	Carátula para entrega de prácticas
Facultad de Ingeniería	Laboratorio de docencia

Laboratorios de computación

salas A y B

Profesor : Edgar Tista García

Asignatura: EDA 2

Grupo: 4

No. Práctica : 10

Integrante: Chagoya Gonzalez Leonardo

Celis Hernandez Ronie

No. Equipo de Cómputo: N/A

No. de Lista o Brigada: #Lista 08

Semestre: 2022-2

Fecha de entrega: 28 de abril 2022

Observaciones:

CALIFICACIÓN: _____

OBJETIVO:
El ESTUDIANTE CONOCERÁ E IDENTIFICARÁ ASPECTOS SOBRE LOS ARCHIVOS, COMO LAS OPERACIONES, EL TIPO DE ACCESO Y ORGANIZACIÓN LÓGICA.

Desarrollo:

1.Investiga en el api de Java para que sirve la clase File, y para qué sirven los siguientes métodos.

La clase File permite acceder a los ficheros, y se usa para poder crear o eliminar archivos. Dentro de los métodos que se investigaron para la clase File se encuentran los siguientes:

- *canRead() :* Devuelve un valor de tipo booleano, en caso de que el fichero se pueda leer, de acuerdo al nombre de la ruta específica.

- *canWrite() :* Devuelve un valor de tipo booleano, en caso de que se pueda escribir en el fichero.

- createNewFile(): Devuelve un valor de tipo booleano, si el archivo con nombre existe devuelve un false, en caso de que el archivo no exista en la ruta lo crea y devuelve un true

- *getName():* Devuelve el nombre del fichero instanciado mediante la clase File.

- *getPath():* Devuelve una cadena de la ruta en donde se encuentra el archivo.

- *length():* Sirve para identificar el tamaño del fichero en bytes.

2. Investiga las principales diferencias entre las clases:

FileWriter y BufferedWriter:

BufferedWriter almacena en un buffer la salida de caracteres antes de escribirlos al archivo en físico, por lo cual permite hacer más eficiente el proceso de escritura, mientras que FileWriter escribe flujos de bytes sin procesar de forma directa en el archivo.

FileReader y BufferedReader:

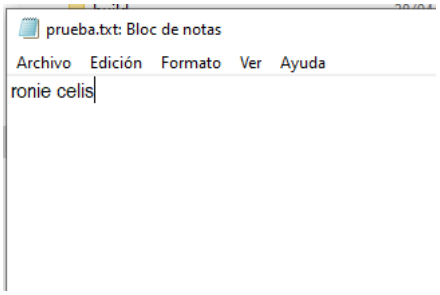
BufferedReader lee un flujo de cadenas que posteriormente se almacenarán en un buffer de un tamaño predeterminado, al poder leer por bloques bufferedReader no solo lee bytes sino cadenas, arrays o líneas, mientras que FileReader lee flujos de bytes sin procesar.

3.- ¿Cómo hace el programa proporcionado para establecer la diferencia entre agregar contenido a un archivo o sobreescribir la información anterior?

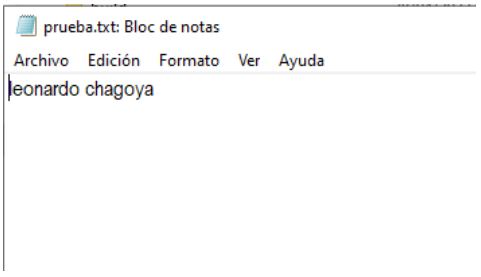
Al momento de utilizar el método escribir se le pasa como parámetros el nombre del archivo y el modo (este valor será un booleano), posteriormente dentro de escribir se instancia un objeto de tipo FileWriter si el modo es de tipo True se añade la información al archivo, en caso contrario se sobrescribe la información. A continuación se muestran evidencias de las pruebas que se hicieron:

Creación del Archivo

```
Archivos :D

1.Crear Archivo
2.Sobreescribir en el archivo
3.Añadir contenido en el archivo
4.Elimina el archivo
5.Salir

1
Ingresa el Nombre del archivo con su extension:
prueba.txt
Se creó el archivo exitosamente.

Añadir contenido:
ronie celis
```

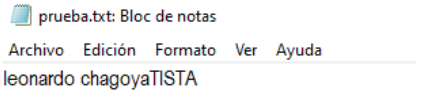


Sobreescribir Información




```
1.Crear Archivo
2.Sobreescribir en el archivo
3.Añadir contenido en el archivo
4.Elimina el archivo
5.Salir

2
Ingresa el Nombre del archivo con su extension:
prueba.txt
Añadir contenido:
leonardo chagoya
```

Añadir información




```
Ingresa el Nombre del archivo con su extension:
prueba.txt
Añadir contenido:
leonardo chagoya

1.Crear Archivo
2.Sobreescribir en el archivo
3.Añadir contenido en el archivo
4.Elimina el archivo
5.Salir

3
Ingresa el Nombre del archivo con su extension:
prueba.txt
Añadir contenido:
```

4. ¿Qué ocurre si el usuario desea agregar contenido a un archivo pero no se indica la extensión?.

Se pasa como parámetro el nombre del archivo al método escribir, lo primero que hace este método es detectar si el archivo existe en base a la instancia de un objeto llamado *archivo*, esto se hace con la ayuda del método exists de la clase File, de modo que si se ingresa el nombre del archivo sin la extensión, no podrá encontrarlo en el directorio, y por lo tanto el método exists retorna un valor False, indicando que no se encontró el archivo.

Con base al ejercicio anterior se intentó añadir más contenido al archivo prueba.txt,sin agregar la extension txt con lo cual detectó que no existe dicho archivo

```
1.Crear Archivo
2.Sobreescribir en el archivo
3.Añadir contenido en el archivo
4.Elimina el archivo
5.Salir

3
Ingresa el Nombre del archivo con su extension:
prueba
No existe el archivo.
```

5.- ¿Por qué el programa proporcionado no maneja la excepción (no tiene un try-catch) FileNotFoundException en ningún lugar, y esa excepción no ocurre cuando el usuario ingresa un archivo que no existe?

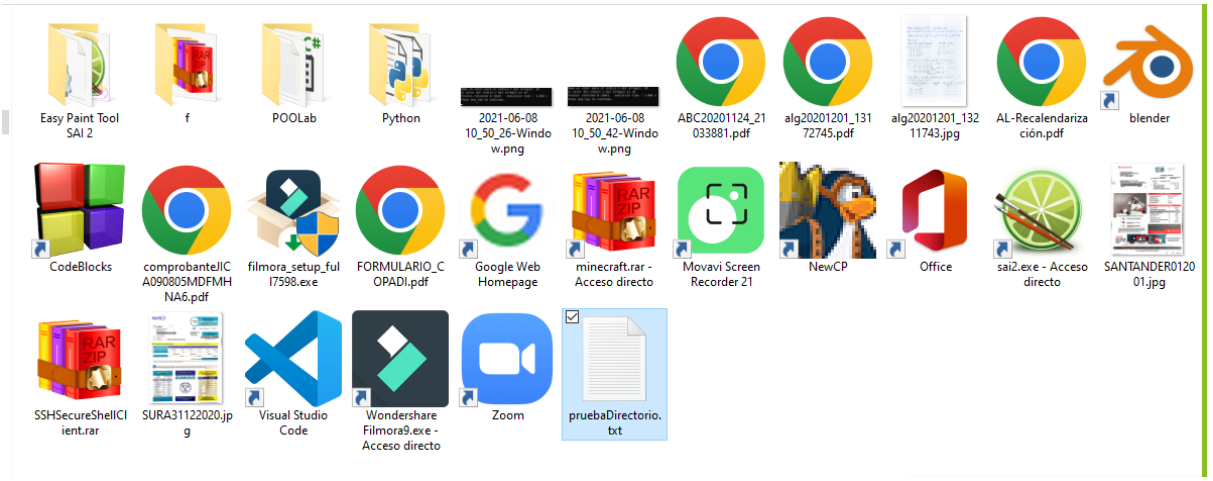
Porque en cada uno de los métodos en los cuales se opera con los archivos, se utiliza throws IOException indicando que dichos métodos pueden arrojar una excepción en la entrada o salida de datos de un fichero.

```
public void crearArchivo(String nombre) throws IOException {
    File archivo = new File (nombre);
    if (archivo.exists()){
        System.out.println("Ya existe el archivo.\n");
    } else{
        FileWriter fw = new FileWriter(archivo);
        System.out.println("Se creó el archivo exitosamente.\n");
        escribir(nombre, false);
        fw.close();
    }

}
```

6. Modifica el programa de tal manera que, al momento de crear un nuevo archivo, el usuario indique la ruta donde el lo quiera crear (por ejemplo el escritorio) Agrega en este cuestionario un recorte con el fragmento o fragmentos de código modificados y una imagen del programa corriendo donde se vea claramente que el archivo si se crea en la ruta ingresada por el usuario

Mediante el método crearArchivoDirectorio se le paso como paramtero el nombre del archivo a crear si dicho archivo existe el metodo lo detecta si no realiza ningun procedimiento, si el nombre del archivo no existe se le asigna la ruta a un string llamado ruta (aqui se implementa una concatenación de cadenas pero no nos dio tiempo) posteriormente crea un nuevo archivo

```
public void crearArchivoDirectorio(String nombre)throws IOException{

    File archivo = new File (nombre);
    String ruta = "/C:/Users/devyl/Desktop/pruebaDirectorio.txt";


    if (archivo.exists()){
        System.out.println("Ya existe el archivo.\n");
    } else{
        File directorio = new File(ruta);
        directorio.createNewFile();
        System.out.println("Se creó el archivo exitosamente.\n");


    }

}
```



7. Eliminación de Carpeta

Para eliminar un directorio en java se puede implementar el método delete a un objeto de tipo File el cual será instanciado con la ruta de la carpeta.

Conclusiones

Chagoya Gonzalez Leonardo: Durante la realización de esta práctica se llegaron a conocer más métodos de los implementados para el proyecto 1, el hecho de haber realizado una búsqueda previa ayudó a un entendimiento más rápido.

Celis Hernández Ronie: Se pudo llegar a la conclusión que mediante la realización de esta práctica se aprendió a utilizar los archivos, la escritura, la sobreescritura, así como crear un archivo en una ruta específica. Se comprendió que esta es una buena forma de manejar la información.

Equipo

La práctica se realizó en un 80% pero aun así se lograron cumplir los aspectos principales ya que se logró comprender la forma en la que funcionan los métodos con archivos