# UNLaM



Escuela de Formación Continua

## PROGRAMACIÓN AVANZADA I

**2do. Parcial**

Apellido y Nombre **05 / 07 / 2021**

### DNI:

1. Tomando como base las clases **DAO** y **AlumnoDAOTxt** vistas en clase, desarrollar el método ***insertar()*** y los atributos que el método utiliza.

### Suponer que se dispone de un método ***existe(Integer dni)***, no debe desarrollarlo, que le informa de la existencia o no de un alumno en el archivo.

(Utilizar tipos de datos genéricos)

Respuesta:

public class AlumnoDaoTXT extends DAO<Alumno, Integer> {

private static final String RWS\_MODE = "rws";

private RandomAccessFile raf;

public AlumnoDaoTXT(String pathFilename) throws DaoException {

try {

raf = new RandomAccessFile(new File(pathFilename), RWS\_MODE);

} catch (FileNotFoundException ex) {

Logger.getLogger(AlumnoDaoTXT.class.getName()).log(Level.SEVERE, null, ex);

throw new DaoException("Error de I/O =>" + ex.getLocalizedMessage());

}

}

@Override

public void create(Alumno alumno) throws DaoException {

try {

if (exist(alumno.getDni())) {

throw new DaoException("El alumno ya existe (DNI=" + alumno.getDni() + ")");

}

raf.seek(raf.length());

raf.writeBytes(alumno.toString() + System.lineSeparator());

} catch (IOException ex) {

Logger.getLogger(AlumnoDaoTXT.class.getName()).log(Level.SEVERE, null, ex);

throw new DaoException("Error al crear el alumno =>" + ex.getLocalizedMessage());

}

}

}

1. Dado el sig. código**,** indique cual/es de las afirmaciones son correctas:

@Override

public void update(Alumno alu) throws DAOException

{

archivoRandomAccessFile.seek(0); String linea;

String dniString; long posLinea = 0;

while((linea = archivoRandomAccessFile.readLine()) != null)

{

dniString = linea.substring(0, 8); if(Integer.valueOf(dniString).equals(alu.getDni()))

{

archivoRandomAccessFile.seek(posLinea); archivoRandomAccessFile.writeBytes(alu.toString()); return;

}

posLinea = archivoRandomAccessFile.getFilePointer();

}

}

### El objetivo de la variable ***posLinea*** es la de guardar la posición de fin de la línea de texto a actua- lizar para que ***archivoRandomAccessFile*** pueda continuar con la lectura en la siguiente línea.

### Esta afirmación es Correcta.

* 1. La línea de código ***archivoRandomAccessFile.seek(0)***; se utiliza para abrir el archivo de texto para comenzar a utilizarlo.
  2. El método ***getFilePointer()*** de ***RandomAccessFile*** me permite posicionar el puntero del archivo en la posición deseada.

### Esta afirmación es Correcta.

### Todas son correctas.

* 1. Ninguna es correcta.

1. Completar el siguiente código para que compile sin errores, y para que en tiempo de ejecución no arroje ninguna excepción:

try {

String sqlInsert = "INSERT INTO alumnos\n" +

"(dni, apellido, nombre, fecha\_nac, promedio, cant\_mat\_aprob)\n" + "VALUES (?,?,?,?);";

prepareStatementInsert = conn.prepareStatement(sqlInsert); prepareStatementInsert.setInt(index++, alu.getDni()); prepareStatementInsert.setString(index++, alu.getNombre()); prepareStatementInsert.setDate(index++,

MiCalendario.convert2SqlDate(alu.getFechaNac())); prepareStatementInsert.setInt(index++, alu.getCantMatAprob());

prepareStatementInsert.executeUpdate();

} catch (SQLException ex) { Logger.getLogger(AlumnoDAOSQL.class.getName()).log(Level.SEVERE, null, ex);

}

Respuesta:

try {

String sqlInsert = "INSERT INTO alumnos\n" +

"(dni, apellido, nombre, fecha\_nac, promedio, cant\_mat\_aprob)\n" +

"VALUES (?,?,?,?,?,?);";

prepareStatementInsert = conn.prepareStatement(sqlInsert);

prepareStatementInsert.setInt(1, alu.getDni());

prepareStatementInsert.setString(2, alu.getApellido());

prepareStatementInsert.setString(3, alu.getNombre());

prepareStatementInsert.setDate(4, MiCalendario.convert2SqlDate(alu.getFechaNac()));

prepareStatementInsert.setDouble(5, alu.getPromedio());

prepareStatementInsert.setInt(6, alu.getCantMatAprob());

prepareStatementInsert.executeUpdate();

} catch (SQLException ex) {

Logger.getLogger(AlumnoDAOSQL.class.getName()).log(Level.SEVERE, null, ex);

}

1. Completar dando los valores a los **‘?#’** según corresponda:

public class DAOAlumnoFactory {

public static final String TIPO\_DAO = "TIPO\_DAO"; public static final String DAO\_TXT = "DAO\_TXT"; public static final String DAO\_SQL = "DAO\_SQL"; public static final String FILE\_NAME = "FILE\_NAME";

public static final String SQL\_CONNECTION = "SQL\_CONNECTION";

public DAO crearDAO(**?1**<String, String> config) **?2** DAOException { String tipo = config.get(TIPO\_DAO);

switch (tipo){ case **?3**:

String filename = config.get(FILE\_NAME); return new AlumnoDAOTXT(**?4**);

case DAO\_SQL:

return new AlumnoDAOSQL(config.get(**?5**), "root", "root"); default:

throw new **?6**("Tipo de DAO no implementado");

}

}

}

Respuesta:

public class DAOAlumnoFactory {

public static final String TIPO\_DAO = "TIPO\_DAO";

public static final String DAO\_TXT = "DAO\_TXT";

public static final String DAO\_SQL = "DAO\_SQL";

public static final String FILE\_NAME = "FILE\_NAME";

public static final String SQL\_CONNECTION = "SQL\_CONNECTION";

public DAO crearDAO(Map<String, String> config) throws DAOException {

String tipo = config.get(TIPO\_DAO);

switch (tipo) {

case DAO\_TXT:

String filename = config.get(FILE\_NAME);

return new AlumnoDAOTXT(filename);

case DAO\_SQL:

return new AlumnoDAOSQL(config.get(SQL\_CONNECTION), "root", "root");

default:

throw new DAOException("Tipo de DAO no implementado");

}

}

}

### Desarrollar una clase que cumpla con el patrón de diseño **Singleton**

### Respuesta:

public class Singleton {

private static final Singleton instance;

private Singleton() {

}

public static Singleton getInstance() {

if(instance == null) {

instance == Singleton();

}

return instance;

}

}