Método para leer un Binario

```java
public List<Producto> leerBinario(String ruta) throws IOException{
    ObjectInputStream ois = null;
    List<Producto> listaProductos = new ArrayList<>();
    try{
        ois = new ObjectInputStream(new FileInputStream(ruta));
        while(true){
            try{
                Producto p = (Producto) ois.readObject();
                listaProductos.add(p);
            }catch (EOFException e){
                break;
            }catch (ClassNotFoundException e){
                System.out.println("No se encontro el archivo");
            }
        }
    }catch (FileNotFoundException e){
        System.out.println("No se encontro el archivo");
    }finally {
        if(ois != null){
            ois.close();
        }
    }
    return listaProductos;
}
```

Método para escribir un Binario:

```java
public void escribirBinario(List<Producto> lista, String ruta)throws IOException{
    ObjectOutputStream oos = null;
    try {
        oos = new ObjectOutputStream(new FileOutputStream(ruta));
        for (Producto p : lista) {
            oos.writeObject(p);
        }
    }catch (IOException e){
        System.out.println(e.getMessage());
    }finally {
        if(oos != null){
            oos.close();
        }
    }
}
```

Método para leer un CSV:

```java
public List<Producto> leerCSV (String ruta) throws IOException {
    BufferedReader br = null;
    List<Producto> lista = new ArrayList<>();
    try{
        br = new BufferedReader(new FileReader(ruta));
        String linea;
        while((linea = br.readLine()) != null ){
            String[] partesLista = linea.split( regex: ",");
            int id = Integer.parseInt(partesLista[0]);
            String nombre = partesLista[1];
            double precio = Double.parseDouble(partesLista[2]);

            Producto p = new Producto(id, nombre, precio);

            lista.add(p);
        }
    }catch(IOException e){
        e.printStackTrace();
    }finally {
        if(br != null){
            br.close();
        }
    }
    return lista;
}
```

Método para escribir un CSV:

```java
public void escribirCSV(List<Producto> lista, String ruta) throws IOException{
    BufferedWriter bw = null;
    try{
        bw = new BufferedWriter(new FileWriter(ruta));
        for(Producto p : lista){
            bw.write( str: p.getId()+","+p.getNombre()+","+p.getPrecio());
            bw.newLine();
        }
    }catch (IOException e){
        System.out.println(e.getMessage());
    }finally {
        if(bw != null){
            bw.close();
        }
    }
}
```

Método para leer un XML:

```java
public List<Alumno> leerXMLAlumno(String ruta)throws Exception{ 1 usage  JorgeDelHoyo*
    List<Alumno> lista = new ArrayList<>();
    try{
        // Necesario
        Document doc = DocumentBuilderFactory.newInstance().newDocumentBuilder().parse(new File(ruta));
        doc.getDocumentElement().normalize();

        // Cada <alumno> en <alumnos> es un nodo
        NodeList nodos = doc.getElementsByTagName("alumno");
        // Recorre todos los nodos <alumno>
        for(int i = 0; i < nodos.getLength(); i++){
            Node nodo = nodos.item(i);
            if(nodo.getNodeType() == Node.ELEMENT_NODE){ // Verifica que <alumno> es un elemento nodo
                Element elem = (Element) nodo;
                int id = Integer.parseInt(elem.getAttribute( name: "id")); // Se lee el atributo
                String nombre = elem.getElementsByTagName("nombre").item( index: 0).getTextContent();
                double nota = Double.parseDouble(elem.getElementsByTagName("nota").item( index: 0).getTextContent());

                lista.add(new Alumno(id, nombre, nota));
            }
        }
    }catch(Exception e){
        System.out.println("Error: "+e.getMessage());
    }
    return lista;
}
```

