**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 | Low coupling refers to the degree of interdependence between modules.  Modules are low couped implies changes for a module are less likely to change other modules.  High cohesion refers relation between modules contributing to a single purpose. | -user authentication module (login,register and authentication).  -Account management module (update and deleting user accounts).  -transaction module (deposit,withdrawals and transfers)  -report module (account statement, transaction histories)  Each module is independent interact with other with specific interfaces so there is low coupling also each module encapsule functionality for a objective |
| Thread safety | Refers to a capacity of a program to function reliably when multiple threads are executing concurrently | Any program with semaphore system |
| Acoplamiento | High coupling refers a system when changing a module implies high change in other modules | User Interface Module (UI): This module is responsible for displaying the user interface elements such as message threads, contacts, and input fields.  Messaging Module: This module handles sending and receiving messages, as well as storing them in the database.  Notification Module: This module is responsible for sending notifications to users when they receive new messages. |
| Polimorfismo | Is the property when can use the same method in objects of distinct classes. The result of these call method can be different. | class Lion(Animal):  def roar(self):  print("The lion roars: Roar!")  class Wolf(Animal):  def roar(self):  print("The wolf roars: Howl!")  lion = Lion()  wolf = Wolf()  lion.roar()  wolf.roar() |

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:

def rotate(array,m):

n=len(array)

m%=n

return array[(-m):]+array[:(-m)]

# 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.
2. Qué arquitectura y tecnologías usaría para resolver el caso Bluesoft Bank
3. Subir la implementación del caso a un repositorio público y compartir enlace