

## Colecciones

17 de 17 puntos

✓ **El Java Collection Framework es: \*** 2/2

- ☐ Un conjunto de objetos
- ☒ Un conjunto de clases e interfaces
- ☐ Una base de datos



✓ **Qué puede almacenar una colección? \*** 2/2

- ☐ Datos primitivos, como char e int
- ☐ Objetos mezclados sin importar de qué clase son
- ☒ Objetos de la misma clase



✓ **Los conjuntos o Sets pueden contener elementos repetidos, simplemente se duplica el valor** \*2/2

- ☐ Verdadero
- ☒ Falso



✓ **Cómo se define el tamaño de una colección? \*** 3/3

- ☐ Con el método size()
- ☒ Las colecciones son dinámicas, su tamaño depende de la cantidad de objetos que le agreguemos o quitemos
- ☐ Se define al declarar la colección



✓ **Las listas son colecciones de objetos ordenados por posición donde los elementos pueden repetirse.** \*2/2

☒ Verdadero



☐ Falso

✓ **Los mapas son colecciones de Llave/Valor. Las llaves son únicas pero los valores sí pueden repetirse.** \*2/2

☐ Falso

☒ Verdadero



✓ **Cuál de las siguientes herramientas nos facilita recorrer una lista?** \*2/2

☐ Bucle For

☐ Un Switch con condicionales en cada caso

☒ Bucle ForEach



✓ **Un framework es un marco de trabajo el cual contiene un conjunto estandarizado de conceptos, prácticas y criterios para hacer frente a un tipo de problemática particular y resolver nuevos problemas de índole similar.** \*2/2

☐ Falso

☒ Verdadero



✓ **Las relaciones entre clases realmente significan que una clase <sup>\*2/2</sup> contiene una referencia a un objeto u objetos, de la otra clase en la forma de un atributo.**

☒ Verdadero



☐ Falso

✓ **Las clases no actúan aisladas entre sí, al contrario, las clases <sup>\*2/2</sup> están relacionadas unas con otras.**

☐ Falso

☒ Verdadero



✓ **La composición es un tipo de relación donde un objeto <sup>\*2/2</sup> complejo es conformado por objetos más pequeños.**

☒ Verdadero



☐ Falso

✓ **Las relaciones son siempre unidireccionales <sup>\*</sup> 2/2**

☒ Falso



☐ Verdadero



✓ **En una relación de agregación, un objeto depende del otro, no pueden existir individualmente.** \*2/2

- ☒ Falso ✓
- ☐ Verdadero

✓ **Las relaciones pueden ser \*** 2/2

- ☒ Uno a Uno, Uno a Muchos, Muchos a Muchos, Muchos a Uno ✓
- ☐ Uno a Uno solamente
- ☐ Uno a Uno y Muchos a Muchos

## Herencia

15 de 15 puntos

✓ **La Sub Clase hereda atributos y métodos. \*** 1/1

- ☒ Verdadero ✓
- ☐ Falso

✓ **Qué hace la anotación @Override? \*** 1/1

- ☐ Desbloquea nuevos métodos disponibles en Java
- ☐ Permite que la clase hija elimine métodos que considere innecesarios
- ☒ Permite que la clase hija haga funcionar un método de manera diferente a la clase padre ✓



✓ Una clase hija puede heredar de muchas clases padres. \*

1/1

☐ Verdadero

☒ Falso



✓ En el siguiente código ¿cuál es la clase padre? \*

3/3

```
1 public class Persona extends SerVivo{
2
3     String nombre;
4
5 }
```

☐ Persona

☐ Java.util

☒ SerVivo

☐ public class



✓ Una clase padre puede heredar a muchas clases hijas. \*

2/2

☒ Verdadero

☐ Falso



✓ **Qué es la Super Clase? \***

2/2

- ☐ Una clase con superpoderes
- ☐ Una clase con muchos métodos
- ☒ Como se llama a la clase superior(Padre o Madre)



✓ **Cuándo usamos la palabra reservada super? \***

2/2

- ☐ Cuando queremos referirnos a una clase que está muy bien hecha
- ☐ Cuando queremos acceder a los métodos, atributos y constructores de la clase inferior
- ☒ Cuando queremos acceder a los métodos, atributos y constructores de la clase superior



✓ **La Sub Clase hereda el constructor \***

1/1

- ☒ Falso
- ☐ Verdadero



✓ **Para qué sirve el modificador Protected? \***

2/2

- ☒ Para que los atributos sean accesibles sólo por las clases que heredan sin usar getters ni setters
- ☐ Para que suene a que sabemos más que los demás
- ☐ Para proteger el código para que no sea borrado



## Manejo de Excepciones

13 de 13 puntos

✓ **Cuál es el resultado del siguiente código? siguiendo el ejemplo que venimos viendo**

