PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
FACULTAD DE CIENCIAS E INGENIERÍA

PROGRAMACIÓN 3
5ta. práctica (tipo b)
(Segundo Semestre 2024)

Indicaciones Generales:
- Tiempo estimado: 1h 50 minutos
- Se les recuerda que, de acuerdo al reglamento disciplinario de nuestra institución, constituye una falta grave copiar del trabajo realizado por otro estudiante o cometer plagio para el desarrollo de esta práctica.
- Para el desarrollo de toda la práctica debe utilizar el sistema operativo Windows y el **JDK 21**.
- Para el desarrollo de las Preguntas Nro. 1, 2 y 3 puede utilizar únicamente Notepad++/Sublime y consola CMD. Está prohibido el uso de VISUAL STUDIO CODE o de un entorno de desarrollo integrado como NETBEANS.
- Debe hacer uso del entorno de desarrollo integrado de **NETBEANS 22** para el desarrollo de la Pregunta Nro. 4.
- Está permitido el uso de apuntes de clase, diapositivas, ejercicios de clase y código fuente. (Debe descargarlos antes de iniciar con la solución del enunciado)
- Está permitido el uso de Internet (únicamente para consultar páginas oficiales de Microsoft y Oracle). No obstante, está prohibida toda forma de comunicación con otros estudiantes o terceros.

PARTE PRÁCTICA (20 puntos)

PUEDE UTILIZAR MATERIAL DE CONSULTA.
Antes de comenzar el laboratorio, descargue todos los proyectos, apuntes, diapositivas que utilizará.
Se considerará en la calificación el uso de buenas prácticas de programación (aquellas vistas en clase).

PREGUNTA 1: Creación de un motor de base de datos MySQL en AWS Academy (3 puntos)

Se le solicita ingresar al servicio de AWS Academy (**https://www.awsacademy.com/vforcesite/LMS_Login**) y crear una instancia o motor de base de datos MySQL con los siguientes parámetros:
- Método de creación de base de datos: **Creación estándar**
- Tipo de motor: **MySQL**
- Edición: **Comunidad de MySQL**
- Versión del motor: **MySQL 8.0.39**
- Plantilla: **Capa gratuita**
- Identificador de instancias de bases de datos: **labs-1inf30-prog3-20242**
- Nombre de usuario maestro: **admin**
- Master password: **prog320242labs**
- Clase de instancia de base de datos: **db.t3.micro (2 vCPUs, 1 GiB RAM)**
- Capacidad de disco duro para la BD: **20 GiB SSD (disco sólido) sin posibilidad de escalado automático**
- Acceso público: **SI**
- Puerto de la base de datos: **3306**
- Nombre de la base de datos inicial: **lab04**
- Habilitar el cifrado: **NO**

- **Configure las reglas de entrada para que cualquier IP o computador tenga acceso a su base de datos.**
- **IMPORTANTE: El motor de base de datos debe ser configurado para que cualquier computador o IP pueda acceder al mismo. Si esta configuración no es realizada y no es posible conectarse a su motor de base de datos desde la computadora de los jefes de práctica o docente, entonces no se considerará puntaje alguno.**

Una vez creada su base de datos, se le solicita comprobar que funciona correctamente. Se le solicita dejar la base de datos **ACTIVA** en AWS y colocar los datos de conexión en un archivo llamado **Pregunta01_códigoAlumno.txt** (en el archivo deberá colocar el *hostname*, el *username*, el *password* y el nombre de la BD o esquema). Compruebe que es posible acceder a su base de datos creada utilizando MySQL Workbench y **suba el archivo .txt a PAIDEIA**.

PREGUNTA 2: Ejercicio de paquetes (Sin uso de IDE) (6 puntos)
Como parte de un proceso de reingeniería, se ha establecido modificar la estructura de paquetes inicial de las clases del software que ha desarrollado en el Laboratorio 02.

En la Fig. 01, se muestra a través de un diagrama de clases, la nueva estructura de paquetes.
Se le solicita descargar los archivos fuente que se adjuntan a este enunciado (**no aquellos que se encuentran en la Semana 02**). Agregue las instrucciones **package** e **import** a todas las clases, **modifique los ámbitos** correspondientes de las mismas, y **genere la estructura de carpetas** para que el programa JAVA funcione bajo la nueva estructura de paquetes solicitada. (**Recuerde que cada paquete en JAVA representa contar con una carpeta a nivel de estructura**

de directorios). Compruebe que es posible compilar y ejecutar bajo la nueva estructura. Está prohibido utilizar el comodín * para realizar importaciones en las clases. Para compilar, sí puede utilizar el *.
Se descontarán puntos por importaciones innecesarias.

