	Carátula para entrega de prácticas
Facultad de Ingeniería	Laboratorio de docencia

Laboratorios de computación
Salas A y B

Profesor: Ing. José Antonio Ayala Barbosa

Asignatura: Programación Orientada a Objetos

Grupo: 1

No de Práctica(s): Práctica 11 Manejo de archivos

Integrante(s): José Luis Arroyo Chavarría
Francisco Moisés Barrera Guardia
Juan Manuel Peralta Rodríguez
Rodrigo Daniel Reséndiz Cruz

No. de Lista o Brigada: Equipo I

Semestre: 3

Fecha de entrega: 14/1/2021

Observaciones:

CALIFICACIÓN: _____

I. 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).

II. Introducción

Durante el desarrollo de un proyecto es de suma importancia que uno como programador guarde los datos adquiridos en un formato de documento que sea de fácil acceso al público en general, o bien a personas que no saben a profundidad sobre la finalidad del programa, de igual forma, de ser necesario, se deben de guardar los datos obtenidos para no perderlos y así poder consultarlos sin problema, es debido a esto que es necesaria la implementación de archivos al momento de programar y por lo general son archivos con la extensión "txt".

Los distintos tipos de ficheros se diferencian por las clases que usaremos para representarlos y manipularlos, además, de que dentro de un programa se pueden identificar y guardar más fácil estos ficheros, para así guardar al salir del programa cualquier información que nos pueda ser útil

¿Cómo manipular archivos en java?

Para controlar archivos en java ay dos opciones:

1) Si se desea procesar datos de un archivo existente se debe:
   a) Abrir el archivo.
   b) Leer o introducir los datos en las variables un elemento a la vez.
   c) Cerrar el archivo cuando se termine de trabajar con él.
2. Para transferir algunos datos de ciertas variables a un archivo, se debe:
   a) Abrir el archivo.
   b) Extraer o escribir los elementos en la secuencia requerida.
   c) Cerrar el archivo cuando se termine de trabajar con él.

Y para poder controlar archivos desde java es necesaria la instrucción "import java.io" que se complementa con la clase File.

Clase file

Java permite acceder a los archivos almacenados en el sistema de

archivos local utilizando la clase File, la cual es una representación abstracta, es independiente del sistema de operación o del hardware en la cual JVM se está ejecutando, los objetos de la clase file son inmutables, en otras palabras, una vez que se crea un objeto File para representar una ruta particular, no se puede modificar para apuntar a otra, solo es posible asignarle una estancia diferente de la clase File.

Algunos de los métodos de la clase File son:

- createNewFile()
- delete()
- exists()
- isDirectory()
- isFile()
- list()
- mkdir()
- renameTo()
- getName()
- getPath()
- length()

El principal uso de la clase File es para la creación de nuevos archivos vacíos, la búsqueda de archivos, borrar archivos y hacer directorios.

III. Desarrollo

En esta práctica se hicieron actividades que tiene que ver con el manejo de archivos en java, a continuación se enlistan las actividades.

1. ***Creación y validación de un documento***

```
1  package pool1;
2  import java.io.File;
3  import java.io.FileInputStream;
4  import java.io.FileOutputStream;
5  import java.io.IOException;
6  public class POO11 {
7      public static void main(String[] args){
8          System.out.println("1++++++++++++++++++++++++++++++");
9
10         try{
11             File archivo = new File("miArchivo.txt");
12             System.out.println(archivo.exists());
13             boolean seCreo = archivo.createNewFile();
14             System.out.println(seCreo);
15             System.out.println(archivo.exists());
16         }catch(IOException e){}
```

Ilustración 1.- Código de la primera actividad

En la primera actividad se crea un archivo con new file y después utilizamos archivo.exists para corroborar que el archivo se ha creado, una vez comprobado se crea una variable de tipo boolean que creara un archivo igual al anterior, como ya existe, recibirá un false y finalmente se corrobora de nuevo que exista el archivo.

```
run:
1++++++++++++++++++++++++++++++
true
false
true
```

Ilustración 2.-Ejecución de la actividad 1

En la ejecución se ve que primero imprime true porque existe el archivo, luego devuelve false porque ya se había creado el archivo y finalmente escribe true, porque el archivo creado sigue existiendo.

