**ARCHIVOS RANDÓMICOS (ACCESO DIRECTO)**

A diferencia de los archivos secuenciales, los archivos con organización randómica, permiten la lectura y escritura en cualquier lugar del archivo.

**CLASES PARA ADMINISTRAR UN ARCHIVO RANDÓMICO**

**File**

Los objetos de esta clase, permiten definir la ruta y el nombre del archivo.

Ejemplo de instancia de la clase File:

**static** File *f* = **new** File("g:/sistema/datos.dat");

Declaramos el objeto f, siendo la ubicación: *datos.da*t y la ruta de ubicación es: *g:/demosran*.

**RandomAccesFile**

Ejemplo de instancia de la clase RandomAccesFile:

RandomAccesFile raf = new RandomAccesFile(f,tipo);

Donde: f, es el objeto de la ruta.

tipo, es el modo que define si el proceso será, sólo de lectura (“r”), sólo de escritura (“w”) o de lectura y/o escritura (“rw”).

Los objetos de esta clase contienen métodos que permiten administrar acciones de: ubicación, lectura y escritura en el archivo, siendo ellos los siguientes:

.seek() .- Ubica el puntero de lectura y/o escritura, en algún lugar del archivo.

.length() .- Devuelve la longitud del archivo.

.close() .- Cierra la comunicación con el archivo.

.writeBoolean() .- Escribe en el archivo un campo tipo boolean, ocupando un byte.

.writeInt() .- Escribe en el archivo un campo tipo entero corto, ocupando cuatro bytes.

.writeFloat() .- Escribe en el archivo un campo tipo real corto, ocupando cuatro bytes.

.writeLong() .- Escribe en el archivo un campo tipo entero largo, ocupando cuatro bytes.

.writeDouble() .- Escribe en el archivo un campo tipo real largo, ocupando cuatro bytes.

.writeUTF() .- Escribe en el archivo un campo tipo String (cadena de caracteres), ocupando un tamaño predefinido más dos bytes, en caso de ser la longitud menor al determinado, se debe agregar o completar con espacios a la derecha.

.readBoolean() .- Lee del archivo, un campo tipo boolean.

.readInt() .- Lee del archivo un campo tipo entero.

.readFloat() .- Lee del archivo un campo tipo real corto.

.redLong() .- Lee del archivo un campo tipo entero largo.

.redDouble() .- Lee del archivo un campo tipo real largo.

.readUTF() .- Lee del archivo un campo tipo String (cadena de caracteres).

**FORMA DE DETERMINAR LA LONGITUD DE REGISTRO DEL ARCHIVO**

Antes de implementar el archivo, se debe definir la estructura del registro, para determinar la longitud del mismo, a partir de longitudes de los campos que forman el registro.

Ejemplo:

Se desea almacenar los siguientes datos: idProducto (int), nombreProducto(String 30), costo (float)

|  |  |  |  |
| --- | --- | --- | --- |
| flag | Id del producto | Nmbre del producto | Costo unitario |
| Boolean (1 byte) | Int (4 bytes) | String (30 bytes mas 2 ) | Float (4 bytes) |

Como se puede ver en el cuadro anterior, se agrega el campo flag tipo boolean, el cual nos permitirá realizar bajas lógicas, siendo el valor true para registros validos y false para registros dados de baja.

Dado los campos, se llega a la conclusión de que cada registro del archivo estará formado por 41 bytes.

**Ejemplo de alta:**

Antes de ejecutar el siguiente código, se debe crear la carpeta dentro la unidad g:/sistema, si en su computadora no tiene la unidad g, cámbielo por una unidad de su computadora.

**package** abcrANDOM;

**import** java.io.File;

**import** java.io.RandomAccessFile;

**import** javax.swing.JOptionPane;

**public** **class** Demo {

**static** File *f* = **new** File("g:/sistema/datos2.dat");

**public** **static** **void** main(String[] args) {

*alta*(**true**, 323, "Leche", 42.34f);

*alta*(**true**, 256, "Yogourt", 55.34f);

*alta*(**true**, 456, "Azucar", 57.8f);

//listaArchivo();