Para un mejor entendimiento, a continuación, se detallan los paquetes y las clases que deberían contener:

- **pe.edu.pucp.eventmastersoft.contratos.model**: IDataProvider, TipoConcierto.
- **pe.edu.pucp.eventmastersoft.logistica.model**: InfoProvider, Local, TipoLocal, Artista, Funcion.
- **pe.edu.pucp.eventmastersoft.gestion.model**: Evento, Productora, ObraTeatral, Concierto.
- **pe.edu.pucp.eventmastersoft.program.main**: Principal.

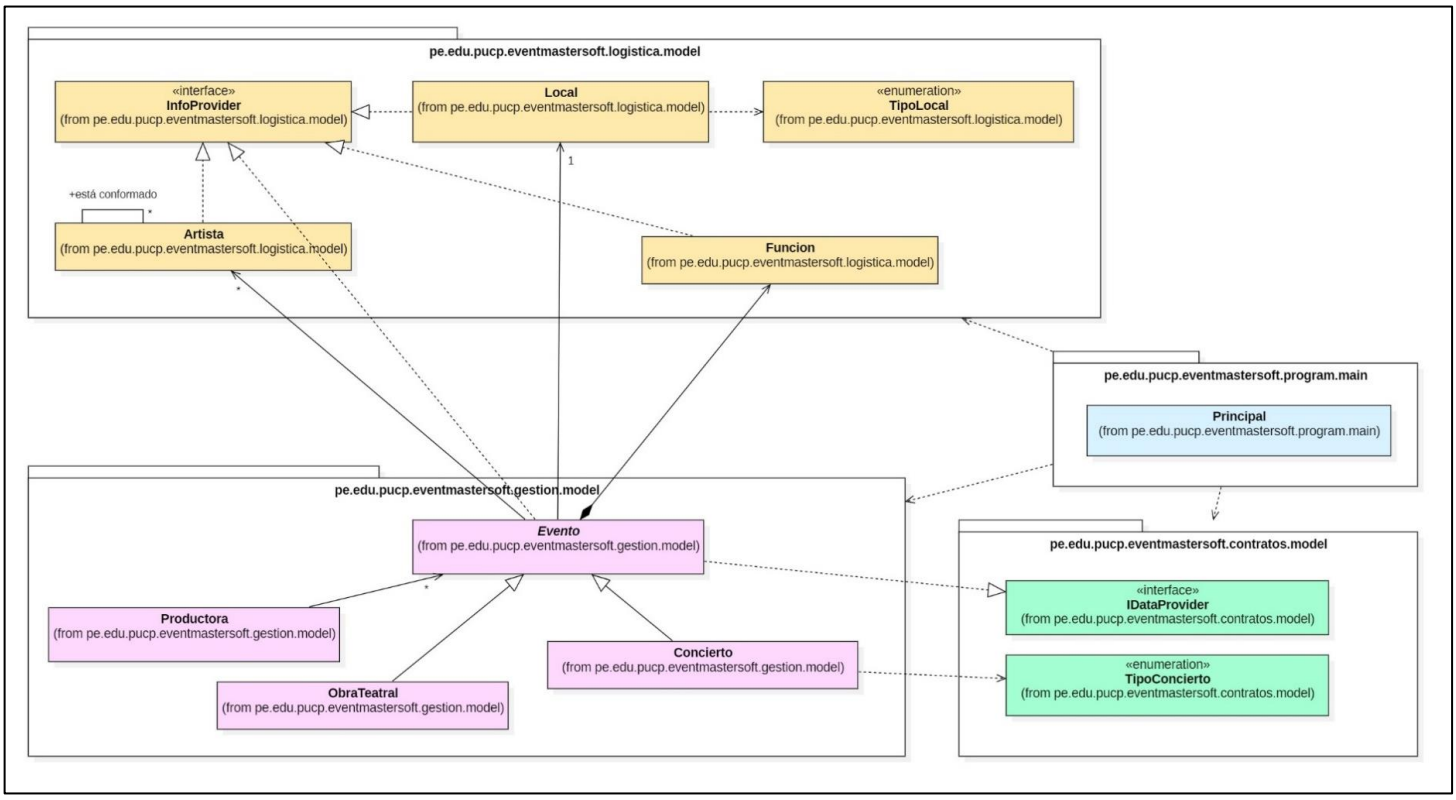

Fig. 01. Diagrama de clases – estructura de paquetes solicitada.

