Objectius:
- Aprendre a serialitzar objectes JAVA

Instruccions:
- Responeu a l'espai de cada pregunta, si ho feu amb diapositives o captures d'images enganxeu la diapositiva en aquest mateix espai.
- Es valorarà la presentació i els comentaris al codi

Criteris d'avaluació:
- Cada exercici té la mateixa puntuació
- Les metodologies de treball pròpies, organització personal i participació valen un 10%

Entrega:
- Un arxiu .zip anomenat: **PRx.y-NomCognom.zip**
  - PRx.y correspon al codi de la pràctica, per exemple PR1.1
  - NomCognom correspon al nom i primer cognom de cada participant
- L'arxiu **.zip** conte:
  - Aquest document emplenat en format **.pdf** anomenat **memoria.pdf**
  - Els arxius necessaris per fer anar la pràctica
- Esteu indicant l'enllaç al repositori Git

Nom i Cognom: Joel Berzal
Enllaç al vostre repositori Git: https://github.com/joelberzalgithub/PR1.3-BerzalJoel

Materials:
Necessiteu una eina per programar en JAVA
Feu servir Google per buscar els tutorials que us serveixin millor
El repositori bàsic és l'usat també en la pràctica anterior
https://github.com/optimisme/DAM-JavaPersitenciaFitxers

Tasques, a cada exercici feu l'explicació i captures que cregueu convenients.

- Preparació - Continueu afegint codi en el menú de java que vau preparar per la pràctica anterior:

```java
import java.io.IOException;

public class Main {
    static Scanner in = new Scanner(System.in); // System.in és global

    public static void main(String[] args) throws InterruptedException, IOException {

        boolean running = true;

        while (running) {

            System.out.println("Escull una opcio:\n\n0) PR120ReadFile\n1) PR121Files\n2) PR122cat\n3) PR123sobreescriu" +
                               "\n4) PR123append\n5) PR124linies\n6) PR125cp\n7) PR1310mainPersonesHashmap\n8) PR131mainEscriu" +
                               "\n9) PR131mainLlegeix\n10) PR132main\n11) PR133mainTreballadors\n12) PR134estudiantsManager\n100) Sortir\n");

            try {

                int opcio = Integer.valueOf(llegirLinia("Opcio: "));

                switch (opcio) {
                    case 0: PR120ReadFile.main(args);
                            break;
                    case 1: PR121Files.main(args);
                            break;
                    case 2: PR122cat.main(args);
                            break;
                    case 3: PR123sobreescriu.main(args);
                            break;
                    case 4: PR123append.main(args);
                            break;
                    case 5: PR124linies.main(args);
                            break;
                    case 6: PR125cp.main(args);
                            break;
                    case 7: PR130mainPersonesHashmap.main(args);
                            break;
                    case 8: PR131mainEscriu.main(args);
                            break;
                    case 9: PR131mainLlegeix.main(args);
                            break;
                    case 10: PR132main.main(args);
                            break;
                    case 11: PR133mainTreballadors.main(args);
                            break;
                    case 12: PR134estudiantsManager.main(args);
                            break;
                    case 100: running = false;
                            break;
                    default: System.out.println("\nOpcio fora del rang!");
                            break;
                }

                System.out.println("\n**********************************************************************************************************\n");

            } catch (Exception e) {
                System.out.println("\nOpcio no numerica!\n\n**********************************************************************************************************
            }
        }
        in.close();
    }

    public static String llegirLinia (String text) {
        System.out.print(text);
        return in.nextLine();
    }
}
```

- Exercici 0

PR130mainPersonesHashmap.java

- Crea un HashMap<String, Integer> amb el nom i l'edat de 5 persones (dades predefinides).
- Empra DataOutputStream per guardar aquestes dades en un arxiu **PR130persones.dat**.
- Llegeix **PR130persones.dat** amb DataInputStream i mostra el seu contingut per pantalla.

