**PRUEBA TÉCNICA DESARROLLADORES**

Resuelva esta prueba con sus propios conocimientos, sin buscar respuesta en internet, no esperamos que todas las respuestas sean correctas, esto nos permite identificar mejor sus capacidades y plantear un plan de carrera

1. Explique en sus propias palabras y de un ejemplo de los siguientes conceptos / principios de software

|  |  |  |
| --- | --- | --- |
|  | Explicación | Ejemplo |
| Bajo acoplamiento / Alta cohesión | Bajo acoplamiento equivale a que los componente o módulos de un sistemas están lo menos interconectados posible. Alta Cohesión es algo diferente y es a nivel modular, ya que la mayoría o todas las partes internas de un modulo deben estar muy relacionadas y trabajar en conjunto. | Bajo acoplamiento: Puede ser el sistema creado para una tienda de algún producto, tendría módulos de ventas, inventario, contabilidad, etc, esos módulos al estar débilmente acoplados, los cambios en uno no afectarían en gran medida o en ninguna medida afectaría a los demás. Alta cohesión: Un módulo de gestión de usuarios tendría métodos para crear, actualizar, eliminar, entre otros y estarían centrado en la gestión de usuarios, no en los demás servicios que presta la aplicación. |
| Thread safety | Equivaldría a que una aplicación puede manejar múltiples hilos de manera simultánea sin presentar errores. | Un ejemplo puede ser las reservas de turno para recepción de medicamentos u otro elemento, si se hace online el sistema deber al poder gestionar las peticiones que se realicen por varios clientes en el mismo instante sin darles el mismo turno a ambos o bloquearse, para ello lo ideal es esencial controlar que las modificaciones se puedan realizar por un hilo a la vez. |
| Acoplamiento | Tiene que ver con el grado de dependencia entre módulos de un sistema. | Acomplamiento fuerte: Un módulo de ventas que accede directamente a la base de datos de Clientes para obtener información, por ello cualquier cambio en la tabla clientes puede presentar riesgo de problemas en el módulo clientes. |
| Polimorfismo | Es cuando un objeto puede tomar diferentes comportamientos o características según el contexto dónde se use. | Al tener una clase base llamada Forma, la cual tiene métodos como Dibujar puede tener clases derivadas como Circulo, Triangulo, Cuadrado, etc. Precisamente esto se aplica en el patrón de diseño llamado Clase Abstracta. |

1. Escriba un fragmento de código que solucione el siguiente problema, puede utilizar cualquier lenguaje de programación

Problema:

rotar a la derecha m veces los elementos de un arreglo, donde m >= 0 y el arreglo tiene tamaño n, donde n >= 0. Note que no se pierden elementos en el arreglo.

|  |  |  |
| --- | --- | --- |
| **Ejemplo** | **Valor de m** | **Resultado** |
| arr[1,2,3,4] | m = 1 | arr[4,1,2,3] |
| arr[1,2,3,4] | m = 2 | arr[3,4,1,2] |
| arr[1,2,3,4] | m = 3 | arr[2,3,4,1] |
| arr[1,2,3,4] | m = 4 | arr[1,2,3,4] |

Solución:

|  |
| --- |
| function rotarDerecha(arr, m) {if (m >= arr.length) {return arr;}let indiceInicio = arr.length - m;let parteRotada = arr.slice(indiceInicio);let parteNoRotada = arr.slice(0, indiceInicio);let arrRotado = parteRotada.concat(parteNoRotada);return arrRotado;} //Realizado con Javascript para probarlo más rápidamente en vscode.  Pruebas: console.log(rotarDerecha([1, 2, 3, 4], 1)); // [4, 1, 2, 3]  console.log(rotarDerecha([1, 2, 3, 4], 2)); // [3, 4, 1, 2]  console.log(rotarDerecha([1, 2, 3, 4], 3)); // [2, 3, 4, 1]  console.log(rotarDerecha([1, 2, 3, 4], 4)); // [1, 2, 3, 4] |

# Caso Bluesoft Bank

Bluesoft Bank es un banco tradicional que se encarga de guardar el dinero de sus ahorradores, ofrece dos tipos de cuenta; ahorros para personas naturales y corrientes para empresas. Adicionalmente para cada cuenta se pueden hacer consignaciones y retiros.  
  
Adicionalmente tiene que soportar algunos requerimientos para sus ahorradores:

* Consultar el saldo de la cuenta
* Consultar los movimientos más recientes
* Generar extractos mensuales

Reglas de negocio:

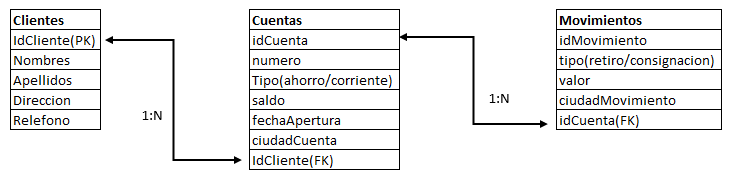
* Una cuenta no puede tener un saldo negativo.
* El saldo de la cuenta siempre debe ser consistente frente a dos operaciones concurrentes (consignación, retiro)

También se deben generar reportes en tiempo real como:

* Listado de clientes con el número de transacciones para un mes es particular, organizado descendentemente (primero el cliente con mayor # de transacciones en el mes)
* Clientes que retiran dinero fuera de la ciudad de origen de la cuenta con el valor total de los retiros realizados superior a $1.000.000.

**En base a lo anterior, por favor dar respuesta a los siguientes puntos:**

1. Cree un diagrama de clases que modelo el problema, identifica los elementos principales y sus relaciones.



El diagrama anterior indica las siguientes relaciones:

Una relación 1:N entre Clientes y Cuentas, lo que significa que un cliente puede tener varias cuentas.

Una relación 1:N entre Cuentas y Movimientos, lo que indica que una cuenta puede tener varios movimientos.

1. Qué arquitectura y tecnologías usaría para resolver el caso Bluesoft Bank

La arquitectura que usaría serpia de microservicios, la cuál presenta muchas ventajas debido a su capacidad para mejorar la escalabilidad, la flexibilidad y el mantenimiento de aplicaciones de software.  
La base de datos puede ser SQL en AZURE, en este caso y para mayor velocidad de desarrollo usé una arquitectura monolítica y base de datos local.

1. Subir la implementación del caso a un repositorio público y compartir enlace