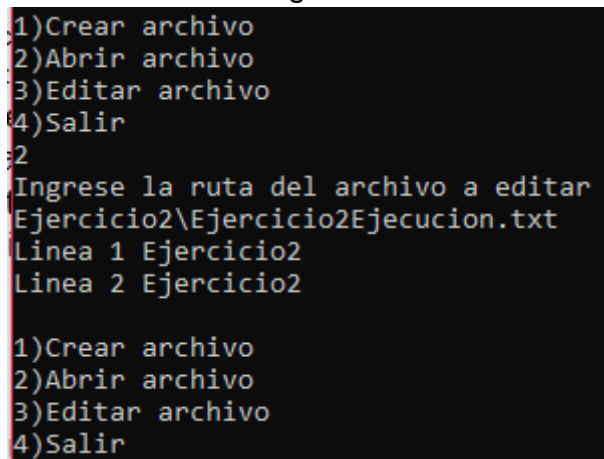


Imagen 10.- Agregar Contenido.

En esta parte el archivo ya tenía contenido por lo cual al abrirlo se muestra el mismo.

Ejercicio 3. Otro tipo de archivos

a.- Investiga brevemente para que sirven o en qué se aplican los archivos binarios y los de objetos.

Los archivos binarios son archivos normales que contienen información que el sistema puede leer. Podrían ser archivos ejecutables que indicaran al sistema que ha de realizar un trabajo. Los mandatos y los programas se almacenan en archivos binarios ejecutables. Los programas de compilación especial convierten texto ASCII en código binario.

Estos tienen como propósito el almacenamiento y procesamiento en ordenadores.

Los archivos de objeto (.obj) son archivos intermedios generados por el compilador antes de crear un ejecutable. El archivo de objeto consiste de una tabla de símbolos.

Es un archivo de computadora que contiene código de objeto, es decir, salida de código de máquina de un ensamblador o compilador. El código objeto suele ser reubicable y no suele ser directamente ejecutable.

Funcionan como la interfaz lógica entre los procesos del kernel y el modo de usuario y los datos de archivo que residen en el disco físico.

b.- Elabora un programa que pueda convertir archivos de texto plano a binarios y viceversa.

Para la creación de este código lo primero que se presenta es un menú donde se le da la opción al usuario de que escoja si quiere convertir un archivo de texto a uno binario o un archivo binario a uno de texto, cabe aclarar que se necesita de que el archivo de texto o binario estén creados para poder ser transformados.

Para empezar se le pide la ruta del archivo, después se le pide que escriba la ruta en donde se guardara el archivo transformado, enseguida se abre el flujo de escritura al archivo mediante FileOutputStream, y mientras el archivo a convertir, en este caso de texto aun tenga líneas se leerán estas y con la función write se irán agregando al archivo binario, en cuanto se termine de leer el archivo y de agregar el contenido al archivo binario se cerrara el flujo y los archivos.

Si se selecciona la opción de binario a texto de igual forma se le pide que agregue la ruta de el archivo binario a leer y la ruta a guardar el archivo de texto, con variables de FileWriter, BufferedWriter, FileInputStream y demás, se comienza a recuperar el texto de el archivo binario, este se transforma a String y se va agregando al archivo que, de texto, finalmente los flujos y archivos se cierran.

Para la ejecución de este código se tiene creado el siguiente archivo:

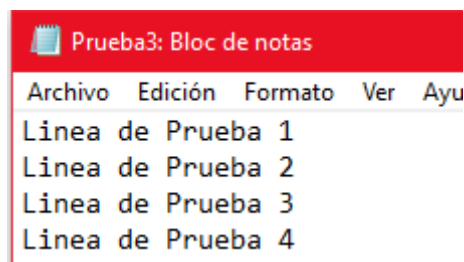


Imagen 11.- Archivo.