  (Mirar exemple EscripturaDadesPrimitives.java i LecturaDadesPrimitives.java)

```java
import java.io.ByteArrayOutputStream;
import java.io.ByteArrayInputStream;
import java.io.DataOutputStream;
import java.io.DataInputStream;
import java.io.File;
import java.io.FileOutputStream;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.ObjectOutputStream;
import java.io.ObjectInputStream;
import java.io.UnsupportedEncodingException;
import java.util.HashMap;
import java.util.Iterator;

public class PR130mainPersonesHashmap {

    public static void main(String[] args) {

        // Crea un HashMap<String, Integer> amb el nom i l'edat de 5 persones (dades predefinides)

        HashMap<String, Integer> persones = new HashMap<String, Integer>();

        persones.put("A", 10);
        persones.put("B", 20);
        persones.put("C", 30);
        persones.put("D", 40);
        persones.put("E", 50);

        String rutaBase = System.getProperty("user.dir") + "/data/";
        String rutaArxiu = rutaBase + "PR132people.dat";

        // Crear la carpeta 'data' si no existeix

        File dir = new File("data");

        if (!dir.exists()) {
            if (!dir.mkdirs()) {
                System.out.println("Error en la creacio de la carpeta 'data'");
            }
        }
        // Empra DataOutputStream per guardar aquestes dades en un arxiu PR130persones.dat

        try {

            FileOutputStream fos = new FileOutputStream(rutaArxiu);
            DataOutputStream dos = new DataOutputStream(fos);

            for (Iterator i = persones.keySet().iterator(); i.hasNext();) {

                String k = (String)i.next();
                Integer v = (Integer)persones.get(k);

                dos.writeUTF(k);
                dos.writeInt(v);
            }

            Objecte obj  = new Objecte("Escriptori", "Estudiar");
            writeSerializableObject(obj, dos);

            dos.flush();
            fos.close();
            dos.close();

        } catch (IOException e) { e.printStackTrace(); }

```

```java
        // Llegeix PR130persones.dat amb DataInputStream i mostra el seu contingut per pantalla

        try {

            FileInputStream fis = new FileInputStream(rutaArxiu);
            DataInputStream dis = new DataInputStream(fis);

            String nom_1 = dis.readUTF();
            int edat_1 = dis.readInt();

            String nom_2 = dis.readUTF();
            int edat_2 = dis.readInt();

            String nom_3 = dis.readUTF();
            int edat_3 = dis.readInt();

            String nom_4 = dis.readUTF();
            int edat_4 = dis.readInt();

            String nom_5 = dis.readUTF();
            int edat_5 = dis.readInt();

            Object obj = (Object) readSerializableObject(dis);

            System.out.println("Persona 1 [Nom: " + nom_1 + ", Edat: " + edat_1 + "]");
            System.out.println("Persona 2 [Nom: " + nom_2 + ", Edat: " + edat_2 + "]");
            System.out.println("Persona 3 [Nom: " + nom_3 + ", Edat: " + edat_3 + "]");
            System.out.println("Persona 4 [Nom: " + nom_4 + ", Edat: " + edat_4 + "]");
            System.out.println("Persona 5 [Nom: " + nom_5 + ", Edat: " + edat_5 + "]");

            fis.close();
            dis.close();

        } catch(IOException e) { e.printStackTrace(); }
    }

    public static void writeSerializableObject (Object obj, DataOutputStream dos) {

        try {

            // Transforma l'objecte a bytes[]

            ByteArrayOutputStream bos = new ByteArrayOutputStream();
            ObjectOutputStream oos = new ObjectOutputStream(bos);
            oos.writeObject(obj);
            oos.flush();
            byte[] data = bos.toByteArray();

            // Guarda la longitud de l'array
            dos.writeInt(data.length);

            // Guarda els bytes de l'objecte
            dos.write(data);

        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public static Object readSerializableObject (DataInputStream dis) {

        try {

            // LLegeix la longitud de l'array

            int longitud = dis.readInt();
            byte[] data = new byte[longitud];

            // LLegeix l'array que conté l'objecte
            dis.readFully(data, 0, longitud);

            // Transforma l'array de bytes en objecte

            ByteArrayInputStream bais = new  ByteArrayInputStream(data);
            ObjectInputStream ois = new ObjectInputStream(bais);
            return ois.readObject();

        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        } catch (UnsupportedEncodingException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return new java.lang.AbstractMethodError();
    }
}
```