2. Escritura de un archivo

```
System.out.println("2++++++++++++++++++++++++++++++");
System.out.println("FileOutputStream");

FileOutputStream fos = null;
byte[] buffer = new byte[81];
int nBytes;

try{
    System.out.println("Escribe el texto a guardar");
    nBytes = System.in.read(buffer);
    fos = new FileOutputStream("fos.txt");
    fos.write(buffer, 0, nBytes);
}catch(IOException ioe){}
finally{
    try{
        if(fos != null)
            fos.close();
    }catch(IOException ioe){}
}
```

Ilustración 3.- Código de la actividad 2

La segunda actividad trato de hacer una entrada del teclado que nos permita escribirla en un documento. Para ello se declara a null el flujo de datos y después se define la cantidad de bytes que permitirá escribir nuestro programa.

Luego, en un try-catch se pide al usuario que escriba un texto a guardar, esto se va a capturar mediante un buffer del teclado, una vez que el usuario escriba algo, se utilizará el método. write para escribir el contenido que le dimos al archivo, con un finally se comprueba si el archivo está abierto, si lo está, lo cierra.

```
2++++++++++++++++++++++++++++++
FileOutputStream
Escribe el texto a guardar
Texto de ejemplo
```

Ilustración 4.- Ejecución de la actividad 2

En la ejecución se puede visualizar que el programa nos pide escribir algo, una vez escrito un texto este se guarda en el archivo de texto llamado fos.

3. Lectura de un archivo

```
System.out.println("3++++++++++++++++++++++++++");
System.out.println("FileInputStream");
FileInputStream fis = null;
try{
fis = new FileInputStream("fos.txt");
nBytes = fis.read(buffer, 0, 81);
String textoLeido = new String(buffer, 0, nBytes);
System.out.println(textoLeido);
}catch(IOException ioe){}
finally{
    try{
    if(fis != null)
        fis.close();
    }catch(IOException ioe){}
```

Ilustración 5.- Código Actividad 3

En la tercera actividad se utilizó FileInputStream para leer un archivo

anteriormente escrito, en este caso el que hicimos en la anterior actividad.

Primero creamos un flujo de datos llamado fis y lo inicializamos a nulo, luego se le atribuye el archivo anterior a dicha variable y mediante la función read leemos el archivo restringido a 81 bytes, luego se imprime el contenido de dicho archivo. Finalmente se hace lo mismo que en la anterior actividad, primero comprobamos si se cerró el archivo y de no estarlo se cierra.

```
3++++++++++++++++++++++++++
FileInputStream
Texto de ejemp
```

Ilustración 6.- Ejecución de la Actividad 3

En la ejecución se ve el texto que se escribió en el archivo con anterioridad, pero está incompleto, esto se debe a que el texto a leer excedió la cantidad de bytes permitido.

4. *implementación de file writter*

```
package bufferedReader;

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;

public class ClaseFileWriter {
    public static void main(String[] args) {
        System.out.println("4++++++++++++++++++");
        System.out.println("FileWriter");
        String texto = "";
        BufferedReader br;
        try{
            br = new BufferedReader(new InputStreamReader(System.in));
            System.out.println("Escribir texto:");
            texto = br.readLine();
            System.out.println(texto);

            FileWriter fw = new FileWriter("fw.txt");
            BufferedWriter bw = new BufferedWriter(fw);
            PrintWriter salida = new PrintWriter(bw);

            salida.println(texto);
            salida.println("Segunda Linea");
            salida.close();
        }catch(IOException ioe){}
```

Ilustración 7.-Código de la actividad 4

En la cuarta actividad se implementó la escritura de texto mediante file writer, primero declaramos una variable vacía de tipo cadena llamada texto, luego creamos un lector de buffer y lo llamamos br, después en un try-catch le pedimos al usuario que escriba un texto el cual va a ser capturado con un readline y se almacena en la variable texto antes creada.

Una vez escrito el texto se crea un archivo de texto con filewriter llamado fw.txt, luego creamos un lector de buffer y un impresor de buffer, con ello vamos a escribir el texto y luego a leerlo.

