**Prueba técnica teórica java unificada**

**(Duración 1 hora)**

|  |  |
| --- | --- |
| Nombre candidato | Milmx Jeffrey |
| Teléfono fijo | 3153148046 |
| Celular | 3153148046 |
| Fecha | 2/12/2022 |
| C.C. | 1144204572 |

Las preguntas a continuación hacen parte del proceso de selección para el perfil de ingeniero de desarrollo JAVA.

Responda con base en sus conocimientos y experiencia, ya que de acuerdo con sus respuestas se realizará la categorización dentro del perfil. Las preguntas no contestadas no restan puntos.

**Parte 1 – Conocimientos java (20 puntos)**

* ¿Qué es un “HashMap”? ¿Qué es un “Map” ?

HashMap es una clase en java que implementa la interface Map, para representar colecciones de elementos clave/valor sin indexado. Map facilita los métodos y la estructura a la clase HashMap para representar los elementos, como objetos de una tabla hash.

* ¿Cuál es la diferencia entre excepciones chequeadas y excepciones no chequeadas? Mencione la clase Java que implementa cada una de ellas y un ejemplo de uso

Las excepciones no chequeadas son aquellas que se producen por errores de programación ej. String nombre = 23;

Las excepciones chequeadas son aquellas que son en tiempo de ejecución y pueden ser manejadas ejemplo

try{

String path = reding. get(file\_path)}

catch(IOException e){

System.out.println(e)

}

* Escriba el código que permita realizar el casting del atributo persona de tipo “Usuario” al tipo “Cliente”.

/\* \*\*\*\* Código de clase \*\*\*\*/

//Atributos de clase

@Override

public void Castingtipo(String tipo){

this.personatipo = tipo;

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

//Creando usuario tipo Usuario

Usuario p1 = new Usuario(...);

p1.Castingtipo("Cliente")

* Escriba el código java que permite imprimir por consola el objeto con valor “rojo” sabiendo que se tiene del siguiente Collection de objetos de tipo “Referencia” donde no es posible determinar el orden de sus elementos. El código debe evidenciar como se recorre la colección.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Referencia | | | | Referencia | | | | | Referencia | | | | | Referencia | | | | | Referencia | | | | |  | |
|  | | 5 | verde |  | |  | 8 | rojo |  | |  | 3 | amarillo |  | |  | 4 | gris |  | |  | 1 | blanco |  | |  |

for (Inferencia p: coleccion) {

if(p.getTipo().equals("rojo"){

return p;

}

}

**Parte 2. Conocimientos SQL y Base de datos (20 puntos)**

Para las siguientes preguntas asuma la siguiente tabla de "Ordenes":

|  |  |  |  |
| --- | --- | --- | --- |
| **Id** | **Fecha** | **Precio** | **Cliente** |
| 1 | 2020/11/12 | 1000 | Pedro |
| 2 | 2020/10/23 | 1600 | Jairo |
| 3 | 2020/09/02 | 700 | Pedro |
| 4 | 2020/09/03 | 300 | Pedro |
| 5 | 2020/08/30 | 2000 | Juan |
| 6 | 2020/10/04 | 100 | Jairo |

* Escriba la consulta SQL que cuenta los registros de la tabla “Ordenes”

SELECT COUNT(id)

FROM Ordenes;

* Escriba la consulta SQL que trae las ordenes con precio mayor a “500”, y que estén ordenadas por el campo fecha (de la más reciente hasta la menos reciente)

SELECT \* FROM Ordenes WHERE Precio > 500 ORDER BY Fecha DESC;

* Escriba la consulta SQL que trae las ordenes de los clientes que comienzan por “J”

SELECT \*

FROM Ordenes

WHERE Cliente LIKE J%;

* Escriba la consulta SQL que trae los siguientes resultados

|  |  |
| --- | --- |
| **Cliente** | **Suma de “Precio”** |
| Pedro | 2000 |
| Jairo | 1700 |
| Juan | 2000 |

SELECT Cliente, SUM(Precio) as [Suma de "Precio"] FROM [Ordenes] GROUP BY Cliente;

* Haga un modelo “Entidad-Relación” de las entidades “Persona” (nombres, documento,) , “Producto” (Precio, nombre, marca) , “Orden” (N productos dentro de la orden, valor total, cliente).

**Parte 3. Conocimientos JEE y JPA (20 puntos)**

3.1 ¿Qué es un EJB?. Enumere y explique los tipos de EJBs que existen.

Un EJB es una interfaz que permite desarrollar el paradigma de POO mediante componentes desplegados en contenedores. De esta forma se pueden crear servicios reutilizables para aplicaciones empresariales, entre ellos se encuentran los siguientes cuales pueden sub dividirse, crearse o personalizarse.

beans de sesión

beans de entidad

beans dirigidos

3.2 Con cual anotación JEE / JPA se obtiene una instancia del “EntityManager” para trabajar con la persistencia de objetos en la base de datos?

@Entity

* Explique qué significan las relaciones EAGER y LAZY entre entidades JPA (JEE5). ¿En qué casos se usa cada una de ellas?

Son modos en los que se cargan ciertas funcionalidades en las entidades , dada l la relación que posean entre ellas. Si un atributo posee una relación de uno a uno o de muchos a uno se carga la entidad en modo eager. Pero cuando se tiene relación de muchos a uno o de muchos a muchos se carga en modo lazy. Y todo esto es con el objetivo de optimizar tiempos de consultas basado en el número de queries que se ejecutarán.