- Exercici 1

Crea una classe **PR131hashmap** que implementa Serializable i conté un HashMap. Crea dos procediments:

- **PR131mainEscriu.java**: Escriu el HashMap a PR131HashMapData.ser.
- **PR131mainLlegeix.java**: Llegeix PR131HashMapData.ser i mostra el seu contingut per pantalla.

```java
import java.io.Serializable;
import java.util.HashMap;

public class PR131hashmap implements Serializable {

    public HashMap map = new HashMap();

    public PR131hashmap() {}

    public PR131hashmap(HashMap map) {
        super();
        this.map = map;
    }

    public HashMap getMap() {
        return map;
    }

    public void setMap(HashMap map) {
        this.map = map;
    }
}
```

```java
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.ObjectOutputStream;
import java.util.HashMap;
import java.util.Iterator;

public class PR131mainEscriu {

    public static PR131hashmap PR131 = new PR131hashmap();
    public static HashMap map = PR131.getMap();

    public PR131mainEscriu(PR131hashmap PR131) {
        super();
        this.PR131 = PR131;
    }

    public static HashMap getMap() {
        return map;
    }

    public static void main(String[] args) {

        String rutaBase = System.getProperty("user.dir") + "/data/";
        String rutaArxiu = rutaBase + "PR131HashMapData.ser";

        // Crear la carpeta 'data' si no existeix

        File dir = new File(rutaBase);
        if (!dir.exists()) {
            if (!dir.mkdirs()) {
                System.out.println("Error en la creacio de la carpeta 'data'");
            }
        }

        try {

            FileOutputStream fos = new FileOutputStream(rutaArxiu);
            ObjectOutputStream oos = new ObjectOutputStream(fos);

            for (Iterator i = getMap().keySet().iterator(); i.hasNext();) {

                String k = (String)i.next();
                String v = Integer.toString((Integer)getMap().get(k));

                oos.writeObject(new Objecte(k, v));
            }

            oos.close();
            fos.close();

        } catch (IOException e) { e.printStackTrace(); }
    }
}
```

```java
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.util.HashMap;
import java.util.Iterator;

public class PR131mainLlegeix {

    public static PR131hashmap PR131 = new PR131hashmap();
    public static HashMap map = PR131.getMap();

    public PR131mainLlegeix(PR131hashmap PR131) {
        super();
        this.PR131 = PR131;
    }

    public static HashMap getMap() {
        return map;
    }

    public static void main(String[] args) {

        String rutaBase = System.getProperty("user.dir") + "/data/";
        String rutaArxiu = rutaBase + "PR131HashMapData.ser";

        // Crear la carpeta 'data' si no existeix

        File dir = new File(rutaBase);
        if (!dir.exists()) {
            if (!dir.mkdirs()) {
                System.out.println("Error en la creacio de la carpeta 'data'");
            }
        }

        try {

            FileInputStream fis = new FileInputStream(rutaArxiu);
            ObjectInputStream ois = new ObjectInputStream(fis);

            for (Iterator i = getMap().keySet().iterator(); i.hasNext();) {
                System.out.println(ois.readObject());
            }

            ois.close();
            fis.close();

        } catch(ClassNotFoundException e) {
            e.printStackTrace();
        } catch(IOException e) { e.printStackTrace(); }
    }
}
```

- Exercici 2

Crea una clase **'PR132persona'** que implementi Serializable amb els atributs: Nom, Cognom, Edat. Després fes un programa "**PR132main.java**" amb objectes que tinguin les següents dades (que es mostren a la taula) i guarda'ls en un arxiu "**PR132people.dat**". Finalment llegeix l'arxiu que s'acaba de guardar i mostra la informació per pantalla:

Nom	**Cognom**	**Edat**
Maria	López	36
Gustavo	Ponts	63
Irene	Sales	54

```java
import java.io.Serializable;

public class PR132persona implements Serializable {

    public String nom;
    public String cognom;
    public int edat;

    public PR132persona(String nom, String cognom, int edat) {
        super();
        this.nom = nom;
        this.cognom = cognom;
        this.edat = edat;
    }

    public String getNom() {
        return nom;
    }

    public String getCognom() {
        return cognom;
    }

    public int getEdat() {
        return edat;
    }
}
```

```java
import java.io.File;
import java.io.FileOutputStream;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.ObjectInputStream;
import java.io.ObjectOutputStream;

public class PR132main {

    public static void main(String args[]) {

        String rutaBase = System.getProperty("user.dir") + "/data/";
        String rutaArxiu = rutaBase + "PR132people.dat";

        // Crear la carpeta 'data' si no existeix

        File dir = new File(rutaBase);
        if (!dir.exists()) {
            if (!dir.mkdirs()) {
                System.out.println("Error en la creacio de la carpeta 'data'");
            }
        }

        try {

            // Empra ObjectOutputStream per guardar aquestes dades en un arxiu PR132people.dat

            FileOutputStream fos = new FileOutputStream(rutaArxiu);
            ObjectOutputStream oos = new ObjectOutputStream(fos);

            oos.writeObject(new PR132persona("Maria", "Lopez", 36));
            oos.writeObject(new PR132persona("Gustavo", "Ponts", 63));
            oos.writeObject(new PR132persona("Irene", "Sales", 54));

            oos.close();
            fos.close();

            // Llegeix PR132people.dat amb ObjectInputStream i mostra el seu contingut per pantalla

            FileInputStream fis = new FileInputStream(rutaArxiu);
            ObjectInputStream ois = new ObjectInputStream(fis);

            PR132persona persona_1 = (PR132persona) ois.readObject();
            PR132persona persona_2 = (PR132persona) ois.readObject();
            PR132persona persona_3 = (PR132persona) ois.readObject();

            System.out.println("Persona 1 [Nom: " + persona_1.getNom() + ", Cognom: " + persona_1.getCognom() + ", Edat: " + persona_1.getEdat() + "]");
            System.out.println("Persona 2 [Nom: " + persona_2.getNom() + ", Cognom: " + persona_2.getCognom() + ", Edat: " + persona_2.getEdat() + "]");
            System.out.println("Persona 3 [Nom: " + persona_3.getNom() + ", Cognom: " + persona_3.getCognom() + ", Edat: " + persona_3.getEdat() + "]");

            ois.close();
            fis.close();

        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        } catch (IOException e) { e.printStackTrace(); }
    }
}
```

- Exercici 3

Crea un programa "**PR133mainTreballadors.java**", i crea manualment un arxiu anomenat "**PR133treballadors.csv**", amb les dades de la taula següent.

Id	**Nom**	**Cognom**	**Departament**	**Salari**
123	Nicolás	Rana	2	1000.00
435	Xavi	Gil	2	1800.50
876	Daniel	Ramos	6	700.30
285	Pedro	Drake	4	2500.00
224	Joan	Potter	6	1000.00

Fes que el programa demani a l'usuari un identificador de treballador, quina dada vol modificar i el nou valor i faci la modificació al propi arxiu .csv

```java
import java.io.BufferedReader;

public class PR133mainTreballadors {

    public static void main(String args[]) {

        try {

            Scanner sc = new Scanner(System.in);

            String rutaBase = System.getProperty("user.dir") + "/data/";
            String rutaArxiu = rutaBase + "PR133treballadors.csv";

            // Crear la carpeta 'data' si no existeix

            File dir = new File(rutaBase);
            if (!dir.exists()) {
                if (!dir.mkdirs()) {
                    System.out.println("Error en la creacio de la carpeta 'data'");
                }
            }

            // Demanar a l'usuari l'identificador del treballador

            int id = 0;
            boolean validId = false;

            while (!validId) {
                try {
                    System.out.print("\nEscriu l'identificador d'un treballador: ");
                    id = sc.nextInt();
                    validId = true;
                } catch (java.util.InputMismatchException e) {
                    System.out.println("L'identificador ha de ser un nombre enter");
                    sc.nextLine();
                }
            }