Primero se ejecutará el código dando este archivo para ser transformado a binario y después ese binario se transformará a texto para verificar que el contenido sea el mismo

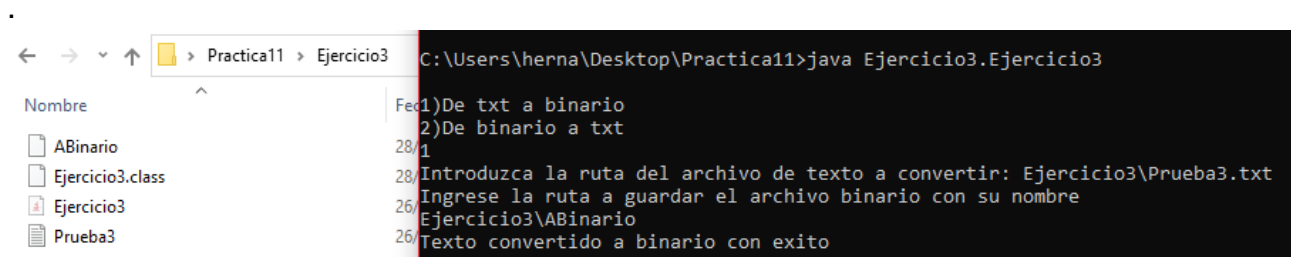


Imagen 12.- Binario.

Esta primera ejecución se dio el archivo de texto y se creó el archivo binario.

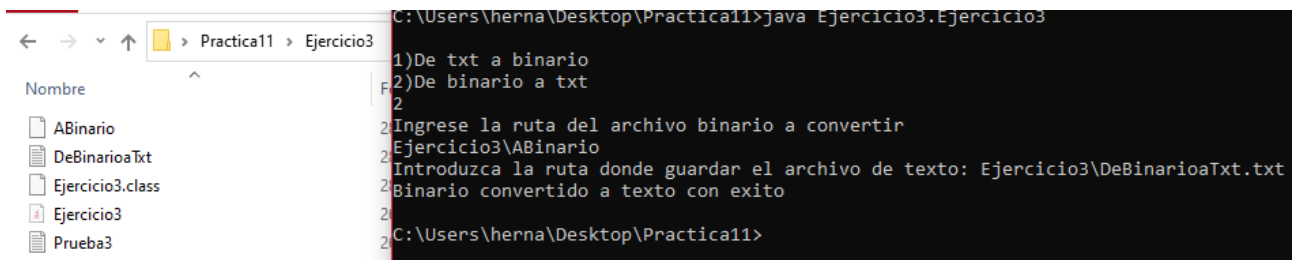


Imagen 13.- De binario a texto.

En esta segunda ejecución se transformo el archivo binario a texto y si abrimos el archivo de texto se puede ver lo siguiente:

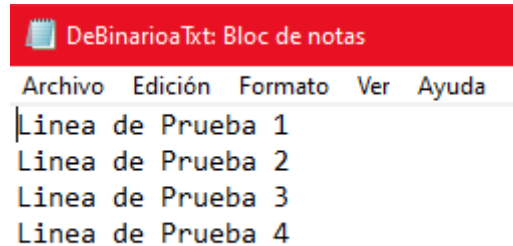


Imagen 14.- Recuperado.

Como se puede ver en este archivo el texto original fue recuperado y se transformo de manera correcta por lo que pasamos de un archivo de texto a uno binario y de este binario a uno de texto sin alterar el contenido.

Ejercicio 4. Archivos de Objetos

En este ejercicio se nos pide que con la clase Computadora que se hizo en la practica 4 se elabore un programa que con archivos el usuario pueda consultar en el archivo las computadoras existentes y agregar nuevas, actualizar datos o eliminar algunas, para de esta forma crear un archivo de objetos.

Para esto lo primero que se debe de hacer es que la clase Computadora implemente la interfaz "Serializable" para que pueda ser transformada y almacenada.

Después de esto modifique el archivo de la practica 4 para que me retorne una lista de computadoras creadas.

