Nom i Cognom:	**Laura Toro**
Enllaç al repositori Git:	**https://github.com/laurattt/M06-Acceso-Datos.git**

Objectius:

- Aprendre a serialitzar objectes JAVA

Instruccions:

- Desenvolupa el codi necessari per a cada exercici seguint les especificacions indicades. El codi ha de superar els tests proporcionats per garantir-ne la funcionalitat.
- En cas d'haver de contestar alguna pregunta en aquest document, cal fer-ho dins del quadre indicat.

Criteris d'avaluació:

Cada exercici tindrà la mateixa puntuació. Es valorarà:

- Format correcte del codi (indentació i compliment de la guia d'estil de Java).
- Noms clars i descriptius per a mètodes i variables.
- Qualitat general del codi, amb comentaris explicatius quan sigui necessari.
- Les còpies seran penalitzades amb una puntuació de 0.

Entrega:

- Repositori Git privat, compartit amb l'usuari jpala4-ieti, que contingui el codi complet de la pràctica.
- Subdirectori "doc" dins del repositori amb el fitxer memoria.pdf.
- URL del repositori: S'ha de pujar a Moodle.

Recursos i materials:

- JDK de Java versió 21, Maven i Git instal·lats.
- Eina de programació: necessiteu una IDE per programar en Java (VS Code, IntelliJ, o una altra).
- Repositori amb exemples: https://github.com/jpala4-ieti/DAM-M0486-RA1-Repositori-Referencia-25-26 (llegiu el fitxer README.md per obtenir instruccions sobre com executar-lo).
- Altres recursos: Podeu utilitzar Google per buscar tutorials que us ajudin a resoldre els exercicis.

Punt de partida de la pràctica:

https://github.com/jpala4-ieti/DAM-M0486-RA1-PR1.2-Practica-Punt-Partida-25-26

Resol els exercicis proposats. El directori de treball ha de ser 'data', de la mateixa forma que es fa en el repositori d'exemples

```
String camiBase = System.getProperty("user.dir") + "/data/";
```

Exercici 0: Gestió de dades de persones amb HashMap i arxius binaris

Modifica el programa PR120mainPersonesHashmap.java. Des dins del programa, realitza les següents tasques:

- 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 PR120persones.dat.
- Llegeix PR120persones.dat amb DataInputStream i mostra el seu contingut per pantalla.
- La ruta del fitxer serà System.getProperty("user.dir") + "/data/PR120persones.dat".
- Si el fitxer no existeix o es produeix un error de lectura els mètodes han de llançar una excepció IOFitxerExcepcio amb un missatge adequat. El main ha de gestionar l'Excepció per evitar que l'execució es finalitzi de manera abrupta.

(Mirar exemple EscripturaDadesPrimitives.java i LecturaDadesPrimitives.java per orientació)

Dades a introduir i sortida esperada

```
Carla: 22 anys
Bernat: 30 anys
David: 35 anys
Anna: 25 anys
Elena: 28 anys
```

Exercici 1: Serialització d'un HashMap

Modifica el programa PR121mainLlegeix.java per gestionar la serialització i deserialització d'un HashMap. Realitza les següents tasques:

1. Crea una classe anomenada PR121hashmap que implementi Serializable i contingui un HashMap<String, Integer>.
2. Desenvolupa dos programes separats: a. PR121mainEscriu.java:
    - Crea una instància de PR121hashmap i omple el HashMap amb dades (per exemple, noms i edats).
    - Utilitza ObjectOutputStream per serialitzar l'objecte PR121hashmap i guardar-lo a un arxiu anomenat PR121HashMapData.ser.
    - La ruta del fitxer serà System.getProperty("user.dir") + "/data/PR121HashMapData.ser".
    - Gestiona adequadament les possibles excepcions.
3. b. PR121mainLlegeix.java:
    - Utilitza ObjectInputStream per llegir l'objecte serialitzat de l'arxiu PR121HashMapData.ser.
    - Mostra el contingut del HashMap per pantalla.
    - Generar l'excepció corresponent si el fitxer no existeix o es produeix un error de lectura.
4. Assegura't que ambdós programes utilitzin la mateixa ruta per al fitxer: System.getProperty("user.dir") + "/data/PR121HashMapData.ser"
5. Si hi ha problemes treballant amb els fitxer, els mètodes han de llançar una excepció IOFitxerExcepcio amb un missatge descriptiu. Implementa una gestió d'errors robusta per evitar que l'execució es finalitzi de manera abrupta en cas d'errors.

(Pots consultar els exemples EscripturaObjectes.java i LecturaObjectes.java per orientació sobre com implementar la serialització i deserialització)

Sortida esperada del programa de Llegir:

```
Carla: 22 anys
Bernat: 30 anys
Anna: 25 anys
```

Exercici 2: Serialització d'objectes Persona

Modifica el programa PR122main.java per gestionar informació de persones mitjançant serialització d'objectes. Segueix aquestes instruccions:

1. Crea una classe anomenada PR122persona que implementi Serializable amb els següents atributs:
   - Nom (String)
   - Cognom (String)
   - Edat (int)
2. Implementa els mètodes necessaris a la classe PR122persona:
   - Constructor
   - Getters i setters
   - toString() per a una correcta visualització de l'objecte
3. El programa principal PR122main.java ha de realitzar les següents tasques:

   a. Creació d'objectes. Crea objectes PR122persona amb les següents dades:

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

   b. Serialització:
   - Utilitza ObjectOutputStream per serialitzar una llista que conté els objectes PR122persona creats.
   - Guarda'ls en un arxiu anomenat PR122persones.dat.
   - La ruta del fitxer serà System.getProperty("user.dir") + "/data/PR122persones.dat".

   c. Deserialització i visualització:
   - Utilitza ObjectInputStream per llegir la llista d'objectes serialitzats de l'arxiu PR122people.dat.
   - Mostra la informació de cada persona per pantalla.