```

```java
            // Demanar a l'usuari una dada per modificar

            String dada = "";
            boolean validDada = false;

            while (!validDada) {

                System.out.print("\nQuina dada vols modificar? ");
                dada = sc.next();

                if (dada.equals("Nom") || dada.equals("Cognom") || dada.equals("Departament") || dada.equals("Salari")) {
                    validDada = true;
                }

                else {
                    System.out.println("Nomes hi han 4 tipus de dades: Nom, Cognom, Departament o Salari");
                    sc.nextLine();
                }
            }

            // Demanar a l'usuari el nou valor de la dada anterior

            String valor = "";
            boolean validValor = false;

            while (!validValor) {
                try {
                    System.out.print("\nEscriu el nou " + dada + ": ");
                    valor = sc.next();
                    if (dada.equals("Departament")) { Integer.parseInt(valor); }
                    else if (dada.equals("Salari")) { Float.parseFloat(valor); }
                    validValor = true;

                } catch (NumberFormatException e) {
                    System.out.println("Tant el departament com el salari han de ser valors numerics");
                    sc.nextLine();
                }
            }

            // Crear un fitxer temporal per guardar les dades modificades

            File arxiu = new File(rutaArxiu);
            File arxiuTemporal = new File(rutaBase + "PR133treballadors_temp.csv");

            BufferedReader br = new BufferedReader(new FileReader(rutaArxiu));
            BufferedWriter bw = new BufferedWriter(new FileWriter(arxiuTemporal));

            // Modificar la dada corresponent

            String line;

            while ((line = br.readLine()) != null) {

                String[] parts = line.split(",");

                if (id == Integer.parseInt(parts[0].trim())) {

                    if (dada.equalsIgnoreCase("Nom")) { parts[1] = valor; }
                    else if (dada.equalsIgnoreCase("Cognom")) { parts[2] = valor; }
                    else if (dada.equalsIgnoreCase("Departament")) { parts[3] = valor; }
                    else if (dada.equalsIgnoreCase("Salari")) { parts[4] = valor; }

                    line = String.join(",", parts);
                }
                bw.write(line);
                bw.newLine();
            }
            br.close();
            bw.close();

            // Eliminar l'arxiu original i canviar el nom del fitxer temporal

            arxiu.delete();
            arxiuTemporal.renameTo(arxiu);

        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
}
```

- Exercici 4 - Registre d'estudiants amb RandomAccessFile

Descripció:
Una universitat vol gestionar les notes dels seus estudiants de manera eficient. Cada estudiant té un número de registre únic (en format enter) i una nota final associada. Per permetre un accés ràpid a les dades i poder actualitzar-les sense haver de carregar tot el fitxer a memòria o recórrer-lo completament, decideixen utilitzar RandomAccessFile.

Requisits:

1. El fitxer d'estudiants ha de tenir una estructura amb una longitud fixa per registre, permetent l'accés directe a les dades de qualsevol estudiant. Es suggereix una estructura on:
    - El número de registre ocupa 4 bytes (ja que és un enter).
    - El nom ocupa 20 caràcters.
    - La nota ocupa 4 bytes (en format float).
2. El programa ha de permetre a l'usuari:
    - Afegir un nou estudiant amb la seva nota.
    - Actualitzar la nota d'un estudiant existent mitjançant el seu número de registre.
    - Consultar la nota d'un estudiant mitjançant el seu número de registre.
3. L'accés i modificació de les dades han de ser eficients, evitant recórrer tot el fitxer si no és necessari.

Consideracions addicionals:

- Cal gestionar possibles errors, com ara intentar accedir a un estudiant que no existeix.
- El programa ha de garantir que les dades introduïdes estiguin en el format correcte.
- Encara que per aquest exercici pugui suposar-se un límit d'estudiants, el codi ha de ser adaptable per gestionar un nombre més gran d'entrades si es requereix.

