Text

Description automatically generated

**EXAMEN**: PRIMER TRIMESTRE.

**MÓDULO**: ACCESO A DATOS.

**CURSO**: 2º DAM

**FECHA**: 21-NOV-2024

**DURACIÓN**:100 MINUTOS

**Instrucciones:**

* El examen lo debes realizar usando el IDE Eclipse con el que se ha trabajado en clase.
* Queda **prohibido la conexión a Internet por cualquier medio durante la realización** del examen, si se detecta conexión durante el examen, la puntuación es 0 de todo el examen.
* Cuando **acabes el examen avisa al profesor para que compruebe tu última** conexión, en su presencia deberás “subir” la solución del examen en la tarea creada.
* El examen se guardará en un documento que contendrá este nombre: **Apellidos\_Nombre\_Exa1Eva\_AD.docx**, si yo tuviera que guardar mi examen BorregoArmenteros\_Valentín\_Exa1Eva\_ASGBD.docx, este documento contendrá todas las explicaciones, resolución de los ejercicios y pantallazos necesarios.
* También deberá subir en un .zip con el siguiente formato **Apellidos\_Nombre\_Exa1Eva\_AD.zip** todas las clases (o Intefaces) Java que has creado poniendo el sufijo EjX, por ejemplo, en el ejercicio 1 debes llamar a la clase AlumnoEj1.java, puedes incluir el .docx en el .zip donde contienes las clases.
* Buenas práctica de Java y seguir la nomenclatura serán tenias en cuenta en la evaluación.
* Código que no esté comentado y debidamente explicado no será válido.

**Ejercicio 1 (2.5 puntos)**

Se desea implementar un programa en Java que lea los datos de alumnos de un fichero de texto (alumnos.txt). Cada línea del fichero contiene la información de un alumno con los siguientes campos separados por el carácter |:

1. Nombre (cadena de texto).
2. Grupo (cadena de texto).
3. Nota media en las prácticas en Acceso a Datos.
4. Nota de la primera evaluación.
5. Nota de la segunda evaluación.

Ejemplo fichero alumnos.txt:

Juan Perez|DAM1|7.5|6.0|8.5

Maria Lopez|DAM2|4.5|5.5|3.0

Carlos Sanchez|DAM1|6.0|5.0|7.0

Laura Torres|DAM2|8.5|9.0|7.5

Ana Ruiz|DAM1|3.5|4.0|5.5

1. Crear una clase AlumnoEj1 que almacene los datos de cada alumno.
2. Crear una clase principal MainEj1 con los siguientes métodos para realizar las siguientes operaciones:
   * Leer el fichero alumnos.txt y almacenar los datos en una lista de objetos de tipo Alumno.
   * Calcular y mostrar:
     + La media de aprobados en cada evaluación.
     + El número de suspensos en cada evaluación.
     + La nota media del módulo de "Acceso a Datos" por evaluación.

**Notas:**

* Considera que la nota mínima para aprobar es 5.0.
* Implementa el programa de forma modular y asegúrate de que sea fácil de mantener y entender.

La salida que yo obtengo es:

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

**Ejercicio 2 (3 puntos)**

Se desea, utilizando archivos binarios, gestionar los ciclos formativos de la rama de Informática y telecomunicaciones en IES Venancio Blanco de Salamanca.

Para ello se sabe que hay una clase llamada CFEj2.java que contiene lo siguientes atributos:

* Nombre del ciclo formativo (por ejemplo, Desarrollo de Aplicaciones Multiplataforma)
* Grado (este atributo indica si es de grado superior o grado medio)
* Número de horas del ciclo (los de grado superior son 2000 horas y grado medio 1800)

En otra clase llamada MainEj2.java deberás invocar objetos de la clase CFEj2.java y pasarle lo siguientes valores (ciclos que hay en el Venancio Blanco):

*Técnico en Sistemas Microinformáticos y Redes grado medio 1800 horas*

*Técnico Superior en Desarrollo de Aplicaciones Web grado superior 2000 horas*

*Técnico Superior en Desarrollo de Aplicaciones Multiplataforma grado superior 2000 horas*

*Técnico Superior en Administración de Sistemas Informáticos en Red**Multiplataforma grado superior 2000 horas.*

* Crea un método que obtenga el fichero llamado cfvb.ser
* Lee el archivo cfvb.ser y muestra su contenido por pantalla.

Pista, me sale una salida así:

Imagen que contiene Diagrama

Descripción generada automáticamente

**Ejercicio 3.**

**3.1 (2 puntos).**

Utilizando DOM leer el siguiente fichero XML (llamado hechos.xml):

*<?xml version="1.0" encoding="UTF-8"?>*

*<hechos\_historicos>*

*<hecho descripcion="IBM da a conocer el PC.">*

*<fecha>*

*<dia>12</dia>*

*<mes>8</mes>*

*<año>1981</año>*

*</fecha>*

*</hecho>*

*<hecho descripcion="Se funda Google.">*

*<fecha>*

*<dia>4</dia>*

*<mes>9</mes>*

*<año>1998</año>*

*</fecha>*

*</hecho>*

*<hecho descripcion="Se funda Facebook.">*

*<fecha>*

*<dia>4</dia>*

*<mes>2</mes>*

*<año>2004</año>*

*</fecha>*

*</hecho>*

*</hechos\_historicos>*

La salida es:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**3.2 (2.5 puntos)**

Se tiene el siguiente fichero llamado pom.xml (es parte de un fichero xml utilizado para el despliegue con Maven). Deberá le procesarse el fichero pom.xml utilizando SAX y deberá mostrarse la salida a continuación por pantalla.

**Fichero pom.xml**

<?xml version="1.0" encoding="UTF-8"?>

<build>

<plugins>

<!-- Plugin para compilar -->

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.11.0</version>

<configuration>

<source>${maven.compiler.source}</source>

<target>${maven.compiler.target}</target>

</configuration>

</plugin>

<!-- Plugin para empaquetar el JAR -->

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-jar-plugin</artifactId>

<version>3.3.0</version>

<configuration>

<archive>

<manifest>

<mainClass>com.ejemplo.Main</mainClass> <!-- Cambia a tu clase principal -->

</manifest>

</archive>

</configuration>

</plugin>

</plugins>

</build>

La salida es:

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente