**Módulo Acceso a Datos Curso: 2018/2019**

**UT:02 FICHEROS**

**Clase: File**

https://docs.oracle.com/javase/7/docs/api/java/io/File.html

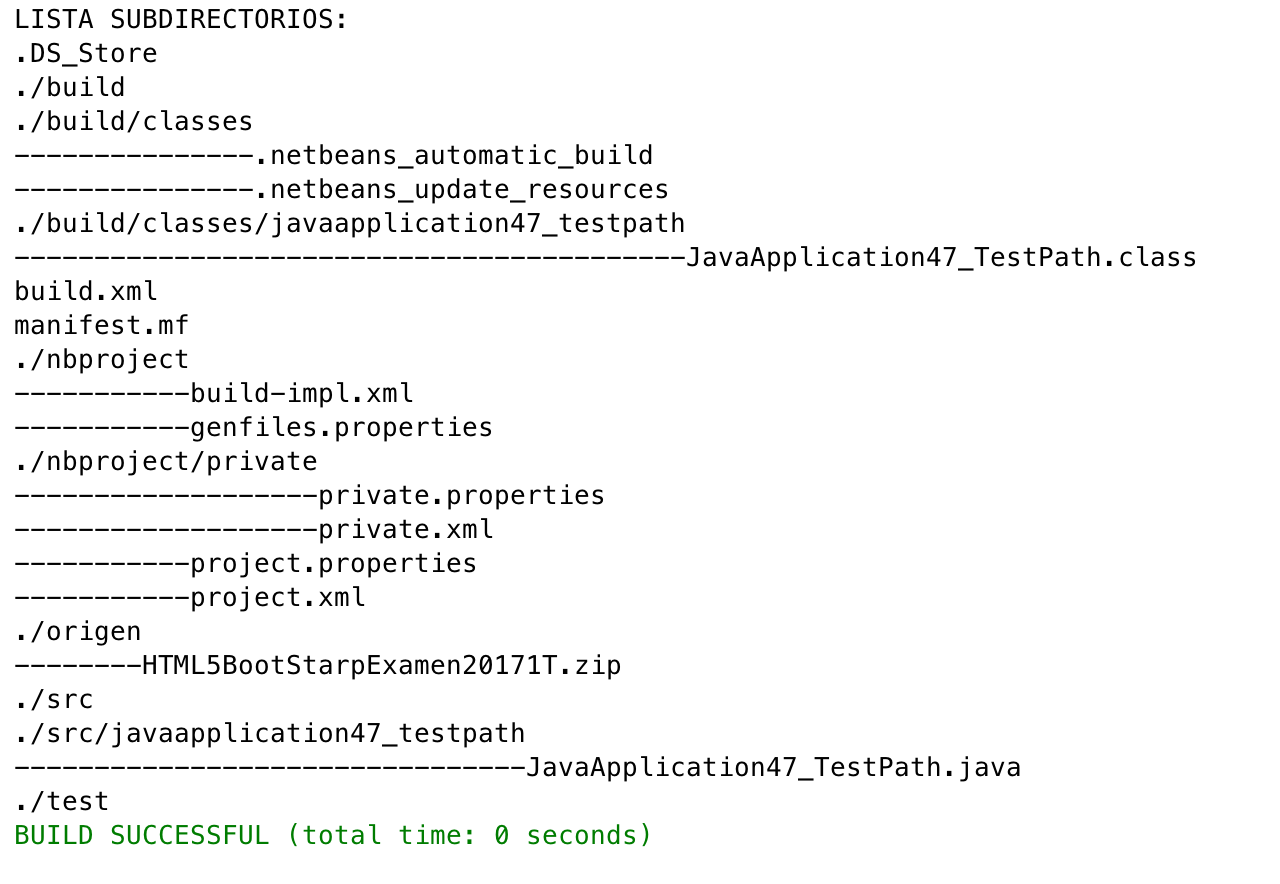
1. Utilizando la clase File crear un proyecto en Netbeans que contenga un paquete Logica con una clase llamada OperacionesFicheros que contenga los métodos:
   1. String [] ListarFicheros(String ruta, boolean ordenadosPorTamaño, boolean soloDirectorios)
      1. Si la ruta está vacia buscara en la raíz en función del OS.
      2. Si ordenadosPorTamaño es True retornará los nombres y tamaño ordenados por tamaño.
      3. Si soloDirectorios es True retornará solo los nombres de los directorios.
      4. Crear dos excepciones personalizadas CarpetaVacia y NoEsUnDirectorioNoSePuedeListar
      5. Probar el correcto funcionamiento desde un método main().
      6. Generar Javadoc.
   2. int crearDirectorios(File rutaOrigen, ArrayList<String> listaDirectorios)
      1. Si la ruta de origen no existe salta una excepción.
      2. Si el directorio existe salta una excepción.
      3. Retorna el total de directorios creados.
      4. Probar el correcto funcionamiento desde un método main().
      5. Generar Javadoc.
   3. int cambiarExtensionFicheros(String ruta, extensionAntigua, extensionNueva)