4. Implementa una gestió d'errors robusta:
   - Gestiona adequadament les possibles excepcions durant la serialització i deserialització.
   - Si el fitxer no existeix o es produeix un error de lectura ha de generar una excepció IOFitxerExcepcio amb el missatge adequat.

(Pots consultar els exemples EscripturaObjectes.java i LecturaObjectes.java per orientació sobre com implementar la serialització i deserialització). Sortida esperada:

```
Maria López, 36 anys
Gustavo Ponts, 63 anys
Irene Sales, 54 anys
```

Exercici 3: Gestió de dades de treballadors en CSV

Completa el programa PR123mainTreballadors.java per gestionar informació de treballadors emmagatzemada en un fitxer CSV. Segueix aquestes instruccions:

1. Examina el fitxer PR123treballadors.csv del directori "data" i comprova que conté les següents dades:

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

2. Modifica el programa principal PR123mainTreballadors.java que realitzi les següents tasques:
3. Lectura del fitxer CSV:
    - Pots utilitzar la classe UtilsCSV per llegir el contingut de l'arxiu PR123treballadors.csv.
    - La ruta del fitxer serà System.getProperty("user.dir") + "/data/PR123treballadors.csv".
    - Mostra el contingut per pantalla.
4. Interacció amb l'usuari:
    - Demana a l'usuari que introdueixi un identificador de treballador.
    - Demana quina dada vol modificar (Nom, Cognom, Departament o Salari).
    - Demana el nou valor per a la dada seleccionada.
    - Mostra la taula amb les modificacions i demana i la vols guardar al fitxer.
    - Localitza el treballador amb l'identificador proporcionat.
    - Guarda les modificacions a l'arxiu PR123treballadors.csv.
5. Implementa una gestió d'errors robusta:
    - Gestiona adequadament les possibles excepcions durant la lectura i escriptura del fitxer CSV.
    - Si el fitxer no existeix o es produeix un error de lectura/escriptura, els mètodes han de llançat una excepció IOFitxerExcepcio i el main l'ha de tractar.
6. Assegura't que el programa utilitza la mateixa ruta per al fitxer tant per llegir com per escriure: System.getProperty("user.dir") + "/data/PR123treballadors.csv"

(Pots consultar l'exemple GestioCSV.java per orientació sobre com treballar amb fitxers CSV)

Exercici 4: Registre d'estudiants amb RandomAccessFile

Moficica el programa PR124main.java per gestionar les notes dels estudiants d'una universitat utilitzant RandomAccessFile. Aquest mètode permet un accés ràpid i eficient a les dades sense necessitat de carregar tot el fitxer a memòria. Podeu mirar aquest exemple com a base RandomAccessFilesVideojocsManager.java (el teniu també en el repositori).

Descripció del problema

Una universitat necessita un sistema per gestionar les notes dels seus estudiants de manera eficient (Accés directe a dades per número de registre). Cada estudiant té un número de registre únic (en format enter) i una nota final associada.

Requisits del programa

1. Estructura del fitxer:
    - Utilitza RandomAccessFile per crear i gestionar un fitxer anomenat PR124estudiants.dat.
    - Cada registre d'estudiant ha de tenir una longitud fixa amb la següent estructura:
        - Número de registre: 4 bytes (enter)
        - Nom: 20 caràcters aprox (40 bytes, 2 bytes caràcter, UTF-8)
        - Nota: 4 bytes (float)
2. Funcionalitats del programa: Modificar el PR124main.java que permeti a l'usuari realitzar les següents accions:
        - Llistar els estudiants amb el seu número de registre.
        - Afegir un nou estudiant amb la seva nota.
        - Consultar la nota d'un estudiant mitjançant el seu número de registre.
        - Actualitzar la nota d'un estudiant existent mitjançant el seu número de registre.
3. Eficiència:
    - Utilitza els mètodes apropiats de RandomAccessFile per posicionar el punter de lectura/escriptura directament a la posició correcta del fitxer.
4. Gestió d'errors:
    - Implementa una gestió d'errors robusta per tractar situacions com:
        - Intentar accedir a un estudiant que no existeix.
        - Entrades de dades incorrectes per part de l'usuari.
    - Mostra missatges d'error apropiats (Veure secció missatges informatius més avall) sense que el programa finalitzi de manera abrupta.
5. Validació de dades:
    - Assegura't que les dades introduïdes estiguin en el format correcte:
        - El número de registre ha de ser un enter positiu.

- El nom no excedeixi el màxim fixat.
- La nota ha de ser un valor float entre 0 i 10.

Missatges informatius que ha de mostrar per pantalla

Cal que el programa informi adequadament del resultat de certes operacions.

```
// Missatge quan no hi ha estudiants registrats (fitxer buit o inexistent):
System.out.println("No hi ha estudiants registrats.");

// Missatge quan un estudiant s'ha afegit correctament:
System.out.println("Estudiant afegit correctament.");

// Missatge quan es consulta un estudiant existent:
System.out.println("Registre: " + registre + ", Nom: " + nom + ", Nota: " + nota);

// Missatge quan es vol actualitzar una nota existent:
System.out.println("Nota actualitzada correctament.");

// Missatge quan no es troba un estudiant quan es consulta per registre
System.out.println("No s'ha trobat l'estudiant amb registre: " + registre);
```

Consideracions addicionals

- Utilitza la següent ruta per al fitxer: System.getProperty("user.dir") + "/data/PR124estudiants.dat"
- Documenta adequadament el codi, especialment les parts relacionades amb la manipulació del RandomAccessFile.