ILERNA Online

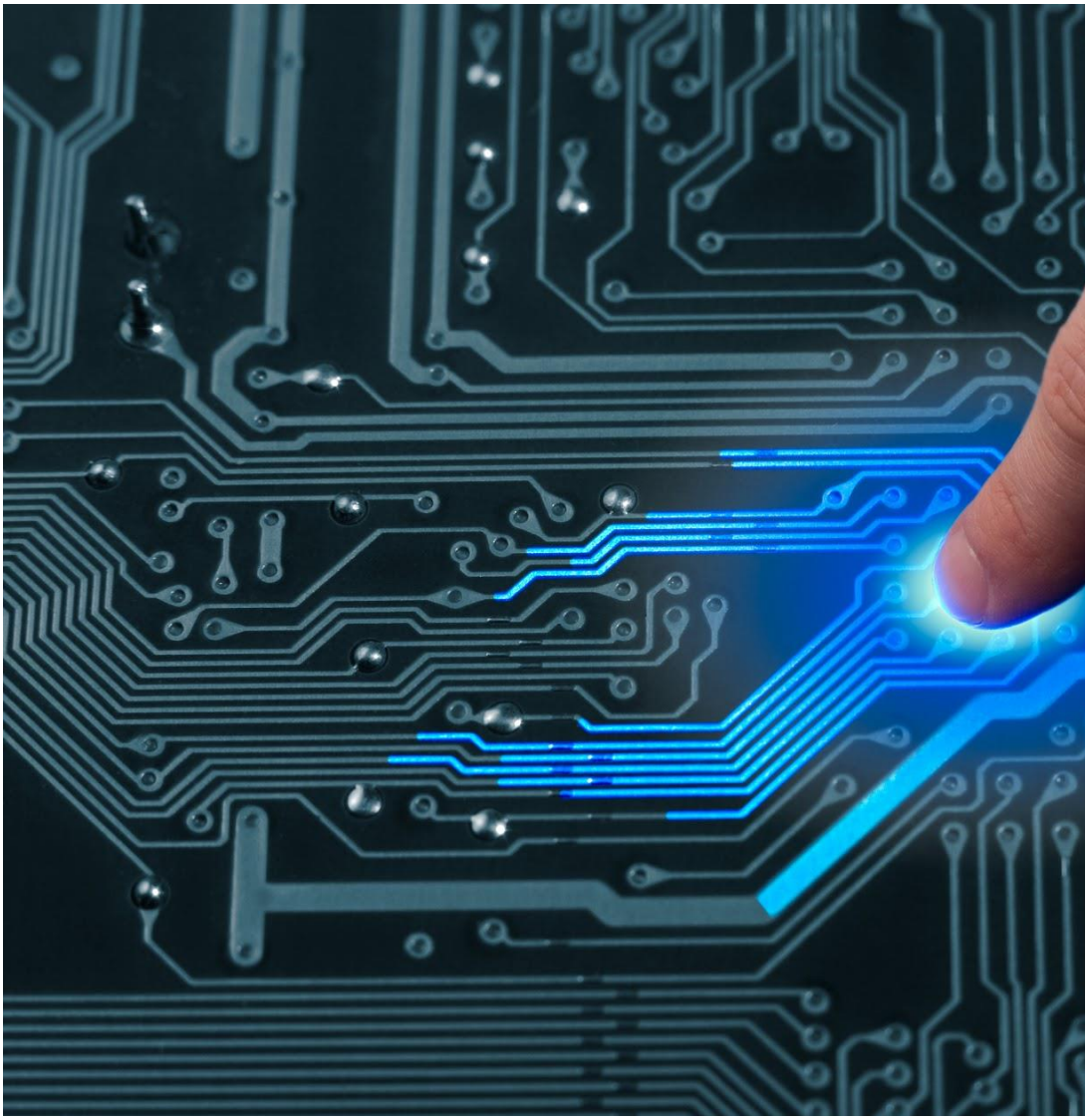
CFGS

DESARROLLO DE APLICACIONES MULTIPLATAFORMA

ACCESO A DATOS

PERSISTENCIA EN FICHEROS

PAC 1: Gestión de ficheros



PAC 1: Gestión de ficheros

INFORMACIÓN

Para responder a las siguientes cuestiones deberás ayudarte del material didáctico y consultar internet.

Requisitos varios que deben cumplirse en vuestros trabajos:

- En los ejercicios, si se requieren de cálculos, estos deben aparecer en la respuesta que planteéis.
- Siempre que utilicéis información de Internet para responder / resolver alguna pregunta, tenéis que citar la fuente (la página web) de dónde habéis sacado aquella información.
- Siempre que utilicéis información del libro digital para responder / resolver alguna pregunta, tenéis que citar el tema y la página de dónde habéis sacado aquella información.
- No se aceptaran respuestas sacadas de Internet utilizando la metodología de copiar y pegar. Podéis utilizar Internet para localizar información, pero el redactado de las respuestas ha de ser vuestro.
- Las respuestas a las preguntas deben estar bien argumentadas, no se admiten respuestas escuetas o monosílabas.
- Si el proyecto no compila, la puntuación máxima será un 4.
- Se valorará la presentación, ortografía y gramática de vuestro trabajo hasta con un punto de la nota final.

Entrega de la PAC:

- Se debe entregar junto con el proyecto comprimido, una memoria en el que se expliquen los pasos realizados junto con los puntos más importantes del código explicados.

1. Responde a este test en la matriz de respuestas que se encuentra al final de las preguntas:

1. ¿Cuál de estas opciones consideras una gran ventaja de los ficheros?
   a. Es volátil
   b. Permite control sobre los datos
   c. Mayor disponibilidad de los datos que en una base de datos
   d. Ninguna de las opciones anteriores es correcta

2. El método list() de la clase File devuelve
   a. Boolean
   b. String[]
   c. String
   d. File[]

3. ¿Con qué método de la clase File podemos conocer el número de líneas de un fichero de texto?
   a. isFile()
   b. length()
   c. exists()
   d. No existe un método para esto, es necesario recorrer el fichero

4. ¿Qué operación no podremos realizar en un fichero? Selecciona la opción d si consideras que todas las opciones son correctas.
   a. Alta
   b. Modificación
   c. Buscar
   d. Se pueden realizar todas las operaciones anteriores

5. ¿Cuál de estos métodos no pertenece a la clase FileReader?
   a. int read (char[] buf)
   b. int read (char[] buf, int desplazamiento, int numCaracteres)
   c. int read (byte[] b)
   d. Todas las opciones anteriores son métodos de la clase FileReader

Matriz de respuestas

1	2	3	4	5
D	B	D	D	C

2. Realiza una comparación de los archivos según su método de acceso.

- **Acceso secuencial**

Los datos del fichero se leen y se escriben de forma ordenada. Para ello, si queremos acceder a un dato que se encuentra en la mitad del fichero, es necesario leer todos los datos desde el principio del fichero. La clase utilizada para realizar esta búsqueda en Java depende del tipo de fichero que se vaya a utilizar. Si el fichero es binario entonces tenemos FileInputStream y FileOutputStream. En cambio, si el fichero es de texto entonces tenemos FileReader y FileWriter.

- **Acceso directo o aleatorio**

Para leer un dato del fichero, no es necesario leerlo entero, sino que accederemos directamente al registro buscado. La clase utilizada para realizar esta búsqueda en Java es RandomAccessFile.

3. ¿Qué diferencias encuentras entre los ficheros de texto y los ficheros binarios?

Los ficheros de texto contienen caracteres alfanuméricos en un formato estándar como puede ser ASCII, UNICODE o UFT8 mientras que los ficheros binarios contienen secuencias de dígitos binarios, que hacen ilegible el fichero al usuario. Por

ello, este
último
tipo de ficheros necesita una transformación del contenido para poder mostrarlo. Su ventaja es que ocupa menos espacio en disco.

Por ello, el tratamiento de estos ficheros es diferente. En Java tenemos diferentes clases para el trabajo con ellos.