\*4/4

```
1  Persona p1 = new Persona();
2  String nombre = null;
3
4  try {
5      if (!nombre.equals(null)) {
6          p1.setNombre(nombre);
7      }
8  } catch (Exception e) {
9      System.out.println("El nombre no puede ser nulo");
10 }
```

- ☐ El nombre de p1 queda vacío
- ☐ El nombre de p1 es null
- ☒ Se imprime por consola "El nombre no puede ser nulo"

✓

✓ **Un objeto Exception contiene información sobre un error ocurrido.**

\*2/2

- ☒ Verdadero
- ☐ Falso

✓

✓ **Cuando se lanza una excepción no hay nada más que hacer, termina el programa.**

\*1/1

- ☐ Verdadero
- ☒ Falso

✓



✓ Para aclarar que un método puede lanzar un error se coloca la palabra "extends". \*1/1

☒ Falso



☐ Verdadero

✓ Si una excepción se produce en el bloque catch, puede atraparse en el bloque try y manejarla. \*2/2

☒ Falso



☐ Verdadero

✓ Una excepción es un evento que ocurre durante la ejecución de un programa que interrumpe el flujo normal de las instrucciones del programa. \*1/1

☐ Falso

☒ Verdadero



✓ Sólo se puede usar 1 bloque catch por bloque try. \* 2/2

☐ Verdadero

☒ Falso



## MySQL

8 de 8 puntos

Resultados del ejercicio de los candados





✓ **Candado C \***

1/1

- ☒ Posición 1
- ☐ Posición 2
- ☐ Posición 3
- ☐ Posición 4



✓ **Candado D \***

1/1

- ☐ Posición 1
- ☐ Posición 2
- ☐ Posición 3
- ☒ Posición 4



✓ **Candado A \***

1/1

- ☐ Posición 1
- ☒ Posición 2
- ☐ Posición 3
- ☐ Posición 4



✓ **Candado A \***

1/1

- ☐ Clave: 13539
- ☐ Clave: 15935
- ☐ Clave: 16832
- ☒ Clave: 14043



✓ **Candado C \***

1/1

- ☒ Clave: 631
- ☐ Clave: 963
- ☐ Clave: 145
- ☐ Clave: 601



✓ **Candado B \***

1/1

- ☐ Posición 1
- ☐ Posición 2
- ☒ Posición 3
- ☐ Posición 4



✓ **Candado D \***

1/1

- ☒ Clave: 191
- ☐ Clave: 1063
- ☐ Clave: 153
- ☐ Clave: 101



✓ **Candado B \***

1/1

- ☐ Clave: 4380
- ☒ Clave: 3480
- ☐ Clave: 3830
- ☐ Clave: 4830



## JDBC

20 de 20 puntos

✓ ¿Cuál es la forma más eficiente de realizar múltiples inserciones en una base de datos utilizando JDBC? \*2/2

- ☐ A) Ejecutar una consulta INSERT para cada fila a insertar.
- ☐ D) Utilizar el objeto BatchStatement para ejecutar consultas en lote.
- ☒ C) Utilizar un objeto PreparedStatement y una transacción.
- ☐ B) Utilizar un bucle y ejecutar consultas INSERT individuales.



✓ ¿Cuál es el propósito del objeto ResultSet en JDBC?

2/2

- ☐ D) Define los parámetros de una consulta parametrizada.
- ☐ A) Representa una conexión a una base de datos.
- ☐ C) Ejecuta una consulta SQL.
- ☒ B) Almacena el resultado de una consulta SQL.



✓ ¿Cuál es la forma más segura de ejecutar consultas SQL en JDBC \*2/2 para prevenir ataques de inyección de SQL?

- ☐ D) Utilizar el objeto ResultSet para obtener resultados de consultas.
- ☐ B) Utilizar el objeto Statement para ejecutar consultas sin parámetros.
- ☐ A) Concatenar los parámetros de consulta directamente en la cadena SQL.
- ☒ C) Utilizar el objeto PreparedStatement con parámetros vinculados.



✓ ¿Cuál de las siguientes afirmaciones es cierta acerca de las transacciones en JDBC?

\*2/2

- ☐ D) Las transacciones solo son necesarias cuando se utilizan controladores de base de datos específicos.
- ☐ A) Las transacciones solo se utilizan para consultas SELECT.
- ☐ B) Las transacciones solo se utilizan para consultas de modificación de datos (INSERT, UPDATE, DELETE).
- ☒ C) Las transacciones permiten agrupar múltiples consultas en una única operación atómica.



✓ ¿Cuál de las siguientes declaraciones es cierta acerca de los controladores (drivers) JDBC? \*2/2

- ☐ C) Los controladores JDBC son necesarios solo para la conexión inicial a la base de datos.
- ☒ D) Los controladores JDBC son interfaces utilizadas por la aplicación para interactuar con una base de datos. ✓
- ☐ A) Los controladores JDBC son responsables de la administración de transacciones.
- ☐ B) Los controladores JDBC son proporcionados por el servidor de la base de datos.

