

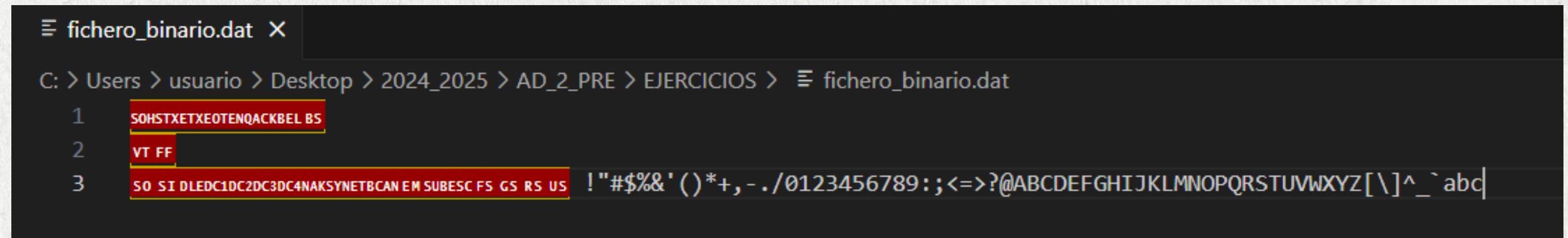
UNIDAD IO Y II: FICHEROS

CLASE 3: FICHEROS BINARIOS

FICHEROS BINARIOS

Contienen secuencias de números binarios que no son legibles por el usuario

Ocupan menos espacio en disco



A screenshot of a terminal window titled "fichero_binario.dat". The window shows the file path: C: > Users > usuario > Desktop > 2024_2025 > AD_2_PRE > EJERCICIOS > fichero_binario.dat. The content of the file is displayed in three lines:

```
1 SOHSTXETXEOTENQACKBEL BS
2 VT FF
3 SO SI DLEDC1DC2DC3DC4NAKSYNETBCAN EM SUBESC FS GS RS US ! "#$%&' ()*+, - ./0123456789:;=>?@ABCDEFGHIJKLMNPQRSTUVWXYZ[\]^_`abc
```

LECTURA DE FICHEROS BINARIOS

Métodos:

- Constructor: FileInputStream
(Objeto_File_a_leer)
- int read()
- int read(byte[] buf)
- int read(byte[] buf, int desplazamiento, int n)

ESCRITURA DE FICHEROS BINARIOS

Métodos:

- Constructor: FileOutputStream
(Objeto_File_a_escribir)
- void write(int b)
- void write(byte[] buf)
- void write(byte[] buf, int desplazamiento, int n)

LECTURA Y ESCRITURA DE FICHEROS BINARIOS

```
package ficheros;
import java.io.*;

public class ejercicio4 {

    public static void main(String[] args) throws IOException {
        File fichero = new File ("C:\\\\Users\\\\usuario\\\\Desktop\\\\2024_2025\\\\AD_2_PRE\\\\EJERCICIOS\\\\fichero_binario.dat"); //declara fichero
        FileOutputStream fileout = new FileOutputStream(fichero); //crea flujo de salida hacia el fichero
        FileInputStream filein = new FileInputStream(fichero); //crea flujo de entrada

        for (int i = 1; i < 100; i++) {
            fileout.write(i); //escribe datos en el flujo de salida
        }
        fileout.close(); //cerrar stream de salida

        int i = 0;
        //visualizar los datos del fichero
        while ((i = filein.read()) != -1) {
            System.out.println(i);
        }
        filein.close(); //cerrar stream de entrada

    }
}
```

LECTURA DE FICHEROS BINARIOS (PRIMITIVAS)

