Java Software Developer Test

A continuación, se proporciona un conjunto de problemas y preguntas. Por favor, responde a cada uno de ellos y, lo más importante, detalla tu proceso de razonamiento durante la resolución. Para nosotros y para una evaluación adecuada de tu postulación es importante que veamos tu comprensión, tu enfoque para resolver problemas y tus habilidades de redacción.

Algoritmos y Estructuras de Datos

1. **Problema:**

Dada una cadena, determina si es posible reordenar sus caracteres para formar un palíndromo. Si es posible, construye y muestra un palíndromo válido. Si no existe un palíndromo valido, muestra "Not Possible".

Formato de Entrada:

- Una sola línea que contenga una cadena compuesta de caracteres alfabéticos.
- La longitud de la cadena es de máximo 1000 caracteres.
- La cadena puede contener caracteres repetidos.
- Asuma que todos los caracteres son convertidos a minúsculas.

Formato de Salida:

- Muestra una sola línea.
- De lo contrario, imprime "Not Possible".

Ejemplo 1:
```
input = “asdasdasdasdf”
output = “asdasdfdsadsa”
```

Ejemplo2
```
input = ”aepriorpio”
output = ”Not Possible”
```

Ejemplo3
```
input = ”laTinaLavaAnita”
output = ”anitalavalatina”
```

Requisitos:

- Una solución en el lenguaje de su elección (puede ser pseudocodigo), donde cumpla con los objetivos del problema previamente planteado
- En tu solución, detalla las estructuras de datos seleccionadas y explica el motivo de tu elección.
- Detalla cada paso de tu solución en una escritura procedimental.
- Realiza una breve evaluación de la complejidad de tu solución.

POO

2. A continuación, se presenta una sucesión de problemas que crean una representación de un sistema de crawling-indexación. Por favor resuelva cada uno de los problemas empleando POO en Java. Este ejercicio te pide que simules este proceso utilizando únicamente conceptos de programación orientada a objetos, sin necesidad de integrar librerías externas ni lógica real de conexión.

📌 **Contexto general**

Estás modelando una parte del proceso de una empresa que se dedica al crawling, procesamiento e indexación de información proveniente de diversas fuentes (archivos, sitios web, bases de datos, servicios en la nube, etc.) y a la visualización de dicha información mediante buscadores internos o dashboards.

🔍 ¿Qué es crawling?

Es el proceso automatizado de acceder a recursos digitales (como páginas web, documentos en sistemas de archivos, APIs o almacenamiento en la nube) para extraer contenido estructurado o semiestructurado. Empresas como OpenAI, Scale AI, Elastic y otras utilizan crawling para alimentar motores de búsqueda, entrenar modelos de lenguaje o analizar grandes volúmenes de datos.

🔎 ¿Qué es búsqueda de datos?

Es el conjunto de técnicas para permitir a usuarios finales acceder rápidamente a documentos, registros o fragmentos de información relevante a partir de consultas. Esto incluye procesamiento de texto, interfaces de consulta (search UI) y estructuras de salida como dashboards o respuestas JSON.

🧩 **Problema 1: Modelado de documentos y fuentes de datos**

Crea una clase Documento con los atributos:

- String titulo;
- String contenido;
- String fuenteOrigen;

Crea una interfaz:

```java
public interface FuenteDatos {
    List<Documento> obtenerDocumentos();
}
```

Implementa al menos 3 clases que representen distintas fuentes de datos y que implementen la interfaz FuenteDatos. Ejemplos:

- FuenteArchivo
- FuenteWeb
- FuenteExcel
- FuenteSharePoint
- FuenteBaseDatos

Cada clase debe retornar una lista de objetos Documento simulados (pueden tener contenido ficticio) y simplemente imprimir un mensaje como:

```
System.out.println("Obteniendo documentos desde [origen]");
```

En una clase App o Main, instancia varias fuentes de datos, obtén todos los documentos y agrégalos a una lista común.

🧩 **Problema 2: Salida de resultados**

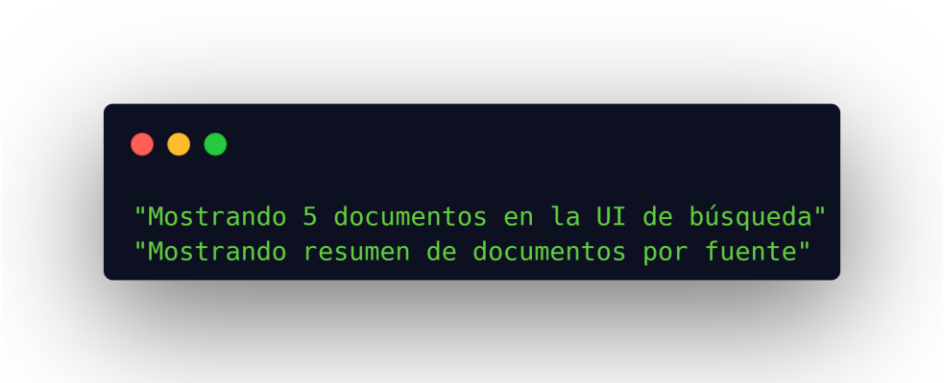
Crea una interfaz:

```
public interface Salida {
    void mostrar(List<Documento> documentos);
}
```

Implementa al menos 2 clases que representen distintas salidas posibles del sistema:

- SalidaSearchUI
- SalidaDashboard

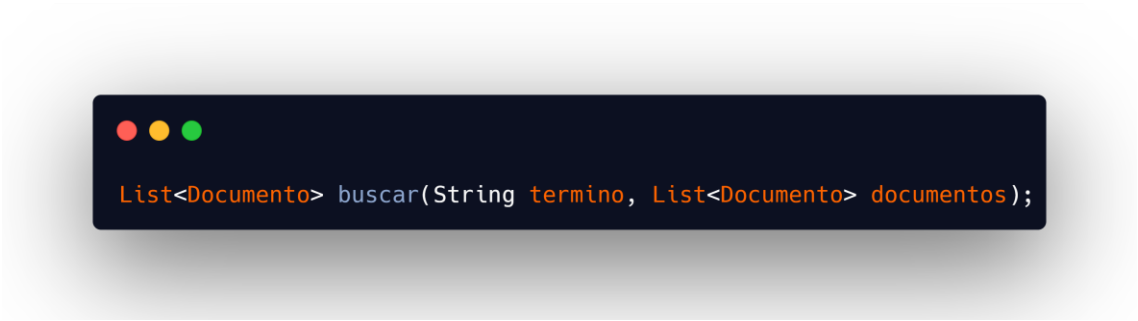
Cada clase debe imprimir un mensaje simulado con System.out.println(), por ejemplo:



Desde el método main, llama a ambas salidas para mostrar los documentos obtenidos.

🧩 Problema 3: Buscador simulado

Implementa una clase Buscador con el siguiente método:



Este método no debe buscar realmente. Solo debe imprimir:

Y retornar una sublista fija (por ejemplo, los primeros 2 documentos).

Requisitos:

- Una explicación general de la solución, donde se indique el porqué de cada clase.
- Diseño orientado a objetos (interfaces, polimorfismo, …)
- Separación de responsabilidad entre clases
- Código bien comentado y clases y data de prueba lista para verificar el funcionamiento

Java Spring Boot 3.

A continuación, se presenta un conjunto de preguntas sobre el lenguaje Java y el framework Spring Boot. Por favor, proporciona una respuesta para cada una en tus propias palabras.

- ¿Qué es Spring Boot?
- ¿Cuál es la diferencia entre Spring y Spring Boot?
- ¿Cuál es el propósito de Maven en Java?
- De un ejemplo de cómo utiliza comúnmente Maven.
- ¿Qué es un repositorio?
- ¿Qué es un controlador?
- ¿Qué es un dto ?
- ¿Por qué es recomendable emplear hibernate?
- ¿Puedo retornar contenido renderizable desde Spring Boot?
- ¿Cómo puedo documentar mis controladores?
- ¿Qué es JPA?
- ¿Qué es la inyección de dependencias?
- De un ejemplo de cómo usar o cuando usar la inyección de dependencias.

React JS

A continuación, se presenta un conjunto de preguntas sobre el lenguaje JS y la librería ReactJS. Por favor, proporciona una respuesta para cada una de estas preguntas, en tus propias palabras.

- ¿Utiliza JS ReactJS?
- ¿Qué es Node JS?
- ¿Cuál es el propósito de npm?
- ¿Cuál es el propósito de vite o webpack?
- Proporcione un ejemplo de cómo emplearía comúnmente npm.
- En el contexto de npm, ¿qué es un paquete?
- ¿Dónde se almacena el árbol de librerías de un proyecto en el contexto de ReactJS?
- ¿Qué es un hook?
- ¿Cuál es el ciclo de vida de un componente en React?
- ¿Puedo emplear el paradigma de POO para desarrollar en ReactJS?
- ¿Cuándo se usa el hook useMemo?

Interacción Navegador Servidor

3. Para las siguientes preguntas sobre la interacción entre el navegador web y el servidor, por favor proporciona una respuesta en tus propias palabras.
   - Cuando un protocolo de comunicación como HTTP tiene el sufijo "s", ¿qué significa? (Por ejemplo, http://mydomain.com/path se convierte en https://mydomain.com/path).
   - Cuando las aplicaciones involucran tanto el navegador web como el servidor, ¿la ejecución del código ocurre únicamente en el servidor?
   - Cuando cierras una pestaña en tu navegador, ¿es posible almacenar información que persista hasta que regreses a esa página? En caso afirmativo, ¿cómo?

- Al interactuar entre el navegador y el servidor, comúnmente utilizamos solicitudes HTTP con métodos específicos (GET, POST, PUT, DELETE, etc.). ¿Se utilizan estos métodos al transmitir contenido como imágenes, sonido, video o archivos? En caso afirmativo, ¿cuál es el método más utilizado?
- ¿Qué lenguajes se utilizan comúnmente en el navegador?
- ¿Se pueden utilizar los lenguajes comúnmente usados en el navegador en el lado del servidor?
- ¿Se pueden utilizar lenguajes del lado del servidor en el navegador?

Docker

4. Para las siguientes preguntas sobre Docker y contenedores, por favor proporciona una respuesta en tus propias palabras.
    - ¿Qué es docker?
    - Seleccione la opción que no corresponde con un comando de Docker válido.
        - docker run
        - docker shutdown
        - docker stop
        - docker images
    - ¿Dos contenedores pueden compartir las mismas librerías?
    - ¿Qué diferencia hay entre una máquina virtual y un contenedor?
    - ¿Pueden dos contenedores desplegar un servicio en el mismo puerto interno?

Buena Suerte!!!!