✓ ¿Cuál de las siguientes interfaces de JDBC se utiliza para ejecutar consultas parametrizadas? \*2/2

- ☐ C) Connection.
- ☐ B) ResultSet.
- ☐ A) Statement.
- ☒ D) PreparedStatement. ✓

✓ ¿Cuál es el propósito principal de JDBC? \* 2/2

- ☐ C) Mapear objetos a tablas de base de datos.
- ☐ D) Realizar operaciones CRUD en una base de datos.
- ☐ A) Crear consultas SQL.
- ☒ B) Establecer la conexión con una base de datos. ✓



✓ ¿Cuál es la forma correcta de manejar excepciones en JDBC? \* 2/2

- ☐ D) Utilizar el bloque finally para liberar recursos, sin importar si se produce una excepción o no.
- ☐ B) Lanzar una nueva excepción personalizada en cada método de JDBC.
- ☐ A) Ignorar las excepciones y continuar con la ejecución del programa.
- ☒ C) Capturar excepciones específicas de JDBC y manejarlas adecuadamente. ✓

✓ ¿Cuál es la forma correcta de cerrar una conexión JDBC correctamente? \*2/2

- ☐ B) Llamar al método close() en el objeto ResultSet.
- ☐ D) Llamar al método close() en todos los objetos anteriores.
- ☐ A) Llamar al método close() en el objeto Statement.
- ☒ C) Llamar al método close() en el objeto Connection. ✓

✓ ¿Cuál es el propósito del objeto Connection en JDBC? \* 2/2

- ☐ D) Proporcionar métodos para obtener metadatos de la base de datos.
- ☒ C) Representar una conexión física a una base de datos. ✓
- ☐ A) Ejecutar consultas SQL.
- ☐ B) Almacenar los resultados de una consulta.

**JPA**

15 de 15 puntos



✓ ¿Cuál de las siguientes anotaciones se utiliza para marcar una clase como una entidad en JPA? \*2/2

- ☐ D) @OneToMany
- ☐ C) @PersistenceContext
- ☐ B) @Table
- ☒ A) @Entity



✓ ¿Cuál de las siguientes anotaciones se utiliza para marcar una propiedad como una clave primaria en JPA? \*2/2

- ☒ A) @Id
- ☐ D) @Column
- ☐ B) @PrimaryKey
- ☐ C) @GeneratedValue



✓ ¿Cuál de las siguientes anotaciones se utiliza para establecer una relación de uno a muchos en JPA? \*2/2

- ☐ C) @OneToOne
- ☐ B) @ManyToOne
- ☐ D) @ManyToMany
- ☒ A) @OneToMany



✓ ¿Cuál de las siguientes opciones describe mejor el concepto de "carga diferida" (lazy loading) en JPA? \*2/2

- ☒ A) Cargar solo las entidades relacionadas necesarias cuando se accede a ellas. ✓
- ☐ D) Cargar todas las propiedades de una entidad de forma anticipada.
- ☐ B) Cargar todas las entidades relacionadas de forma anticipada.
- ☐ C) Cargar solo las propiedades necesarias de una entidad cuando se accede a ella.

✓ ¿Cuál de las siguientes opciones describe mejor el concepto de "cascada" (cascade) en JPA? \*2/2

- ☐ A) La forma de almacenar entidades en cascada utilizando JPA.
- ☐ D) La forma de cargar entidades relacionadas en cascada utilizando JPA.
- ☒ B) La forma en que se propagan las operaciones en cascada a entidades relacionadas. ✓
- ☐ C) La forma de generar automáticamente identificadores únicos para las entidades.

✓ ¿Cuál de las siguientes opciones describe mejor la diferencia entre CascadeType.PERSIST y CascadeType.MERGE en JPA? \*2/2

- ☒ A) CascadeType.PERSIST realiza una operación de inserción, mientras que CascadeType.MERGE realiza una operación de actualización. ✓
- ☐ B) CascadeType.PERSIST guarda todas las entidades relacionadas, mientras que CascadeType.MERGE guarda solo las entidades modificadas.
- ☐ C) CascadeType.PERSIST realiza una operación de actualización, mientras que CascadeType.MERGE realiza una operación de inserción.
- ☐ D) CascadeType.PERSIST guarda solo las entidades modificadas, mientras que CascadeType.MERGE guarda todas las entidades relacionadas.





- ✓ ¿Cuál de las siguientes afirmaciones describe mejor JPA (Java Persistence API)? \*3/3
- ☐ D) Es un framework para el desarrollo de interfaces de usuario en Java.
  - ☐ B) Es una biblioteca para el acceso a bases de datos NoSQL.
  - ☐ A) Es un lenguaje de consulta para bases de datos relacionales.
  - ☒ C) Es una especificación de Java para el mapeo objeto-relacional. ✓

Este formulario se creó en Egg Cooperation.

Google Formularios