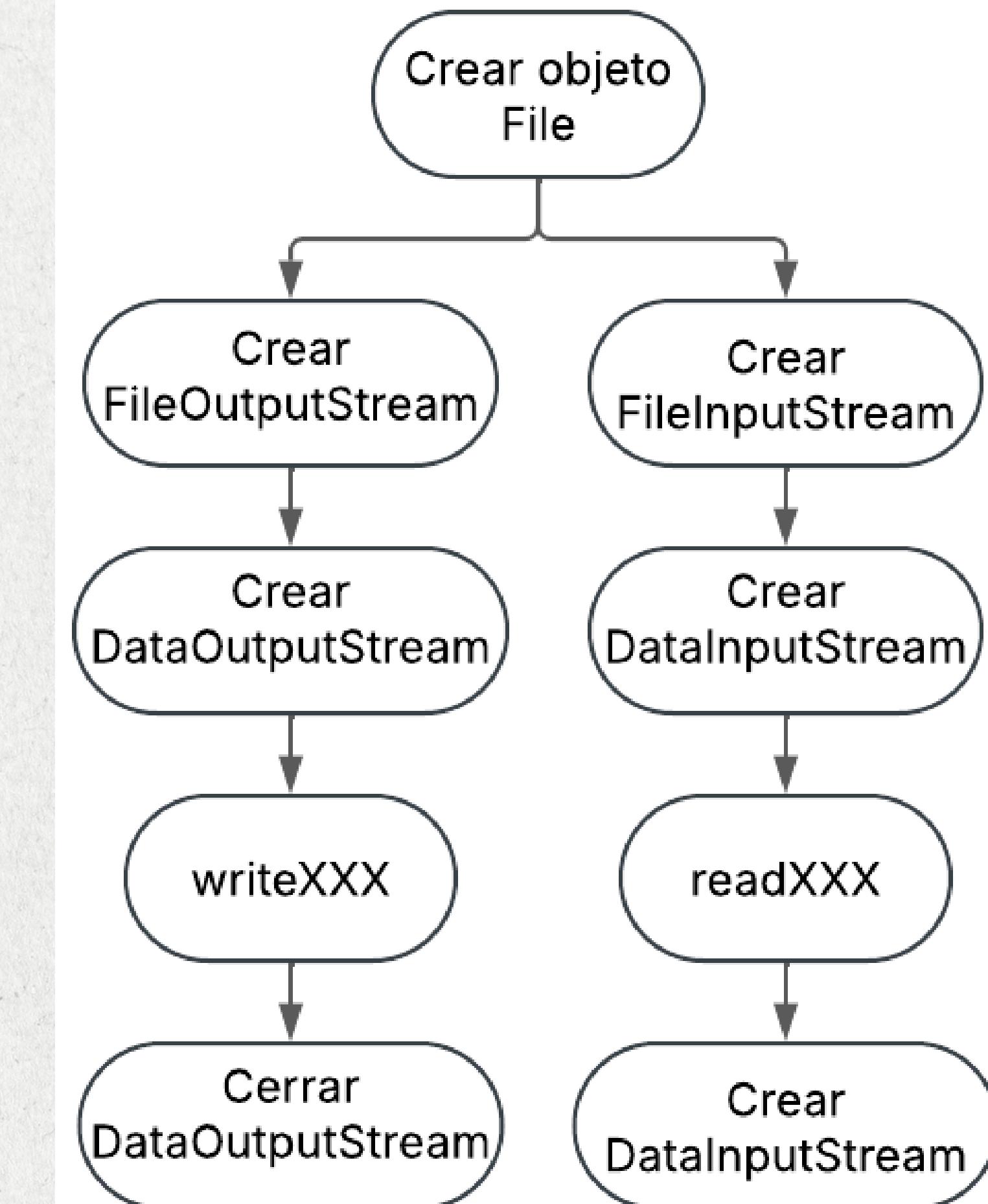
Métodos:

- Constructor Output: DataOutputStream
(Objeto_FileOutputStream)
- Constructor Input: DataInputStream
(Objeto_FileInputStream)

MÉTODOS PARA LECTURA	MÉTODOS PARA ESCRITURA
boolean readBoolean();	void writeBoolean(boolean v);
byte readByte();	void writeByte(int v);
int readUnsignedByte();	void writeBytes(String s);
int readUnsignedShort();	void writeShort(int v);
short readShort();	void writeChars(String s);
char readChar();	void writeChar(int v);
int readInt();	void writeInt(int v);
long readLong();	void writeLong(long v);
float readFloat();	void writeFloat(float v);
double readDouble();	void writeDouble(double v);
String readUTF();	void writeUTF(String str);

ESCRITURA Y LECTURA DE FICHEROS

LECTURA DE FICHEROS BINARIOS (PRIMITIVAS)



SECCIÓN DE LA EXCEPCIÓN

- readXXX lanza la excepción EOFException
“End Of File”

DEBE MANEJARSE CON TRY/CATCH

EJERCICIOS

Ejercicio 1:

Toma los datos del fichero “nombres_y_edades.txt” y, en Java, crea un nuevo fichero y escribe los valores. Después, leelos y escríbelos por pantalla

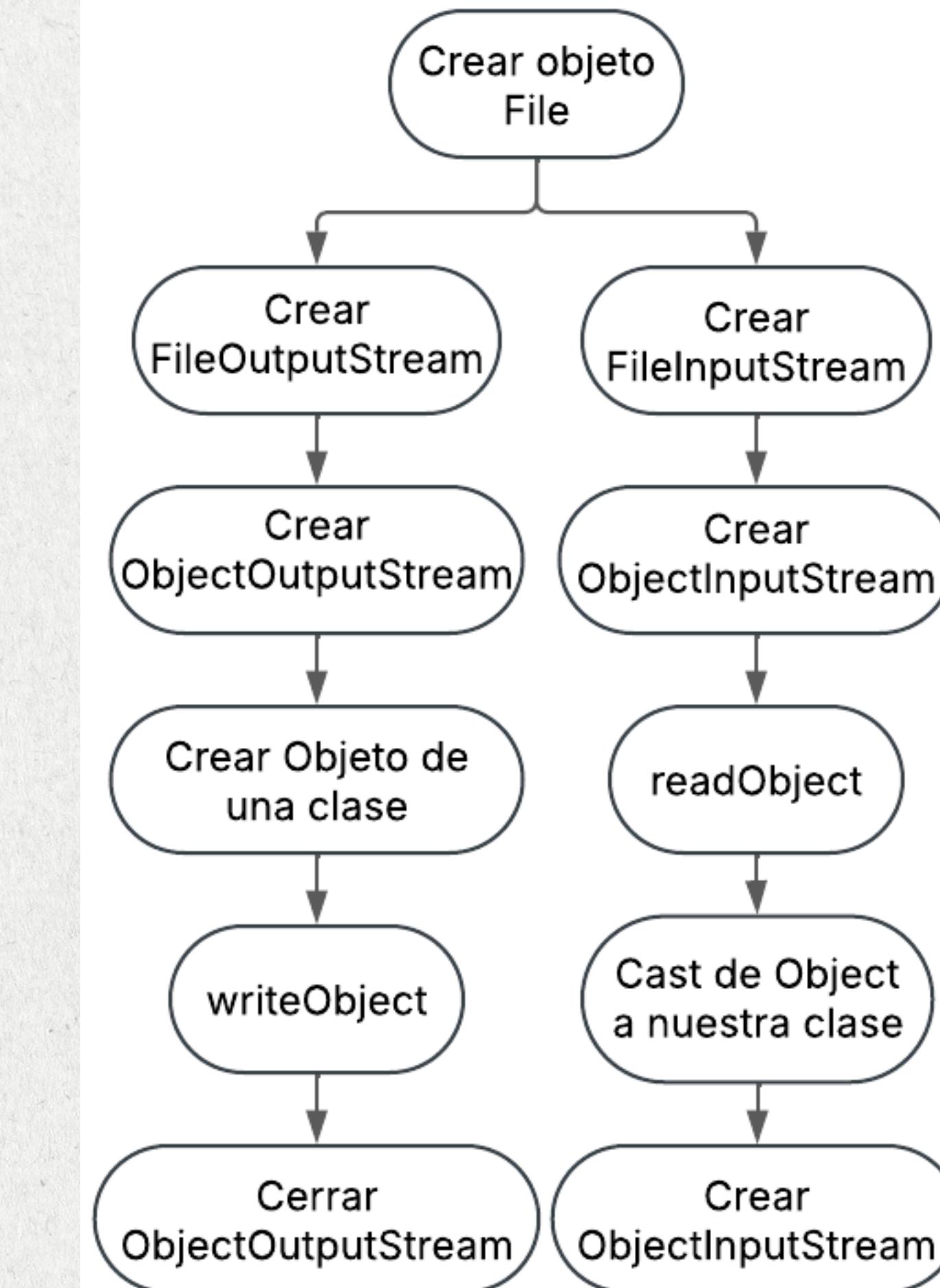
LECTURA Y ESCRITURA DE OBJETOS EN FICHEROS BINARIOS

Métodos:

- Constructor Output: ObjectOutputStream
(Objeto_FileOutputStream)
- Constructor Input: ObjectInputStream
(Objeto_FileInputStream)
- Object readObject();
- writeObject(Object);

ESCRITURA Y LECTURA DE FICHEROS

LECTURA Y ESCRITURA DE OBJETOS EN FICHEROS BINARIOS



SECCIÓN DE LA EXCEPCIÓN

- readObject lanza las excepciones IOException y ClassNotFoundException
- writeObject puede generar la excepción IOException

**AMBAS DEBEN MANEJARSE CON
TRY/CATCH O THROW**

EJERCICIOS

Ejercicio 2:

Con la clase Calle, crea un código que escriba varias calles en un fichero y después lee todos los valores y muéstralos por pantalla

Tamaños de tipos de datos en memoria:

- char 2 bytes
- short 2 bytes
- int 4 bytes
- long 4 bytes
- double 8 bytes

FICHEROS DE ACCESO ALEATORIO

ID:1, Nombre:FERNANDO, Curso:1, Altura:1.45
ID:2, Nombre:ROCIO, Curso:2, Altura:2.0
ID:3, Nombre:LOPE, Curso:1, Altura:1.7
ID:4, Nombre:ROSA, Curso:1, Altura:1.56
ID:5, Nombre:JOAQUIN, Curso:3, Altura:1.6
ID:6, Nombre:MARIO, Curso:3, Altura:1.87
ID:7, Nombre:REY, Curso:2, Altura:2.0|

FICHEROS DE ACCESO ALEATORIO

ID	NOMBRE	CURSO	ALTURA
1	FERNANDO	1	1.45
2	ROCIO	2	2.0
3	LOPE	1	1.7
4	ROSA	1	1.56

FICHEROS DE ACCESO ALEATORIO

1FERNANDO11.45

2ROCIO22.0

3LOPE11.7

4ROSA11.56

5JOAQUIN31.6

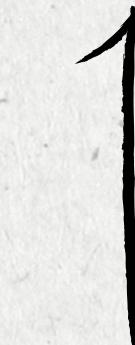
6MARIO31.87

7REY22.0

1|FERNANDO|1|1.45



4 bytes



20 bytes



4 bytes



8 bytes

= 36 bytes

FICHEROS DE ACCESO ALEATORIO

36 bytes

posicion = 0 → 1FERNANDO11.45

posicion = 36 → 2ROCI022.0

posicion = 72 → 3LOPE11.7

posicion = 108 → 4ROSA11.56

posicion = 144 → 5JOAQUIN31.6

posicion = 180 → 6MARIO31.87

posicion = 216 → 7REY22.0

posicion =
 $(id-1)*36$

FICHEROS DE ACCESO ALEATORIO

1FERNANDO11.45
2ROCIO22.0
3LOPE11.7
4ROSA11.56
5JOAQUIN31.6
6MARIO31.87
7REY22.0

FICHEROS DE ACCESO ALEATORIO

1FERNANDO11.45

2ROCIO22.0

-1_00.0

4ROSA11.56

-1JOAQUIN31.6

6MARIO31.87

7REY22.0

LECTURA Y ESCRITURA ALEATORIA

Métodos:

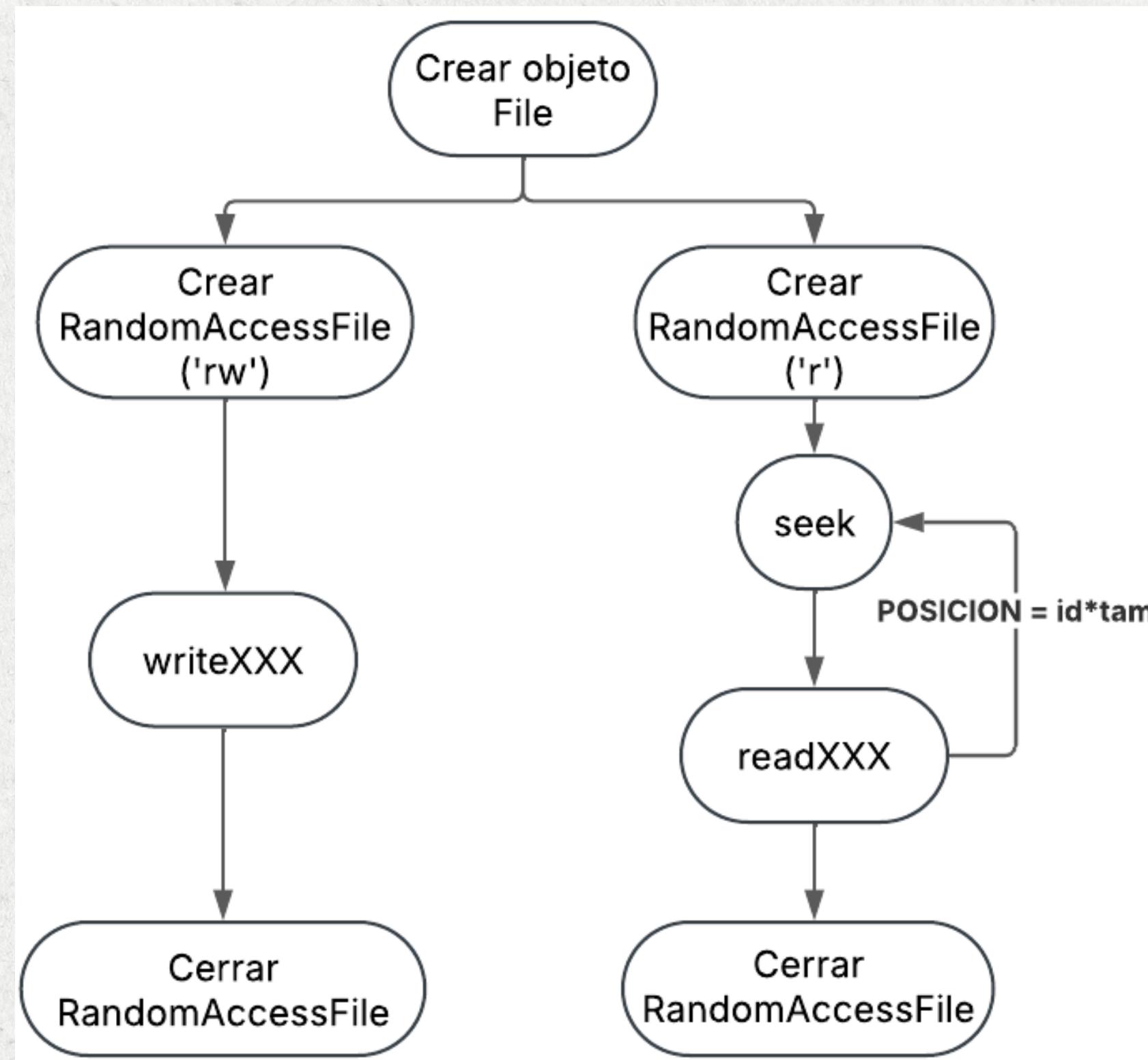
- Constructor: RandomAccessFile (String nombreFichero, String modoAcceso)
- Constructor: RandomAccessFile (File objetoFile, String modoAcceso)
- XXX readXXX()
- XXX writeXXX()