Método para escribir un XML:

```java
transformer.setOutputProperty("{http://xml.apache.org/xslt}indent-amount", "2");
import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
import javax.xml.parsers.*;
import javax.xml.transform.*;
import javax.xml.transform.dom.DOMSource;
import javax.xml.transform.stream.StreamResult;
import java.io.File;
import java.util.ArrayList;
import java.util.List;
```

```java
public void escribirXMLAlumno(List<Alumno>lista, String ruta)throws Exception{  no usages  JorgeDelHoyo
    try{
        //Codigo necesario
        DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
        DocumentBuilder builder = factory.newDocumentBuilder();
        Document doc = builder.newDocument();

        //Elemento raiz <alumnos>...</alumnos>
        Element root = doc.createElement( tagName: "alumnos");
        doc.appendChild(root);

        //Elementos dentro de <alumnos>(...)</alumnos>
        for(Alumno a : lista){
            //Dentro de alumnos se crea un alumno
            Element alumnoElement = doc.createElement( tagName: "alumno");
            //Se añade un atributo al elemento alumno <alumno id=">
            alumnoElement.setAttribute( name: "id",String.valueOf(a.getIdAlumno()));
            //Se crea un elemnto <nombre>(...)</nombre> dentro de alumno
            Element nombre = doc.createElement( tagName: "nombre");
            nombre.setTextContent(a.getNombreAlumno());
            alumnoElement.appendChild(nombre);
            // Se crea un elemento <nota>(...)</nota> dentro de alumno
            Element nota = doc.createElement( tagName: "nota");
            nota.setTextContent(String.valueOf(a.getNotaAlumno()));
            alumnoElement.appendChild(nota);

            root.appendChild(alumnoElement);
        }

        // Guarda el documento XML en un archivo
        TransformerFactory transformerFactory = TransformerFactory.newInstance();
        Transformer transformer = transformerFactory.newTransformer();
        transformer.setOutputProperty(OutputKeys.INDENT, value: "yes");
        transformer.setOutputProperty( name: "{http://xml.apache.org/xslt}indent-amount",  value: "2");

        DOMSource source = new DOMSource(doc);
        StreamResult result = new StreamResult(new File(ruta));
        transformer.transform(source, result);

        System.out.println("XML creado correctamente");
    }catch(Exception e){
        System.out.println("Error: "+e.getMessage());
    }
}
```

Método para leer Aleatorio Por ID:

```
private final int TAMAÑO_REGISTRO = 4+(10*2)+8; // INT+CHAR+DOUBLE  4 usages

public List<Alumno> leerAleatorioAlumnosPorID(String ruta, int id)throws IOException{
    List<Alumno> lista = new ArrayList<>();
    RandomAccessFile raf = null;
    try{
        raf = new RandomAccessFile(ruta, mode: "rw");
        long posicionAlumno = (id-1) * TAMAÑO_REGISTRO;
        if(posicionAlumno >= raf.length()){
            System.out.println("Id fuera de rango");
        }else{
            raf.seek(posicionAlumno);

            int idAlumno = raf.readInt();
            char[] nombreChars = new char[10];
            for(int i = 0; i < nombreChars.length; i++){
                nombreChars[i] = raf.readChar();
            }
            String nombreAlumno = new String(nombreChars);
            double notaAlumno = raf.readDouble();

            lista.add(new Alumno(idAlumno,nombreAlumno,notaAlumno));
        }
    }catch (IOException e){
        System.out.println(e.getMessage());
    }finally {
        if(raf != null){
            raf.close();
        }
    }
    return lista;
}
```