int cambiarExtensionFicheros(File ruta, extensionAntigua, extensionNueva)

* + 1. Si la ruta de origen existe salta una excepción.
    2. Retorna el numero de ficheros modificados.
    3. Probar el correcto funcionamiento desde un método main().
    4. Generar Javadoc.

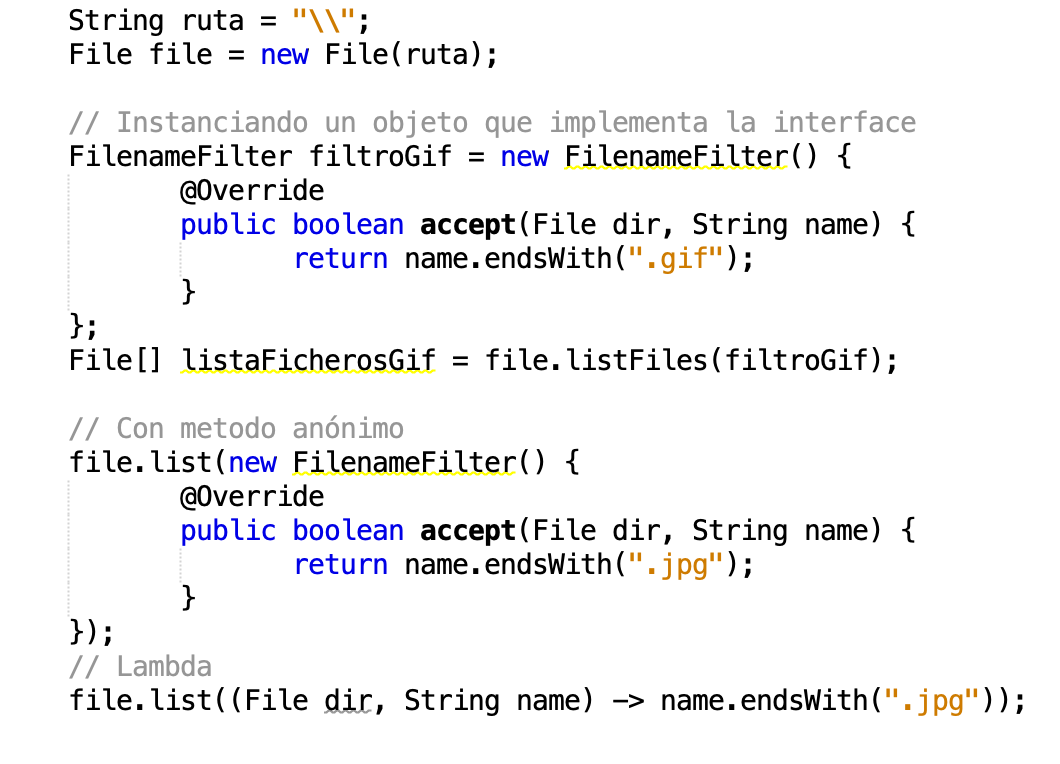
1. Utilizando la clase File crear un proyecto en Netbeans que liste los contenidos de un directorio. (Recursivo)