Finalmente se escribe en el archivo el texto que escribimos más una línea que se escribe directamente desde el código, como ya acabamos de hacer lo que queríamos, cerramos el archivo.

```
4++++++++++++++++++
FileWriter
Escribir texto:
texto ejemplo
texto ejemplo
```

Ilustración 8.-Ejecución de la actividad 4

En la ejecución se nos pidió que escribiéramos algo y al dar enter se nos escribió de nuevo lo que escribimos.

5. ***implementación de file reader***

```
System.out.println("5++++++++++++++++++");
System.out.println("FileReader");

try{
    FileReader fr = new FileReader("fw.txt");
    br = new BufferedReader(fr);
    System.out.println("El contenido del Archivo es:");
    String Linea = br.readLine();
    while(Linea != null){
        System.out.println(Linea);
        Linea = br.readLine();
    }
    br.close();
}catch(IOException ioe){}
```

Ilustración 9.- Código de la actividad 5

En la quinta actividad implementamos una pieza de código que nos permite leer el contenido de un archivo mediante file reader, por lo que primero creamos una variable llamada fr para leer el archivo anterior creado, luego imprimimos el contenido de este utilizando una variable de tipo cadena llamada línea mediante readline la cual nos permite después leer c ad alinea del texto mediante un while, finalmente cerramos el archivo.

```
5++++++++++++++++++
FileReader
El contenido del Archivo es:
texto ejemplo
Segunda Linea
```

Ilustración 10.- Ejecución de la Actividad 5

En la ejecución se nos muestra el contenido del archivo anteriormente hecho y escrito.

6. ***implementación de un serializador.***

```
package Serializar;
public class SerializarFecha {
    public static void main(String[] args) throws InterruptedException {
     serializador serializador = new serializador();
     Thread.sleep(5*100);
     Deserealizdor des = new Deserealizdor();
    }
```

Ilustración 11.- Código Actividad 6 (serializar fecha)

Esta parte del código es por asi decirlo, la clase donde vamos a ejecutar el código de las dos siguientes clases que conforman la actividad las cuales son el serializador y el deserializador.

```
package Serializar;

import java.io.FileOutputStream;
import java.io.IOException;
import java.io.ObjectOutputStream;
import java.util.Date;

public class serializador {
        public serializador() {
            Date d = new Date();
            System.out.println(d);
            try{
            FileOutputStream fos = new FileOutputStream("objetoDate.ser");
            ObjectOutputStream oos= new ObjectOutputStream(fos);
            oos.writeObject(d);
            oos.close();
         }catch(IOException ioe){}
    }
}
```

Ilustración 12.- Código Actividad 6 (serializador)

La anterior clase hace uso de la biblioteca java.util.date para crear una fecha correspondiente a la que

tengamos en el equipo donde se ejecuta el programa, dicha fecha se introduce a una variable recién creada llamada d.

Mediante un fileOutput stream creamos un archivo con extensión .ser y escribimos la fecha ahí., finalmente cerramos el archivo.

```
package Serializar;

import java.io.FileInputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.util.Date;

/**
 *
 * @author pipzo
 */
public class Deserealizdor {
    public Deserealizdor() throws InterruptedException{
        Date d = null;
        try{
            FileInputStream fis = new FileInputStream("objetoDate.ser");
            ObjectInputStream ois = new ObjectInputStream(fis);
            d=(Date) ois.readObject();
            ois.close();
        }catch(IOException ioe){
        }catch(ClassNotFoundException e){}
        System.out.println("Objetos deserealizado");
        System.out.println(d);
        Thread.sleep(5*100);
        d = new Date();
        System.out.println("La fecha actual es: "+d);
    }
}
```

Ilustración 13.-Código Actividad 6 (Deserializador)

En esta clase crea una variable de tipo fecha y se inicializa a nulo, luego toma el archivo anteriormente hecho por el serializador y se utilizara fileinputstream para leer un objeto, como no existe es atrapado por un catch que imprime la fecha del archivo, indica que lo deserializo y por medio de thread.sleep esperamos unos segundos antes de cambiar la fecha a otra que sea actual. Finalmente mandamos a imprimir la nueva fecha, producto de la deserialización.

```
run:
Wed Jan 13 21:35:15 CST 2021
Objetos deserealizado
Wed Jan 13 21:35:15 CST 2021
La fecha actual es: Wed Jan 13 21:35:16 CST 2021
BUILD SUCCESSFUL (total time: 1 second)
```

Ilustración 14.- Ejecución de la Actividad 6

En la ejecución se puede ver la fecha que se serializa en el archivo, luego se nos muestra la impresión del deserealizador el cual nos indica que la fecha fue cambiada por lo que imprime la fecha vieja y la nueva, la cual es de 1 segundo después.