En el archivo "Ejercicio4" el cual será el principal para la ejecución de todo, se crea un ArrayList de computadoras para que se vayan almacenando en este mismo, se le darán las opciones al usuario de consultar las computadoras en el archivo, agregar por primera vez, actualizar datos, agregar a un archivo existente y eliminar alguna computadora del archivo, en el primer caso se le pedirá la ruta del archivo y mediante FileInputStream y ObjectOutputStream se leerá el archivo, se comenzara a leer el archivo y se imprimirá mediante un bucle while esto con la función de "MostrarDatos" que se implemento en la clase Computadora, después de leer todas las computadoras se cierra el archivo.

Para la opción 2 de agregar por primera vez se pide la ruta del archivo donde se va a guardar, después se manda a llamar a la función main del archivo de la practica 4, esto para que el usuario tenga la opción de agregar en un ArrayList todas las computadoras que necesite, se podrán modificar internamente, eliminar y todo lo que se podía en el

ejercicio de la practica 4, pero estas computadoras solo se guardaran en el archivo al terminar con esas operaciones, esto porque al terminar con las operaciones se retorna el ArrayList con las computadoras a agregar, y mediante FileOutputStream y ObjectOutputStream se irán agregando computadora a computadora del ArrayList al archivo, finalmente se cierra el flujo y las computadoras quedan almacenadas.

En la opción de actualizar los datos de igual forma se llama al main, se le pedirá la ruta del archivo para después leer las computadoras, se le pedirá la computadora a modificar y con los métodos hechos en la practica 4 se modificará esa computadora para después de la modificación ser nuevamente ingresados al archivo.

En la opción de agregar al archivo existente se le pedirá la ruta donde se encuentra el archivo, se llamara al método main de la practica 4 para que pueda agregar tantas computadoras a un ArrayList como necesite agregar a el archivo, después de terminar de agregar las computadoras estas se agregaran mediante el uso de FileOutputStream la cual deberá de tener "true" como parámetro para que se puedan agregar las computadoras y de una clase llamada "AgregarContenido" estas se escribirán en el documento, la clase "AgregarContenido" lo que hace es heredar de ObjectOutputStream aquí sobrescribirá su método writeStreamHeader el cual dentro tendrá el método reset, esto porque si no se hace así cuando se intenta agregar un elemento a un archivo ya existente se agrega de manera incorrecta y el archivo ya no se puede leer de manera correcta es por eso que debemos de sobrescribir este método para que se puedan agregar elementos a los archivos de manera correcta, al finalizar esto se cierran los flujos.

En la opción de eliminar se manda a llamar de igual manera al main del archivo "Practica4" pero en este caso en la operación de eliminar, primero se le pedirá la ruta del archivo donde están las computadoras, estas se recuperarán del archivo y se mostraran en pantalla, se le pedirá el índice de la computadora a eliminar y se eliminara del ArrayList para después ser devueltas al archivo sin la computadora eliminada.

La ejecución del código es la siguiente:

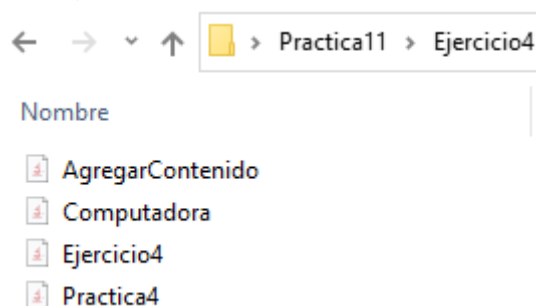


Imagen 15.- Directorio.

Este es el directorio donde todo se creará.

```
C:\Users\herna\Desktop\Practica11>java Ejercicio4.Ejercicio4
1)Consultar Computadoras en el archivo
2)Agregar Primera vez
3)Actualizar Datos
4)Agregar al archivo
5)Eliminar
```

Imagen 16.- Menú.

