FICHEROS

1 Realiza un programa que dado un fichero que se le solicite al usuario, muestre su nombre, si es un ejecutable, si está oculto, la ruta relativa, la ruta absoluta y el tamaño.

2 Realiza un programa que cree un directorio en el directorio actual, luego cree tres ficheros en dicho directorio donde uno se borre y otro se renombre. Crearle también un subdirectorio con un fichero dentro. Después mostrar la ruta absoluta de ambos directorios y sus contenidos.

3 Realiza un programa que muestre el nombre y tipo (fichero o directorio) de los ficheros y subdirectorios contenidos en un directorio solicitado al usuario. Mostrar también el contenido de todos los subdirectorios y si éstos contienen subdirectorios también...y así sucesivamente hasta mostrar todo el contenido de dicho directorio.

4 Un filtro sirve para que el método list devuelva solo aquellos archivos o carpetas que cumplan una condición (que tengan una extensión determinada, contengan en su nombre una cadena concreta, empiecen por un carácter, etc). Un filtro es un objeto de una clase que implementa el interface FilenameFilter. Realiza un programa que muestre los archivos de un directorio que posean una extensión concreta. Tanto la extensión como el directorio se solicita al usuario.

5 Realiza un programa que dadas dos rutas, origen y destino, copie el archivo origen(fichero de texto) en el destino de la siguiente manera:

- Si el destino es un directorio, se creará un archivo con el mismo nombre donde se copiará el archivo origen línea a línea(una línea se considera hasta que se encuentre un salto de línea). Utilizar Stream de la clase BufferedReader.
- Si el destino es un archivo, habrá varias opciones según un booleano:
  - Si el booleano es verdadero y el destino es un archivo existente, se reemplazará su contenido por el del archivo origen copiando carácter a carácter.
  - Si el booleano es verdadero y el destino es un archivo inexistente, se lanzará una excepción.
  - Si el booleano es falso y el destino es un archivo existente, se reemplazará su contenido por el del archivo origen copiándolo usando un buffer(array) sin desplazamiento de 20 caracteres. En el caso de la última escritura, si no se llena el buffer, utilizar el desplazamiento para no dejar basura.
  - Si el booleano es falso y el destino es un archivo inexistente, no se hará nada con el archivo y se le dará un mensaje al usuario de que la copia no se puede realizar.

6 Realiza un programa que dado un fichero de texto, se copie en tres ficheros diferentes de tal manera que copie en el primer fichero los primeros 5 caracteres, en el segundo, los 5 siguientes y en el tercero los 5 siguientes, y así sucesivamente hasta copiar todo el fichero. Utilizar lectura sin desplazamiento y escrituras con desplazamiento de un buffer de 15 caracteres. Después, hacer justamente lo contrario. Dados los 3 ficheros, construir uno como el fichero original. Utilizar ahora lecturas con desplazamiento y escritura sin desplazamiento de un buffer de 15 caracteres. Comprobar por código que ambos son iguales. En el caso de la última lectura/escritura, si no se llena el buffer, utilizar el desplazamiento para no dejar basura.

7 Diseñar un programa para encriptar y desencriptar los datos de un fichero de texto. Se

introduce una cadena por teclado que será la clave a aplicar para la encriptación y desencriptación. A cada carácter del fichero de texto original se le sumará una letra de la clave (el código Unicode), cuando se hayan acabado las letras de la palabra clave y aún no se hayan acabado los caracteres del fichero, se volverá al principio de la cadena para seguir aplicando la encriptación. Los datos encriptados se escribirán en un fichero destino, que será usado como origen para desencriptar. Para desencriptar se aplicará la fórmula a la inversa. Por ejemplo, si el fichero origen contiene “abcdef” y la palabra clave es “rosa”, en el fichero destino se escribirán los siguientes caracteres: ”ÓÑÖÅ×Õ”. Es decir, Ó es el resultado de sumar a y r, Ñ es el resultado de sumar b y o, y así sucesivamente.