[Ejemplos Recursividad](https://foro.elhacker.net/ejercicios/ejercicios_recursivos_en_java_y_sus_soluciones-t231013.0.html)



**Interface: FilenameFilter**

* 1. ArrayList<File> listarFicheros(FilenameFilter filtro)



Crear los siguientes filtros implementado la interface FileNameFilter (Se puede crear una clase Filtros):

* + 1. Filtro ficheros de imágenes (jpg, gif, tiff, …)
    2. Filtro ficheros de video (avi, mp4, mkv, …)
    3. Filtro directorios.
    4. Filtro ficheros con un tamaño mínimo.
    5. Filtro fichero modificados en las últimas 24 horas.
    6. Probar el correcto funcionamiento desde un método main().
    7. Generar Javadoc.

**Clases: InputStream y OutputStream (byte streams).**

<https://docs.oracle.com/javase/tutorial/essential/io/bytestreams.html>

1. Utilizando las subclases **FileInputStream** y **FileOutputStream** implementar:
   1. Una clase que cifre los contenidos de los ficheros txt de una determinada carpeta.
   2. Una clase que descifre los contenidos de los ficheros txt de una determinada carpeta.
   3. Calcular la distribución de las letras en un texto común en español utilizando un HashMap<K,V>.
   4. Probar el correcto funcionamiento desde un método main().
   5. Generar Javadoc.

[https://es.wikipedia.org/wiki/Cifrado\_César](https://es.wikipedia.org/wiki/Cifrado_C%C3%A9sar)