Esta es la ejecución del código donde lo primero que se muestra es el menú, en este caso no tenemos un archivo por lo cual seleccionaremos la opción 2 para que sea creado.

```
1)Consultar Computadoras en el archivo
2)Agregar Primera vez
3)Actualizar Datos
4)Agregar al archivo
5)Eliminar
2
Introduzca la ruta del archivo: Ejercicio4\Computadoras.txt
Estas son las opciones de la lista de computadoras internas
Por lo que no modifica las del archivo

Presiona 1) Para Crear una computadora
Presiona 2) Para Modificar una computadora
Presiona 3) Para Ver computadoras actuales
Presiona 4) Para Eliminar una computadora
Presiona 5) Terminar
```

Imagen 17.- Opción 2.

Al seleccionar la opción 2 se nos pide la ruta a guardar el archivo y también se muestra un mensaje donde se le hace saber al usuario que en ese momento está trabajando con una lista interna de computadoras por lo cual hasta que termine de agregar estas el archivo no se modificará.

```
Elegiste crear una nueva computadora
Dime el nombre del propietario
Jesus
Dime la marca del procesador
Intel
Dime la cantidad de memoria
512
Dime la cantidad de Ram
8
Dame la cantidad de watts de la fuente de poder
400
Dame la marca de la placa madre
Asus
Estas son las opciones de la lista de computadoras internas
Por lo que no modifica las del archivo

Presiona 1) Para Crear una computadora
Presiona 2) Para Modificar una computadora
Presiona 3) Para Ver computadoras actuales
Presiona 4) Para Eliminar una computadora
Presiona 5) Terminar
5
C:\Users\herna\Desktop\Practica11>
```

Imagen 18.- Agregar nuevas.

Las computadoras que se agregaron fueron 3 pero solo se muestra la última creada y cuando el usuario decide finalizar de agregar presiona la opción 5 por lo cual el programa termina con las computadoras agregadas en el archivo.

```

1)Consultar Computadoras en el archivo
2)Agregar Primera vez
3)Actualizar Datos
4)Agregar al archivo
5)Eliminar
1
Introduzca la ruta del archivo: Ejercicio4\Computadoras.txt

Indice de la computadora: 1
El propietario de la computadora es: Sebastian
La marca del procesador es: Ryzen
La cantidad de memoria es: 1024
La cantidad de memoria ram es: 16
La cantidad de watts de la fuente de poder son: 650
La marca de la placa madre es: Asus

Indice de la computadora: 2
El propietario de la computadora es: Carlos
La marca del procesador es: Intel
La cantidad de memoria es: 1024
La cantidad de memoria ram es: 32
La cantidad de watts de la fuente de poder son: 800
La marca de la placa madre es: Gigabyte

Indice de la computadora: 3
El propietario de la computadora es: Jesus
La marca del procesador es: Intel
La cantidad de memoria es: 512
La cantidad de memoria ram es: 8
La cantidad de watts de la fuente de poder son: 400
La marca de la placa madre es: Asus

```

Imagen 19.- Opción 1.

En este caso se seleccionó la opción 1 para poder visualizar los datos del archivo, primero se pide la ruta en donde está el archivo, y entonces comienza a leer y mostrar las computadoras almacenadas en el archivo.

```

1)Consultar Computadoras en el archivo
2)Agregar Primera vez
3)Actualizar Datos
4)Agregar al archivo
5)Eliminar
3
Elegiste Modificar una computadora

Introduzca la ruta del archivo para eliminar Computadora: Ejercicio4\Computadoras.txt
El archivo se leyó
Dame el índice de la Computadora a modificar
3
Opciones de modificación:
Presiona 1) Para Modificar El Procesador, Memoria y Ram
Presiona 2) Para Modificar La Fuente de Poder y Placa Madre
Presiona 3) Para Modificar El Procesador y Ram
Presiona 4) Para terminar las modificaciones
3
Dime la marca del procesador
Ryzen
Dime la cantidad de Ram
16
Opciones de modificación:
Presiona 1) Para Modificar El Procesador, Memoria y Ram
Presiona 2) Para Modificar La Fuente de Poder y Placa Madre
Presiona 3) Para Modificar El Procesador y Ram
Presiona 4) Para terminar las modificaciones
4

```

Imagen 20.- Opción 3.