4. Realiza un programa en Java que cree un fichero binario para guardar la información sobre los alumnos de Ilerna Online. Es necesario conocer el identificador, el nombre del alumno y el número de asignaturas escogidas.
El programa debe tener un menú que permita:

1- crear el fichero con nombre alumnos.dat

```
/*
 * Función de creación/apertura del archivo
 *
 */
public static RandomAccessFile createFile(String file) throws FileNotFoundException{
    //Creamos un fichero en la ruta que pasamos por parametro.
    File fichero=new File(file);

    //Creamos y retornamos un nuevo fichero de acceso aleatorio generado a partir del fichero anterior
    return new RandomAccessFile(fichero, "rw");

}
```

2- introducir un alumno

```
public static boolean createAlumno(RandomAccessFile file) throws IOException {
    Scanner scan = new Scanner(System.in);
    int id=0, asg=0, posicion;
    String nombre;

    //Leemos por teclado los datos del nuevo módulo

    System.out.println("Introduce el ID del alumno (un número entero mayor que 0)");
    id=scan.nextInt();

    posicion=(id-1)*48;

    System.out.println("Introduce nombre (String)(20 Caracteres Max)");
    nombre=scan.next();
    //Para evitarnos problemas en caso que nos ponga un nombre de módulo mayor de 20 carácteres
    //si es mayor de 20, nos quedamos únicamente con los 20 primeros
    if(nombre.length()>20)
        nombre=nombre.substring(0,20);

    System.out.println("Introduce el numero de asignaturas");
    asg=scan.nextInt();

    try {
        file.seek(posicion); //Como queremos añadir el nuevo registro al final del fichero, situamos el puntero en la última posición posible
        file.writeInt(id);//Escribimos el entero que hemos leido por teclado
         /*
         * Para el caso de las cadenas debemos
         * usar un StringBuffer para poder
         * escribirlas en el fichero
         */
        StringBuffer buffer=new StringBuffer(nombre); //Creamos el StringBuilder a partir del String que habiamos leido por teclado
        buffer.setLength(20); //Establecemos la longitud IMPORTANTE--> TODAS LAS CADENAS DEBEN TENER EL MISMO TAMAÑO.
        file.writeChars(buffer.toString()); //Escribimos la cadena en el archivo
        file.writeInt(asg); //Escribimos el número de asginaturas en el fichero.
        return true;
    } catch (IOException e) {
        return false;
    }
}
```

3- modificar el número de asignaturas matriculadas

```
/*
 * Esta función es la encargada de modificar el numero de alumnos de un modulo dado.
 */
public static void alterData(RandomAccessFile file) throws IOException {
    Scanner scan = new Scanner(System.in);
    int id, asg;

    System.out.println("Introduzca el identificador del alumno que desea modificar");
    id=scan.nextInt();
    System.out.println("Introduzca el nuevo número de alumnos");
    asg=scan.nextInt();

    //Buscamos en que línea se encuentra el Alumno


    int posicion=0;
    //Para situarnos en el registro que queremos realizamos la siguiente operacion.
    posicion=(id - 1)*48;

    if(posicion>=file.length()){
        System.out.println("El módulo solicitado no exite");
    }else{
        file.seek(posicion+24);
        file.writeInt(asg);
        file.close();
        System.out.println("Modificación realizada con éxito");
    }
}
```

4- eliminar un alumno

```
public static void deleteData(RandomAccessFile file) throws IOException {
    Scanner scan = new Scanner (System.in);
    int id, asg;

    System.out.println("Introduzca el identificador del alumno que desea eliminar");
    id=scan.nextInt();

    int posicion=0;
    //Para situarnos en el registro que queremos realizamos la siguiente operacion.
    posicion=(id - 1)*48;

    if(posicion>=file.length()){
        System.out.println("El módulo solicitado no exite");
    }else{
        if (file.readInt() != id) {
            System.out.println("El módulo solicitado no exite");
        } else {
            file.seek(posicion);
            file.writeInt(0);
            file.writeChars("");
            file.writeInt(0);
            System.out.println("Eliminación realizada con éxito");
        }
    }
}
```

5- mostrar datos

```
public static ArrayList<Alumno> showData (RandomAccessFile file) throws IOException {
    Alumno alumno = new Alumno();
    ArrayList<Alumno> lista = new ArrayList<Alumno>();
    int posicion = 0;
    String nombre = "";

    do {
        file.seek(posicion);

        int inicio = file.readInt();

        if (inicio > 0) {
            alumno.setId(inicio);
            for (int i = 0; i < 20; i++) {
                nombre += file.readChar();
            }
            alumno.setNombre(nombre);
            nombre = "";
            alumno.setNumAsg(file.readInt());

            lista.add(alumno);
            alumno = new Alumno();
        }

        posicion = posicion + 48;

    } while (posicion < file.length());

    return lista;
}
```

6- salir

```
file.close();
```

Utiliza la parte de excepciones para realizar todas las validaciones de datos que consideres oportunas.

¡Buen trabajo!