- long getFilePointer()
- void seek(long position)
- long length()
- int skipBytes(int desplazamiento)

Modos de acceso:

- r: únicamente lectura
- wr: lectura y escritura

LECTURA Y ESCRITURA ALEATORIA

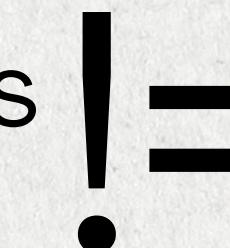


LECTURA Y ESCRITURA ALEATORIA

```
buffer = new StringBuffer ( nombre[i] );
buffer.setLength(10); //10 caracteres para el nombre
fileW.writeChars (buffer.toString()); //insertar nombre|
```

NO PODEMOS ESCRIBIR CADENAS
DIRECTAMENTE CON writeUTF()

Los strings ocupan 10 bytes
+ longitud de la cadena



10 caracteres de 2 bytes
que ocupan 20 bytes

EJERCICIOS

Ejercicio 3.1:
Toma el fichero
acceso_aleatorio.txt y
crea un programa que
pueda leer todos los
registros o leer un
registro concreto según
elija el usuario por teclado

Componentes de cada
registro

- Identificador: Int - 4 bytes
- Nombre: Max 10 caracteres - 20 bytes
- Altura: Double - 8 bytes

Total: 32 bytes

EJERCICIOS

Ejercicio 3.2:
Toma el fichero
acceso_aleatorio.txt y
crea un programa que
pueda modificar los datos
de un registro, pidiendo los
tres datos necesarios y
accediendo a la posición

Componentes de cada
registro

- Identificador: Int - 4 bytes
- Nombre: Max 10 caracteres - 20 bytes
- Altura: Double - 8 bytes

Total: 32 bytes

EJERCICIOS

Ejercicio 3.3:

Toma el fichero acceso_aleatorio.txt y crea un programa que pueda borrar un registro, pidiendo el identificador por teclado. Los datos del registro borrado serán -1 para identificador, nombre a “ ” y altura a 0.00

Componentes de cada registro

- Identificador: Int - 4 bytes
- Nombre: Max 10 caracteres - 20 bytes
- Altura: Double - 8 bytes

Total: 32 bytes

EJERCICIOS

Ejercicio 3.4:
Toma el fichero
acceso_aleatorio.txt y
crea un programa que
pueda insertar registros
en cualquier posición
vacía, si una posición está
ocupada, pregunta por
otra posición

Componentes de cada
registro

- Identificador: Int - 4 bytes
- Nombre: Max 10 caracteres - 20 bytes
- Altura: Double - 8 bytes

Total: 32 bytes