**Repaso Evaluación 2**

Dentro de la carpeta Evidencias crear una subcarpeta con su nombre y apellido y subir la evidencia

Para la evaluación es necesario aplicar todo lo visto durante la capacitación, recuerde respetar estándares de nombrado y utilizar los conceptos revisados.

1. Importar en el IDE el proyecto JSEBEvaluación.
2. Agregar lo necesario para que la clase Cliente contenga y se cumpla el principio de Encapsulamiento:

|  |  |
| --- | --- |
| **Atributos** | **Cedula**: Número de identificación del cliente.  **Nombres**: Nombre completo del cliente.  **Cuenta**: Es la cuenta que tiene el cliente, va a tener las siguientes características: saldo, tipo de cuenta, código. |
| **Constructores** | Crear un constructor que reciba como parámetros cédula, nombres y asignarlos a los atributos correspondientes. |
| **Métodos** | Método **imprimir**: No recibe parámetros y no tiene retorno. Este método debe mostrar la información del cliente en consola de la siguiente manera:  \*\*\*\*\*\*\*\*\*\*\*\*\*\*CLIENTE \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  Nombre: XXXXXX  Cedula: XXXXXX  ---------------CUENTA ------------------------  Codigo: xxxxx  Tipo: Nombre  Saldo Actual: xxxxx  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* |

1. Agregar lo necesario para que la clase Cuenta contenga y se cumpla el principio de encapsulamiento:

|  |  |
| --- | --- |
| Atributos | **Código**: Es un conjunto de caracteres entre letras y números. Por ejemplo, A123C.  **Tipo**: Es el tipo de cuenta, tiene características como código y nombre.  **Saldo**: Es el saldo actual que posee la cuenta. |
| Constructores | Crear un constructor que reciba como parámetros el código y el tipo. Deberá asignar los valores de los parámetros a los atributos correspondientes.  Definir el constructor vacío. |
| Métodos | Método **validarSaldo** recibe como parámetro el saldo que se desea asignar a la cuenta, retorna true o false. Retorna true en caso de que el parámetro saldo que reciba sea positivo, caso contrario retorna false. |

1. Agregar lo necesario para que la clase TipoCuenta contenga:

|  |  |
| --- | --- |
| Atributos | **Código**: Número entero que permite identificar el tipo de cuenta.  **Nombre**: Nombre del tipo de cuenta. |
| Constructores | Crear un constructor que reciba como parámetros el código y nombre. Deberá asignar los valores de los parámetros a los atributos correspondientes. |
| Métodos | Método **obtenerValorRecargo** no recibe parámetros y retorna el valor de recargo que es entero. El valor recargo se calcula en base a la siguiente lógica:   * Si el código es igual o menor 10 se devolverá el valor de recargo 20. * Si el código es mayor a 10 y menor a 20 el valor de recargo es 30. * Si el código es mayoro igual a 20 el valor de recargo es 40. * En caso de que no se cumpla con las condiciones antes dichas el valor de recargo es 0 |

1. Agregar lo necesario en la clase AdminPagos:

|  |  |
| --- | --- |
| **Constructores** | Mantener el constructor por defecto |
| **Métodos** | * **Asignar Cuenta**: Recibe un parámetro de tipo Cuenta y un parámetro de tipo Cliente. No tiene retorno y lo que hace es asignar la Cuenta al Cliente. * **Depositar**: Recibe un parámetro saldo de tipo double y uno de tipo Cliente. Valida que el valor saldo sea correcto, para esto invoca al método **validarSaldo** de la Cuenta. Si el retorno del método validarSaldo es **true** invoca al método **obtenerValorRecargo** de la clase TipoCuenta y asigna al atributo **saldo** de la cuenta el saldo que recibe como parámetro multiplicado por el valor de recargo. En caso de que el retorno del método validarSaldo sea **false** se muestra un mensaje en consola que diga “No se puede depositar”. No tiene retorno. * **Retirar**: Recibe un parámetro saldo de tipo double y uno de tipo Cliente. Valida que el valor saldo sea correcto, para esto invoca almétodo **validarSaldo** de la Cuenta. Si el retorno del método **validarSaldo** es **true** se resta del saldo de la Cuenta el valor que se recibe como parámetro. En caso de que el retorno del método **validarSaldo** sea **false** se muestra un mensaje en consola que diga “No se puede retirar” * **ObtenerTipoCuenta**: Recibe como parámetro un Cliente y retorna el TipoCuenta que posee ese cliente. El método lo que hace es retornar el tipoCuenta que se encuentra en la Cuenta asociada al Cliente. |
| **Librerías** | Importar la librería **Prestamo**. Crear un método que se llame **mostrarPrestamo**, recibe un parámetro que es Cliente y retorna un String con el tipo de préstamo que puede solicitar el cliente. Para esto el método crea una instancia de Prestamo y hace lo siguiente:   * Si el retorno del método **obtenerPrestamo** es menor a 500 el String que se retorna debe decir “Préstamo A” * Si el retorno del método **obtenerPrestamo** es mayor o igual a 500 y menor o igual a 1000 el String que retorna debe decir “Préstamo B” * Si el retorno del método **obtenerPrestamo** es mayor a 1000 el String que retorna debe decir “Préstamo C” * Si el retorno del método **obtenerPrestamo** es cualquier otro valor que no se considera en las anteriores validaciones el String que retorna debe decir “No puede obtener préstamo”.   El método **obtenerPrestamo** recibe como parámetro un doble que es el saldo de la cuenta y retorna un double con el valor del préstamo que se puede entregar. |

1. En la clase **TestCuentas** ubicada en el paquete **tests**. En caso de que existan errores por favor corregir lo que sea necesario **sin modificar nada de la clase TestCuentas**.