Estas fueron todas las actividades abarcadas en la práctica de manejo de archivos.

IV. Código fuente

- ***POOP11:***

```
package poop11;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;
public class POOP11 {
    public static void main(String[] args)
throws IOException {
System.out.println("1**********
**********");
        System.out.println("FILE:");
        try{
            File archivo = new
File("miArchivo.txt");
```

```
System.out.println(archivo.exists());
            boolean seCreo = archivo.createNewFile();
            System.out.println(seCreo);
System.out.println(archivo.exists());
        }catch(IOException e){}
System.out.println("2**********
**********");
System.out.println("FileOutputStream:
");
        FileOutputStream fos = null;
        byte[] buffer = new byte[500];
        int nBytes;
        try{
            System.out.println("Escribe el
texto a guardar en el archivo:");
            nBytes = System.in.read(buffer);
            fos = new
FileOutputStream("fos.txt");
            fos.write(buffer,0,nBytes);
        }catch(IOException ioe){}
        finally{
            try{
                if(fos != null)
                    fos.close();
            }catch(IOException ioe){}
        }
System.out.println("3**********
**********");
System.out.println("FileInputStream:")
;
        FileInputStream fis = null;
        try{
            fis = new
FileInputStream("fos.txt");
            nBytes = fis.read(buffer, 0,
500);
            String textoLeido = new
String(buffer,0,nBytes);
            System.out.println(textoLeido);
        }catch(IOException ioe){}
        finally{
            if(fis != null)
                fis.close();
        }
    }
```

```
}
```

```java
package poopl1;

import java.io.File;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.IOException;

public class POOPl1 {

    public static void main(String[] args) throws IOException {

        System.out.println("1**************************************************");
        System.out.println("FILE:");
        try{
            File archivo = new File("miArchivo.txt");
            System.out.println(archivo.exists());
            boolean seCreo = archivo.createNewFile();
            System.out.println(seCreo);
            System.out.println(archivo.exists());
        }catch(IOException e){}

        System.out.println("2**************************************************");
        System.out.println("FileOutputStream:");
        FileOutputStream fos = null;
        byte[] buffer = new byte[500];
        int nBytes;
        try{
            System.out.println("Escribe el texto a guardar en el archivo:");
            nBytes = System.in.read(buffer);
            fos = new FileOutputStream("fos.txt");
            fos.write(buffer,0,nBytes);
        }catch(IOException ioe){}
        finally{
            try{
                if(fos != null)
                    fos.close();
            }catch(IOException ioe){}
        }

        System.out.println("3**************************************************");
        System.out.println("FileInputStream:");
        FileInputStream fis = null;
        try{
            fis = new FileInputStream("fos.txt");
            nBytes = fis.read(buffer, 0, 500);
            String textoLeido = new String(buffer,0,nBytes);
            System.out.println(textoLeido);
        }catch(IOException ioe){}
        finally{
            if(fis != null)
                fis.close();
        }
    }
}
```

```
run:
1**************************************************
FILE:
true
false
true
2**************************************************
FileOutputStream:
Escribe el texto a guardar en el archivo:
Primera Linea
3**************************************************
FileInputStream:
Primera Linea

BUILD SUCCESSFUL (total time: 22 seconds)
```