Método para leer Aleatorio a partir de ID:

```
public List<Alumno> leerAleatorioAlumnosAPartirID(String ruta, int id)throws IOException{
    List<Alumno> lista = new ArrayList<>();
    RandomAccessFile raf = null;
    try{
        raf = new RandomAccessFile(ruta, mode: "rw");
        long posicionAlumno = (long) (id-1)*TAMAÑO_REGISTRO;
        if(posicionAlumno >= raf.length()){
            System.out.println("Id fuera de rango");
        }else{
            raf.seek(posicionAlumno);
            while(raf.getFilePointer() < raf.length()){
                int idAlumno = raf.readInt();
                char[] nombreCharsAlumno = new char[10];
                for(int i = 0; i < nombreCharsAlumno.length;i++){
                    nombreCharsAlumno[i] = raf.readChar();
                }
                String nombreAlumno = new String(nombreCharsAlumno);
                double notaAlumno = raf.readDouble();

                lista.add(new Alumno(idAlumno,nombreAlumno,notaAlumno));
            }
        }
    }catch (IOException e){
        System.out.println(e.getMessage());
    }finally {
        if(raf != null){
            raf.close();
        }
    }
    return lista;
}
```

Método para escribir Aleatorio:

```
public void escribirAleatorioAlumno(List<Alumno>lista, String ruta)throws IOException{
    RandomAccessFile raf = null;
    try{
        raf = new RandomAccessFile(ruta, mode: "rw");
        for(Alumno a : lista){
            raf.writeInt(a.getIdAlumno());
            StringBuffer sb = new StringBuffer(a.getNombreAlumno());
            sb.setLength(10);
            raf.writeChars(sb.toString());
            raf.writeDouble(a.getNotaAlumno());
        }
    }catch (IOException e){
        System.out.println(e.getMessage());
    }finally {
        if(raf != null){
            raf.close();
        }
    }
}
```

Método para escribir Aleatorio a partir ID:

```
public void escribirAleatorioAPartirID(List<ExamenPrueba>lista, String ruta, int id)throws IOException{
    RandomAccessFile raf = null;
    try{
        raf = new RandomAccessFile(ruta, mode: "rw");
        long posicionExamen = (long) (id-1)*TAMANIO_REGISTRO2;

        raf.seek(posicionExamen);
        raf.setLength(0);
        for(ExamenPrueba e: lista){
            if(e.getId() >= id){
                raf.writeInt(e.getId());
                StringBuffer sb = new StringBuffer(e.getNombre());
                sb.setLength(5);
                raf.writeChars(sb.toString());
                raf.writeFloat(e.getNotaMaxima());
                StringBuffer sb2 = new StringBuffer(e.getFecha());
                sb2.setLength(7);
                raf.writeChars(sb2.toString());
                raf.writeBoolean(e.isAprobado());
            }
        }
    }catch (FileNotFoundException e){
        System.out.println("No se encontro el archivo");
    }finally {
        if(raf != null){
            raf.close();
        }
    }
}
```

Método para escribir Aleatorio por ID:

```
public void escribirAleatorioID(List<ExamenPrueba>lista, String ruta, int id)throws IOException{
    RandomAccessFile raf = null;
    try{
        raf = new RandomAccessFile(ruta, mode: "rw");
        long posicionExamen = (long) (id-1)*TAMANIO_REGISTRO2;

        raf.seek(posicionExamen); // Te posiciona donde debe
        raf.setLength(0); // SOLO ESCRIBE EL DESEADO
            for(ExamenPrueba e: lista){
                if(e.getId() == id){
                    raf.writeInt(e.getId());
                    StringBuffer sb = new StringBuffer(e.getNombre());
                    sb.setLength(5);
                    raf.writeChars(sb.toString());
                    raf.writeFloat(e.getNotaMaxima());
                    StringBuffer sb2 = new StringBuffer(e.getFecha());
                    sb2.setLength(7);
                    raf.writeChars(sb2.toString());
                    raf.writeBoolean(e.isAprobado());
                    break;
                }
            }
    }catch (FileNotFoundException e){
        System.out.println("No se encontro el archivo");
    }finally {
        if(raf != null){
            raf.close();
        }
    }
}
```

Tamaño Bytes:

- Int / Float → 4 byte
- Char / Short → 2 byte
- Double / Long → 8 byte
- Byte / Boolean → 1 bit