Esta es la muestra de la tercera opción donde se pide modificar datos del archivo para esto pide la ruta del archivo y el índice de la computadora a modificar, después se le dan las opciones que puede hacer en las modificaciones, por lo cual selecciona 1 y se comienza a modificar después de terminar selecciona la opción que termina con las modificaciones y el archivo se modifica con los nuevos datos, el correcto funcionamiento se mostrara mas adelante cuando se vuelva a imprimir el archivo y se vean las modificaciones.

```
1)Consultar Computadoras en el archivo
2)Agregar Primera vez
3)Actualizar Datos
4)Agregar al archivo
5)Eliminar
4
Introduzca la ruta del archivo: Ejercicio4\Computadoras.txt
Estas son las opciones de la lista de computadoras internas
Por lo que no modifica las del archivo

Presiona 1) Para Crear una computadora
Presiona 2) Para Modificar una computadora
Presiona 3) Para Ver computadoras actuales
Presiona 4) Para Eliminar una computadora
Presiona 5) Terminar
1

Elegiste crear una nueva computadora

Dime el nombre del propietario
Daniel
Dime la marca del procesador
Ryzen
Dime la cantidad de memoria
512
Dime la cantidad de Ram
16
Dame la cantidad de watts de la fuente de poder
500
Dame la marca de la placa madre
Gigabyte
```

Imagen 21.- Agregar al archivo.

Esta la opción de agregar al archivo, primero se pide la ruta del archivo y entonces se muestra el menú para que pueda agregar las computadoras necesarias, en este caso solo se agregara 1 que es la que se ve en pantalla.

```
1)Consultar Computadoras en el archivo
2)Agregar Primera vez
3)Actualizar Datos
4)Agregar al archivo
5)Eliminar
5

Elegiste Eliminar una computadora

Introduzca la ruta del archivo para eliminar Computadora: Ejercicio4\Computadoras.txt

Indice de la computadora: 1
El propietario de la computadora es: Sebastian
La marca del procesador es: Ryzen
La cantidad de memoria es: 1024
La cantidad de memoria ram es: 16
La cantidad de watts de la fuente de poder son: 650
La marca de la placa madre es: Asus

Indice de la computadora: 2
El propietario de la computadora es: Carlos
La marca del procesador es: Intel
La cantidad de memoria es: 1024
La cantidad de memoria ram es: 32
La cantidad de watts de la fuente de poder son: 800
La marca de la placa madre es: Gigabyte
```

Imagen 22.-Opción 5.

Esto es lo que se muestra cuando se selecciona la opción 5, primero se pide la ruta del archivo, después se muestran 1 por 1 en este caso solo se ven 2 pero recordemos que tenemos 4 en el archivo.

```
Dame el indice de la Computadora a eliminar
1
La computadora fue eliminada.
```

Imagen 23.- Eliminar.

Al finalizar de mostrar las computadoras se le pide el índice de la computadora a eliminar y se elimina del archivo.