8 Queremos hacer una agenda telefónica con los siguientes datos:

- Nombre del contacto
- Teléfono
- Dirección
- Código postal (número entero)
- Fecha de nacimiento
- Si le debo dinero (booleano)
- Cuánto le debo(número decimal)

Realiza un programa que almacene los datos en un fichero binario. A continuación, lee el fichero y muestra el contenido por consola. Hacer dos versiones:

- Sin serialización de objetos
- Con serialización de objetos. Almacena varios contactos y cierra el programa. Luego ábrelo de nuevo y almacena unos cuantos contactos más.

9 Haz un programa que lea los contactos del fichero del ejercicio 8 (la versión con serialización de objetos) y los guarde en un fichero aleatorio. En el fichero aleatorio tienes que añadir un identificador a cada contacto antes de los datos. El identificador empieza en 1. Añádele también en el fichero a cada contacto un boolean para indicar si el contacto ha sido borrado. A continuación, hazle el siguiente menú al fichero aleatorio:

- Consultar todos los contactos.
- Consultar un contacto (pedirle al usuario el identificador).
- Añadir un contacto
  - Por el final
  - En la primera posición libre
- Eliminar un contacto poniendo el boolean a false (pedirle al usuario el identificador).
- Modificar si le debo dinero y la cantidad (pedirle al usuario el identificador).
- Compactación del fichero.

Realizar dos versiones del ejercicio:

- Utilizando writeChars
- Utilizando writeUTF

10 Haz un programa que lea los contactos del fichero binario de la agenda telefónica con serialización de objetos del ejercicio 8 y los escriba en un fichero XML usando la librería XStream.

11 Dado el XML del ejercicio anterior, utilizando XStream, pásalo a un fichero binario sin serialización de objetos. Compara que sea igual al fichero binario sin serialización de objetos obtenido en el ejercicio 8. Haz la comparación en binario.

12 Dado el fichero XML, añádele los siguientes atributos utilizando un editor de textos:

- Prefijo del país en el teléfono
- Localidad en la dirección
- Tipo de moneda en el dinero

Procésalo con SAX y guarda los datos en un fichero de texto. Guarda cada contacto en una línea. Los datos deben estar alineados por columnas. Ejemplo:
Juan García +34 956 23 45 67 C/Romero Peña, 20 Algeciras 11205 09/01/1987 No me debe dinero
María Pérez +34 956 66 54 97 C/Tramontana, 35 La Línea 11207 21/02/1985 Sí me debe dinero: 67,43 euros

13 Crea una plantilla XSL para dar una presentación al fichero XML de la agenda telefónica. Realiza un programa para transformarlo en HTML.

14 A partir del fichero XML de la agenda telefónica, obtener un fichero Json utilizando la librería Gson.

15 Dado el fichero Json del ejercicio anterior, obtener un fichero de texto con el siguiente formato:

```
**********************************************************************************
                                AGENDA TELEFONICA
**********************************************************************************
        Nombre:             Juan García
        Teléfono:           956 23 45 67
        Dirección:          C/Romero Peña, 20
        Código Postal:      11205
        Fecha de nacimiento: 09/01/1987
        Me debe dinero:     No
        Cuánto:             0,0
**********************************************************************************
        Nombre:             María Pérez
        Teléfono:           956 66 54 97
        Dirección:          C/Tramontana, 35
        Código Postal:      11207
        Fecha de nacimiento: 21/02/1985
        Me debe dinero:     Sí
        Cuánto:             67,43
**********************************************************************************
                            FIN DE LA AGENDA TELEFONICA
**********************************************************************************
```

16 Dado el fichero Json anterior, mostrar todos los datos de un contacto cuyo nombre nos indique el usuario. La búsqueda se hará directamente en el fichero Json. No utilizar Gson.

17 Realiza una función que te permita buscar en un json por un número indeterminado de campos.