Con las siguientes instrucciones puede corroborar que su programa se ha estructurado y funciona bajo el nuevo esquema:

```
javac pe/edu/pucp/eventmastersoft/contratos/model/*.java
pe/edu/pucp/eventmastersoft/logistica/model/*.java
pe/edu/pucp/eventmastersoft/gestion/model/*.java
pe/edu/pucp/eventmastersoft/program/main/*.java

java -cp
pe/edu/pucp/eventmastersoft/contratos/model/*.class;pe/edu/eventmastersoft/logistica/mo
del/*.class;pe/edu/pucp/eventmastersoft/gestion/model/*.class;
pe/edu/pucp/eventmastersoft/program/main/Principal
```

Antes de continuar con la Pregunta 03, suba un archivo .zip a PAIDEIA que contenga TODA esta nueva estructura (con toda la estructura de carpetas y todos los paquetes que se visualizan en el diagrama).

PREGUNTA 3: Ejercicio de generación de librerías JAR (Sin uso de IDE) (6 puntos)
Para continuar con este ejercicio, **borre todos los archivos .class generados en el ejercicio anterior en todos los paquetes.** Realice las instrucciones en estricto orden e **indique en un archivo llamado** "**Pregunta03_codigoAlumno.txt**", **las líneas de comandos utilizadas por consola** para alcanzar cada actividad:

1. Compile únicamente las clases contenidas en **pe.edu.pucp.eventmastersoft.contratos.model** desde el directorio raíz de trabajo.
2. Genere un componente .jar llamado "**eventmastersoftContratosModel.jar**" que contenga la estructura y clases en Bytecode contenidas en: **pe.edu.pucp.eventmastersoft.contratos.model**.
Una vez creado el componente, borre todos los archivos .java, .class, y estructura de carpetas que corresponden al componente creado "**eventmastersoftContratosModel.jar**" (**pe.edu.pucp.eventmastersoft.contratos.model**).
Prosiga con las siguientes instrucciones.

3. Compile las clases contenidas en **pe.edu.pucp.eventmastersoft.logistica.model** desde el directorio raíz de trabajo.
4. Genere un componente .jar llamado "**eventmastersoftLogisticaModel.jar**" que contenga la estructura y clases en Bytecode contenidas en: **pe.edu.pucp.eventmastersoft.logistica.model**.
Una vez creado el componente, borre todos los archivos .java, .class, y estructura de carpetas que corresponden al componente creado "**eventmastersoftLogisticaModel.jar**" (**pe.edu.pucp.eventmastersoft.logistica.model**). Prosiga con las siguientes instrucciones.

5. Compile las clases contenidas en **pe.edu.pucp.eventmastersoft.gestion.model** desde el directorio raíz de trabajo utilizando los .jar previamente creados: "**eventmastersoftContratosModel.jar**" y "**eventmastersoftLogisticaModel.jar**".
6. Genere un componente .jar llamado "**eventmastersoftGestionModel.jar**" que contenga la estructura y clases en Bytecode contenidas en: **pe.edu.pucp.eventmastersoft.gestion.model**.
Una vez creado el componente, borre todos los archivos .java, .class, y estructura de carpetas que corresponden al componente creado "**eventmastersoftGestionModel.jar**" (**pe.edu.pucp.eventmastersoft.gestion.model**). Prosiga con las siguientes instrucciones.

7. Compile las clases contenidas en **pe.edu.pucp.eventmastersoft.program.main** desde el directorio raíz de trabajo utilizando los .jar previamente creados: "**eventmastersoftContratosModel.jar**", "**eventmastersoftLogisticaModel.jar**" y "**eventmastersoftGestionModel.jar**".
8. Genere un componente **.jar ejecutable** llamado "**eventmastersoftPrincipal.jar**" que contenga la estructura y clases en Bytecode contenidas en **pe.edu.pucp.eventmastersoft.program.main**. Comprueba que funcione. Prosiga con la siguiente instrucción.

9. Además de las instrucciones ejecutadas, incorpore en el archivo "**Pregunta03_codigoAlumno.txt**" el contenido que debería tener el **MANIFEST** de "**eventmastersoftPrincipal.jar**" para que pueda funcionar como componente ejecutable.

Compruebe que el componente "**eventmastersoftPrincipal.jar**" es ejecutable mediante el siguiente comando:

```
java -jar eventmastersoftPrincipal.jar
```

Suba a PAIDEIA el archivo de texto o documento de texto con todas las instrucciones utilizadas por consola para alcanzar las actividades solicitadas. Además, adjunte también todos los archivos JARs generados y el MANIFEST.MF utilizado para la creación de eventmastersoftPrincipal.jar.

PREGUNTA 4: Conexión a Base de Datos (Con uso de IDE) (5 puntos)
Descargue los proyectos en JAVA que se adjuntan a este enunciado (utilice Netbeans para abrir los proyectos). Asimismo, descargue el Script SQL que se adjunta en PAIDEIA y ejecútelo en el esquema del motor de base de datos (BD) que ha instanciado en AWS como parte de la pregunta Nro. 1. Este Script generará la tabla "**local**" que se visualiza en la Fig. 02.

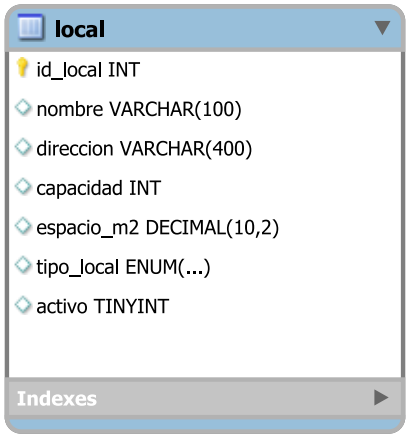

Fig. 02. Diagrama EER de la Base de Datos

Complete la implementación en los proyectos JAVA para que sea posible registrar un local, cuyos datos sean leídos desde consola. Realice los siguientes pasos:

- En el proyecto **EventMasterSoftDBManager**, dentro de la clase **DBManager**, coloque sus datos de conexión en los atributos *url*, *username*, *password*.
- En el paquete "**pe.edu.pucp.eventmastersoft.lib**" del proyecto **EventMasterSoft** coloque el archivo JAR: "**eventmastersoftLogisticaModel.jar**" que se generó en la pregunta 03.
- Agregue la referencia con **ruta relativa** al archivo **JAR** que acaba de agregar en el proyecto **EventMasterSoft.**
- En el paquete "**pe.edu.pucp.eventmastersoft.logistica.lib**" del proyecto **EventMasterSoftLogisticaDA** coloque el archivo JAR: " **eventmastersoftLogisticaModel.jar**" que se generó en la pregunta 03.
- Agregue la referencia con **ruta relativa** al archivo **JAR** que acaba de agregar en el proyecto **EventMasterSoftLogisticaDA**.
- Coloque el JAR del driver de conexión en el paquete "**pe.edu.pucp.eventmastersoft.lib**" del proyecto **EventMasterSoft**.

- Agregue la referencia con ruta relativa al JAR del driver de conexión que acaba de agregar en el proyecto **EventMasterSoft**.
- Complete la implementación de la clase **LocalDAO** en el proyecto **EventMasterSoftLogisticaDA** para permitir únicamente el registro.
- Complete la implementación de la clase **LocalMySQL** en el proyecto **EventMasterSoftLogisticaDA** para permitir únicamente el registro. Puede utilizar la clase **Statement** o la clase **PreparedStatement**.
- Finalmente, complete y modifique la implementación del método **`main()`** de la clase **Principal** en el proyecto **EventMasterSoft** para permitir el registro en BD de un local cuyos datos se leen por consola. El programa debe funcionar como se muestra en la Figura 03.