Por último, volveremos a imprimir el archivo para que se vea que se elimino la primera computadora, se verán los cambios que se hizo a la computadora 3 y tambien que se agregó la computadora 4.

```
1)Consultar Computadoras en el archivo
2)Agregar Primera vez
3)Actualizar Datos
4)Agregar al archivo
5)Eliminar
1
Introduzca la ruta del archivo: Ejercicio4\Computadoras.txt

Indice de la computadora: 1
El propietario de la computadora es: Carlos
La marca del procesador es: Intel
La cantidad de memoria es: 1024
La cantidad de memoria ram es: 32
La cantidad de watts de la fuente de poder son: 800
La marca de la placa madre es: Gigabyte

Indice de la computadora: 2
El propietario de la computadora es: Jesus
La marca del procesador es: Ryzen
La cantidad de memoria es: 512
La cantidad de memoria ram es: 16
La cantidad de watts de la fuente de poder son: 400
La marca de la placa madre es: Asus

Indice de la computadora: 3
El propietario de la computadora es: Daniel
La marca del procesador es: Ryzen
La cantidad de memoria es: 512
La cantidad de memoria ram es: 16
La cantidad de watts de la fuente de poder son: 500
La marca de la placa madre es: Gigabyte
```

Imagen 24.- Impresión final.

Como se puede ver en esta impresión del archivo se eliminó la primera computadora de forma correcta, también se pueden apreciar los cambios en la computadora 3 que en este caso ahora es la 2 y la computadora agregada con anterioridad todo de manera correcta.

CONCLUSIONES

Los objetivos de la práctica se cumplieron de manera correcta, estos eran conocer y aplicar la forma de manejar archivos, como se debían de crear un flujo para que los datos pudieran almacenarse de forma correcta, además de poder aplicar esto en el lenguaje java lo cual con los ejercicios se cumplió de manera correcta.

Los ejercicios mostrados en la práctica durante el ejercicio 1 fueron de mucha ayuda pues eran ejemplos concisos sobre como es que se trabaja con los archivos, como se abren los flujos para que sirven las clases, como se trabaja con archivos de texto y también con archivos de objetos, eran ejemplos que requerían de un buen análisis y entender paso a paso que es lo que pasaba para de esta forma poder hacer los demás ejercicios que se nos pedían, en archivos siempre es un tema algo difícil porque en ocasiones el programa solo funciona si está en cierta carpeta y al moverlo o enviarlo puede que el programa deje de funcionar pero en este caso de los ejercicios se les pedía la ruta al usuario por lo cual no debería de haber problema alguno en la ejecución de los mismos.

En los ejercicios que eran hechos por nosotros fueron algo difíciles de realizar, pues pasar un archivo de texto a uno binario fue difícil de hacer, además de que si se quería buscar ayuda en internet solo tomaban objetos y los transformaban en binario mas no texto, pero gracias a ello pude darme una idea y resolver el ejercicio de una buena manera.

Un buen ejercicio fue el segundo pues este era para darnos una idea del tema, conocerlo mas y entenderlo de verdad para poder trabajar primero con archivos de texto plano y después poder trabajar con los que eran binarios.

El ultimo ejercicio fue el más difícil pero a su vez el que mas nos ayudo a entender el tema y todo el concepto, para empezar porque se debía de recuperar un ejercicio de practicas pasadas y trabajar sobre este intentando no tener que hacerlo de nuevo desde cero pero gracias a que ese ejercicio se realizo de manera correcta solo era modificarlo un poco para que pudiera trabajar de mejor manera en el ejercicio propuesto, al momento de hacer que la clase Computadora implementara la clase Serializable ocurrió un error pues como había un Scanner en esta clase ocurría un error, pero al volver estática esa variable el error se solucionó, este ejercicio fue el más difícil pues ningún ejercicio propuesto nos enseñó a agregar contenido a un archivo la cual fue la parte más difícil de código pero con un poco de investigación esto se logro hacer de manera correcta y el ejercicio fue resuelto.

Las ventajas de esta practica son muchas pues el manejo de archivos es muy importante en todo momento por lo cual conocer lo básico sobre este tema es muy bueno, aunque sería mejor si se llegara a profundizar mejor.

La practica en general fue muy buena, contribuye al entendimiento y aprendizaje del tema además de que nos hace investigar por nuestra cuenta para darnos cuenta que hay mucho mas sobre este tema y que debemos de profundizar por nuestra cuenta.