**Clases: Reader Writer (character streams).**

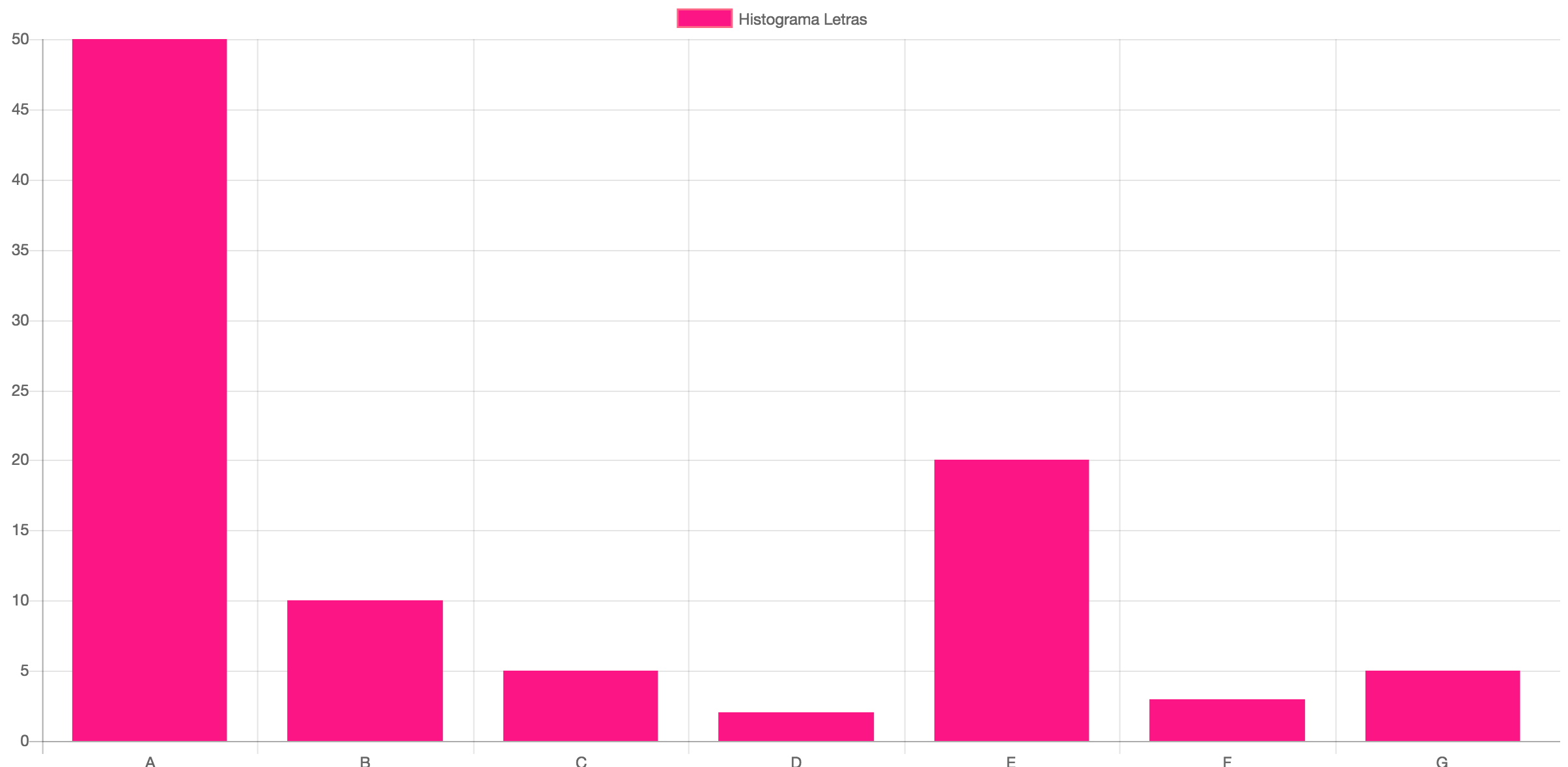
<https://docs.oracle.com/javase/tutorial/essential/io/charstreams.html>

https://docs.oracle.com/javase/tutorial/essential/io/buffers.html

1. Utilizando las subclases clases FileReader y BufferedReader, FileWriter y BuffereWriter:

Realizar un programa que haga un análisis de El Quijote, con los siguientes métodos.

1. Contar el número de líneas.
2. Contar cuantas veces aparece la palabra Quijote.
3. Contar el número de letras.
4. Crear un fichero con las líneas escritas al revés.
5. Mostrar las palabras distintas que hay en el texto y el número de veces que aparecen.
6. Dividir el fichero de El Quijote en los distintos capítulos.
7. Utilizando <http://www.chartjs.org> y las crear un programa que genere una página html con la gráfica de la densidad de letras en español (Reutilizar el método anterior).





1. Con los datos contenidos en el fichero goog.csv que contiene las cotizaciones de google en el último año

* Utilizando el api de chars.js crear una gráfica con los valores “close”. Usar las clases: FileReader y BufferedReader, FileWriter, BuffereWriter y StringTokenizer.

https://docs.oracle.com/javase/7/docs/api/java/util/StringTokenizer.html

1. Transformar el fichero goog.csv en un fichero de datos utilizando el tipo de datos primitivo que sea necesario: boolean, char, byte, short, int, long, float, doublé y string. La fecha debe de ser un string.

<https://docs.oracle.com/javase/tutorial/essential/io/datastreams.html>

1. Crear un fichero de acceso aleatorio que contenga los 100 primeros números primos 2,3,5,7,11… . Crear un método que recibe como argumento un entero y leyendo el fichero nos indica el numero primo que hay en esa posición. Ejemplo numeroPrimo(5) retorna 11.

https://docs.oracle.com/javase/tutorial/essential/io/rafs.html