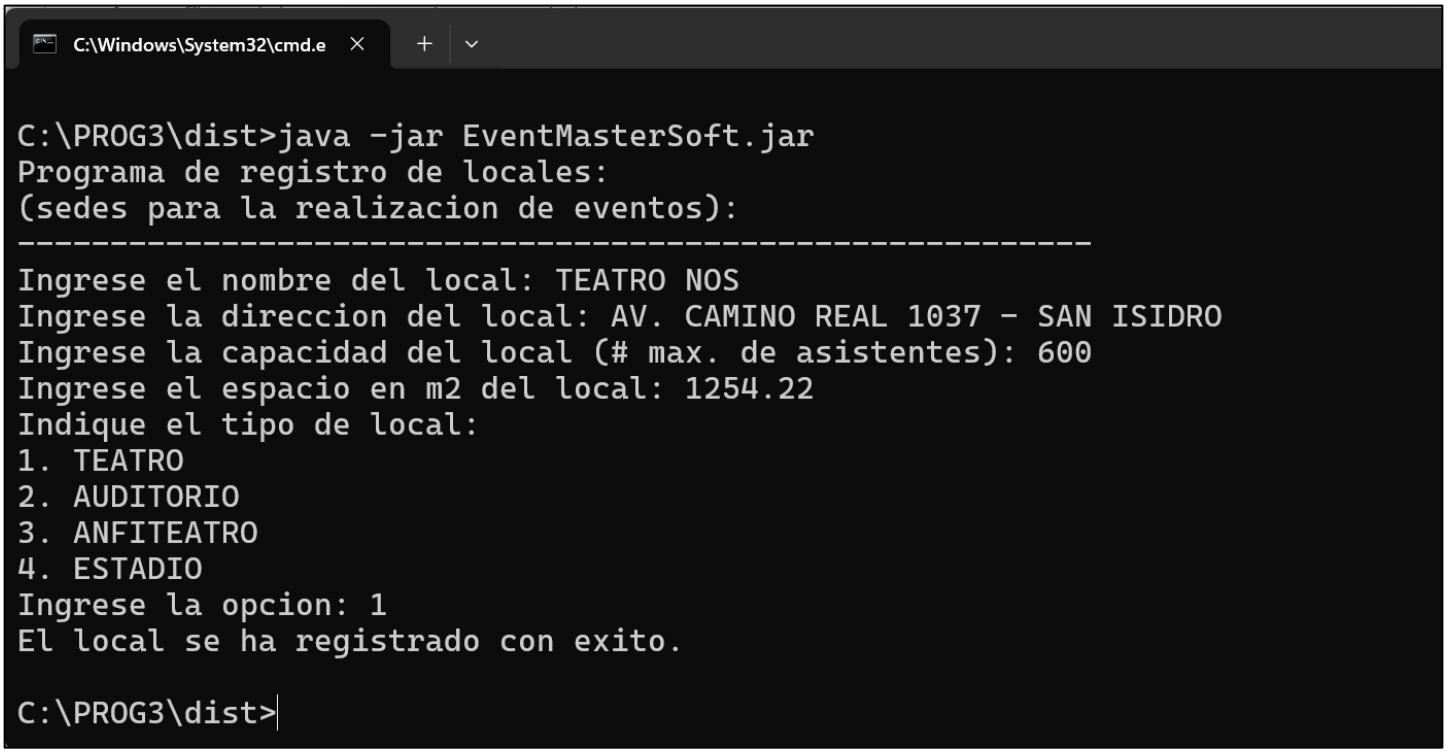

Fig. 03. Salida del programa solicitado

Asuma que el usuario al momento de ejecutar el programa escribe correctamente los valores. No es necesario programar validaciones.

Verifique que el programa realiza correctamente el registro ejecutando en su BD una sentencia de tipo SELECT a la tabla. Coloque sus datos personales en la clase Principal del proyecto **EventMasterSoft.**

Para convertir de String a enumerado puede utilizar la siguiente línea de programación

```
TipoLocal tipoLocal =
TipoLocal.valueOf(opTipoLocal==1?"TEATRO":(opTipoLocal==2?"AUDITORIO":(opTipoLocal==3?"
ANFITEATRO":"ESTADIO")));
```

Suba a PAIDEIA en un archivo .zip todos los proyectos JAVA que involucran su propuesta de solución.

Los siguientes aspectos pueden conllevar a un descuento significativo en su propuesta de solución:

- No seguir las instrucciones establecidas en el enunciado.
- Que los programas no compilen.
- Que los programas presenten errores en tiempo de ejecución.
- Que los proyectos tengan referencias innecesarias.
- Que el driver de conexión a base de datos no se haya adjuntado al proyecto.
- Que el driver de conexión a base de datos no se encuentre referenciado con ruta relativa en el proyecto.
- No seguir las prácticas vistas en clase: patrón DAO, desarrollo por componentes en capas, etc.

18 de setiembre del 2024