}

**static** **void** alta(**boolean** flag, **int** idProducto, String nombre, **float** costoUnitario) {

**try** {

RandomAccessFile raf = **new** RandomAccessFile(*f*, "rw");

raf.seek(raf.length());

raf.writeBoolean(flag);

raf.writeInt(idProducto);

raf.writeUTF(*completaConEspacios*(nombre, 30));

raf.writeFloat(costoUnitario);

raf.close();

System.***out***.println("exito en la alta");

}**catch**(Exception e) {

JOptionPane.*showMessageDialog*(**null**, "error "+e.getMessage());

}

}

**static** String completaConEspacios(String nombre, **int** tamanio) {

**while**(nombre.length()<tamanio) {

nombre = nombre + " ";

}

**return** nombre;

}

**static** **void** listaArchivo() {

**try** {

RandomAccessFile raf = **new** RandomAccessFile(*f*, "r");

**int** pos = 0;

System.***out***.println("Listado del archivo ");

**while**(pos < raf.length()) {

raf.seek(pos);

**boolean** flag = raf.readBoolean();

**if**(flag == **true**) {

**int** idProd = raf.readInt();

String nombre = raf.readUTF();

**float** costo = raf.readFloat();

System.***out***.println(idProd + " " +nombre + " " +costo);

}

pos = pos + 41;

}

}**catch**(Exception e) {

JOptionPane.*showMessageDialog*(**null**, "error "+e.getMessage());

}

}

}

Luego de correr el programa, desde el explorador del archivo puede ir a la unidad o ruta definida, en ella vera el archivo: datos.dat.

Para ver el listado en consola, dentro el módulo main(), comentar las llamadas al módulo alta y activar la llamada al módulo listaArchivo(), de la siguiente maneara:

**public** **static** **void** main(String[] args) {

//alta(true, 323, "Leche", 42.34f);

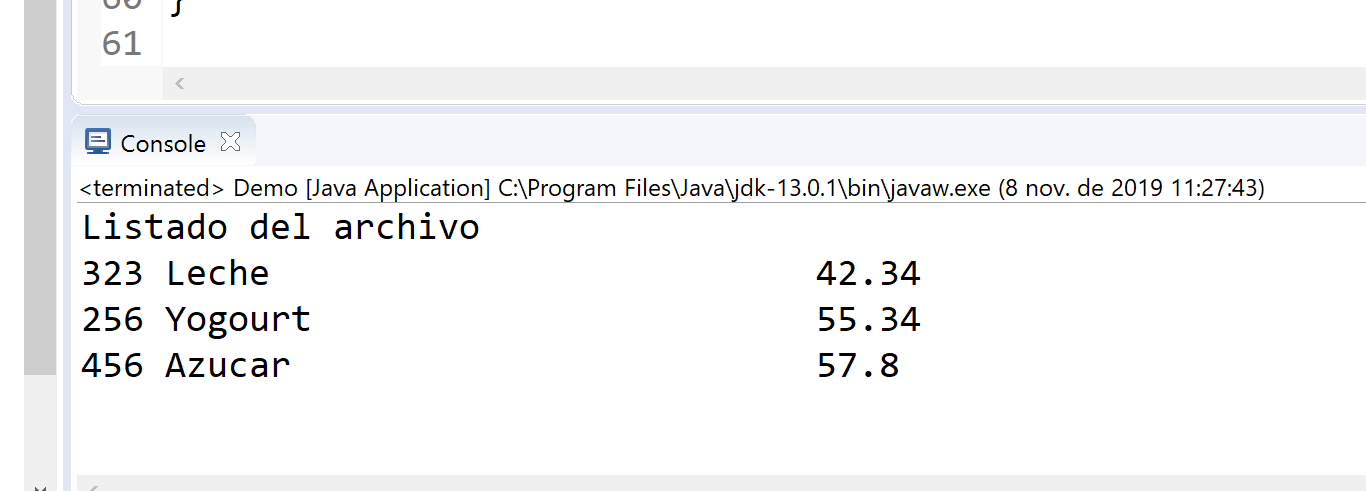
//alta(true, 256, "Yogourt", 55.34f);

//alta(true, 456, "Azucar", 57.8f);

*listaArchivo*();

}

Luego de correr nuevamente el programa, verá lo siguiente:



Favor probar y analizar. Les enviare luego un ejemplo completo donde se realizan: alta, baja, cambio, listado y búsqueda.