- ***<u>BufferedReader</u>***
- ClaseFileWriter

```
package bufferedReader;
//import java.io.*;
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
```

```
import java.io.InputStreamReader;
import java.io.PrintWriter;
public class ClaseFileWriter {
    public static void main(String[] args){
System.out.println("4**********
**********");
        System.out.println("FileWriter:");
        String texto = "";
        BufferedReader br;
        try{
            br = new BufferedReader(new
InputStreamReader(System.in));
            System.out.println("Escribir
texto:");
            texto = br.readLine();
            System.out.println(texto);
            FileWriter fw = new
FileWriter("fw.txt");
            BufferedWriter bw = new
BufferedWriter(fw);
            PrintWriter salida = new
PrintWriter(bw);
            salida.println(texto);
            salida.println("Segunda
Linea");
            salida.close();
        }catch(IOException ioe){}
System.out.println("5**********
**********");
System.out.println("FileReader:");
        try{
            FileReader fr = new
FileReader("fw.txt");
            br = new BufferedReader(fr);
            System.out.println("El
contenido del archivo es:");
            String linea = br.readLine();
            while(linea != null){
                System.out.println(linea);
                linea = br.readLine();
            }
            br.close();
        }catch(IOException ioe){}

    }
}
```

```
package bufferedReader;

//import java.io.*;
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;

public class ClaseFileWriter {

    public static void main(String[] args){
        System.out.println("4**************************************************");
        System.out.println("FileWriter:");
        String texto = "";
        BufferedReader br;
        try{
            br = new BufferedReader(new InputStreamReader(System.in));
            System.out.println("Escribir texto:");
            texto = br.readLine();
            System.out.println(texto);
            FileWriter fw = new FileWriter("fw.txt");
            BufferedWriter bw = new BufferedWriter(fw);
            PrintWriter salida = new PrintWriter(bw);
            salida.println(texto);
            salida.println("Segunda Linea");
            salida.close();
        }catch(IOException ioe){}
```

```
        System.out.println("5**************************************************");
        System.out.println("FileReader:");
        try{
            FileReader fr = new FileReader("fw.txt");
            br = new BufferedReader(fr);
            System.out.println("El contenido del archivo es:");
            String linea = br.readLine();
            while(linea != null){
                System.out.println(linea);
                linea = br.readLine();
            }
            br.close();

        }catch(IOException ioe){}

    }
}
```

```
run:
4**************************************************
FileWriter:
Escribir texto:
Primera Linea
Primera Linea
5**************************************************
FileReader:
El contenido del archivo es:
Primera Linea
Segunda Linea
BUILD SUCCESSFUL (total time: 10 seconds)
```

- ***Serializar***

❖ SerializarFecha

```
package Serializar;
public class SerializarFecha {
    public static void main(String[]
args){
        Serializador serializador = new
Serializador();
```

```
        DesSerializador des = new DesSerializador();
    }
}
```

❖ Serializador

```
package Serializar;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.ObjectOutputStream;
import java.util.Date;
public class Serializador {
    public Serializador(){
        Date d = new Date();
        System.out.println(d);
        try{
            FileOutputStream fos = new FileOutputStream("objetoDate.ser");
            ObjectOutputStream oos = new ObjectOutputStream(fos);
            oos.writeObject(d);
            oos.close();
        }catch(IOException ioe){}
    }
}
```

❖ DesSerializador

```
package Serializar;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.util.Date;
public class DesSerializador{
    public DesSerializador(){
        Date d = null;
        try{
            FileInputStream fis = new FileInputStream("objetoDate.ser");
            ObjectInputStream ois = new ObjectInputStream(fis);
            d = (Date) ois.readObject();
            ois.close();
        }catch(IOException ioe){
        }catch(ClassNotFoundException e){}
        System.out.println("Objecto Deserializado");
        System.out.println(d);
        d = new Date();
        System.out.println("La Fecha Actual es: "+d);
    }
}
```

```java
package Serializar;

public class SerializarFecha {
    public static void main(String[] args){
        Serializador serializador = new Serializador();
        DesSerializador des = new DesSerializador();
    }
}
```

```java
package Serializar;

import java.io.FileOutputStream;
import java.io.IOException;
import java.io.ObjectOutputStream;
import java.util.Date;

public class Serializador {
    public Serializador(){
        Date d = new Date();
        System.out.println(d);
        try{
            FileOutputStream fos = new FileOutputStream("objetoDate.ser");
            ObjectOutputStream oos = new ObjectOutputStream(fos);
            oos.writeObject(d);
            oos.close();
        }catch(IOException ioe){}
    }
}
```