Podeu consultar aquest exemple i fer-lo servir com a base:
https://docs.google.com/document/d/1AKQYnn9CeWwXEmvM_61_OaeM0BB_EbK2gBemO6S_RX8/edit?usp=sharing

```java
import java.io.IOException;
import java.io.RandomAccessFile;

public class PR134estudiantsManager {

    private static final int RECORD_SIZE = 4; // Longitud màxima en bytes del número de registre
    private static final int NAME_SIZE = 20; // Longitud màxima en caràcters del nom
    private static final int GRADE_SIZE = 4; // Longitud màxima en bytes de la nota

    public static void afegirEstudiant(String rutaArxiu, Estudiant estudiant) {

        try (RandomAccessFile raf = new RandomAccessFile(rutaArxiu, "rw")) {

            long posicio = (estudiant.getRegistre() - 1) * (RECORD_SIZE + NAME_SIZE + GRADE_SIZE);
            raf.seek(posicio);
            raf.writeInt(estudiant.getRegistre());
            raf.writeChars(String.format("%-20s", estudiant.getNom()));
            raf.writeFloat(estudiant.getNota());

        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public static Estudiant consultarEstudiant(String rutaArxiu, int registre) {

        Estudiant estudiant = null;

        try (RandomAccessFile raf = new RandomAccessFile(rutaArxiu, "r")) {

            long posicio = (registre - 1) * (RECORD_SIZE + NAME_SIZE + GRADE_SIZE);
            raf.seek(posicio);
            int numeroRegistre = raf.readInt();

            if (numeroRegistre == registre) {
                char[] nameChars = new char[NAME_SIZE];
                for (int i = 0; i < NAME_SIZE; i++) {
                    nameChars[i] = raf.readChar();
                }
                String nom = new String(nameChars).trim();
                float nota = raf.readFloat();
                estudiant = new Estudiant(registre, nom, nota);
            }

        } catch (IOException e) {
            e.printStackTrace();
        }
        return estudiant;
    }

    public static void actualitzarNotaEstudiant(String rutaArxiu, int registre, float novaNota) {

        Estudiant estudiant = consultarEstudiant(rutaArxiu, registre);

        if (estudiant != null) {
            estudiant.setNota(novaNota);
            afegirEstudiant(rutaArxiu, estudiant);
        }
        else {
            System.out.println("L'estudiant amb el numero de registre " + registre + " no existeix");
        }
    }

    public static void main(String[] args) {

        // Afegir un estudiant

        Estudiant estudiant_1 = new Estudiant(1, "A", 2.5f);

        afegirEstudiant("./data/estudiants.dat", estudiant_1);

        // Consultar i mostrar un estudiant afegit

        Estudiant estudiantConsultat_1 = consultarEstudiant("./data/estudiants.dat", 1);

        if (estudiantConsultat_1 != null) {
            System.out.println("Estudiant trobat --> [Nom: " + estudiantConsultat_1.getNom() + ", Nota: " + estudiantConsultat_1.getNota() + "]");
        }
        else {
            System.out.println("L'estudiant no existeix");
        }

        // Consultar i mostrar un estudiant que no existeix

        Estudiant estudiantConsultat_2 = consultarEstudiant("./data/estudiants.dat", 2);

        if (estudiantConsultat_2 != null) {
            System.out.println("Estudiant trobat --> [Nom: " + estudiantConsultat_2.getNom() + ", Nota: " + estudiantConsultat_2.getNota() + "]");
        }
        else {
            System.out.println("L'estudiant no existeix");
        }

        // Actualitzar la nota d'un estudiant
        actualitzarNotaEstudiant("./data/estudiants.dat", 1, 5);

        // Actualitzar la nota d'un estudiant que no existeix
        actualitzarNotaEstudiant("./data/estudiants.dat", 2, 5);

        // Consultar i mostrar l'estudiant actualitzat

        Estudiant estudiantConsultat_3 = consultarEstudiant("./data/estudiants.dat", 1);
        if (estudiantConsultat_3 != null) {
            System.out.println("Estudiant trobat --> [Nom: " + estudiantConsultat_3.getNom() + ", Nota: " + estudiantConsultat_3.getNota() + "]");
        }
        else {
            System.out.println("L'estudiant no existeix");
        }
    }
}
```

```java

class Estudiant {

    public int registre;
    public String nom;
    public float nota;

    public Estudiant(int registre, String nom, float nota) {
        super();
        this.registre = registre;
        this.nom = nom;
        this.nota = nota;
    }

    public int getRegistre() {
        return registre;
    }

    public void setRegistre(int registre) {
        this.registre = registre;
    }

    public String getNom() {
        return nom;
    }

    public void setNom(String nom) {
        this.nom = nom;
    }

    public float getNota() {
        return nota;
    }

    public void setNota(float nota) {
        this.nota = nota;
    }
}
```