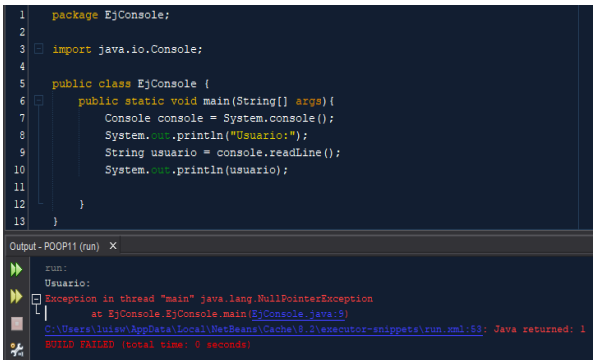
```java
package Serializar;

import java.io.FileInputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.util.Date;

public class DesSerializador{
    public DesSerializador(){
        Date d = null;
        try{
            FileInputStream fis = new FileInputStream("objetoDate.ser");
            ObjectInputStream ois = new ObjectInputStream(fis);
            d = (Date) ois.readObject();
            ois.close();
        }catch(IOException ioe){
        }catch(ClassNotFoundException e){}
        System.out.println("Objecto Deserializado");
        System.out.println(d);
        d = new Date();
        System.out.println("La Fecha Actual es: "+d);
    }
}
```

```
run:
Wed Jan 13 07:03:44 CST 2021
Objecto Deserializado
Wed Jan 13 07:03:44 CST 2021
La Fecha Actual es: Wed Jan 13 07:03:44 CST 2021
BUILD SUCCESSFUL (total time: 0 seconds)
```

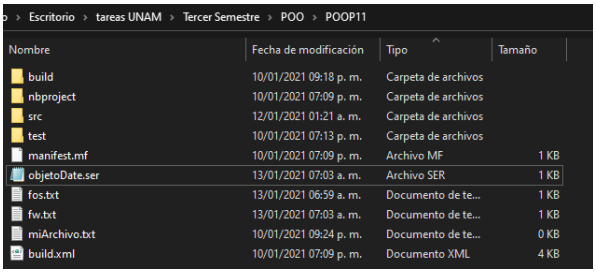
➢ ***EjConsole***

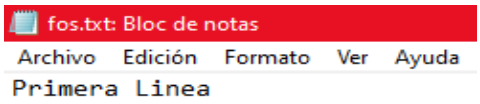
```
package EjConsole;
import java.io.Console;
```

```
public class EjConsole {
    public static void main(String[] args){
        Console console = System.console();
        System.out.println("Usuario:");
        String usuario = console.readLine();
        System.out.println(usuario);
    }
}
```



➢ Archivos (.txt)





V. Conclusión

- José Luis Arroyo Chavarría:

Al desarrollar y al concluir con lo hecho de la práctica vi los manejos de creación de los archivos de texto y a su vez la escritura de estos archivos, esto como programador puede ayudarnos para dar un mensaje al usuario si su información ingresada es correcta. Personalmente estos métodos me podrán ayudar para mi proyecto final al hacer un programa que el usuario ingrese una información y dar el mensaje de lo que ingreso.

- Francisco Moisés Barrera Guardia:

En esta práctica pude aprender a guardar información dentro de un programa, y el cómo funciona el manejo de archivos en java, aunque al principio fue un poco nuevo y pesado con la práctica se fue mejorando este ámbito

- Juan Manuel Peralta Rodríguez:

Gracias al desarrollo de esta práctica se pudo revisar a profundidad los diferentes usos de los archivos en el lenguaje de programación Java, así como su aplicación para la resolución de diferentes problemas cotidianos, de igual forma se pudieron cumplir los

objetivos previamente propuestos dúrate el desarrollo de las actividades.

- Rodrigo Daniel Reséndiz Cruz:

En esta práctica pudimos ver de forma práctica el manejo de archivos utilizando java y me complace decir que gracias a ella tengo elementos que me permitan avanzar en la elaboración del proyecto propuesto para esta materia.

VI. Referencias

- https://ingenieria.udistrital.edu.co/pluginfile.php/39198/mod_resource/content/1/7.%20Archivos%20en%20Java.pdf
- https://steemit.com/spanish/@nelioobatis/manejo-de-archivo-en-java