UNIVERSIDAD DE LOS LAGOS

Laboratorio RAI

Programación Orientada Objetos 2023-1

Integrantes:

El banco ULagosBank es una institución anexa a la Universidad de Los Lagos que permite a los estudiantes acceder a servicios bancarios acptados en el mercado, tales como tarjetas de crédito y préstamos.

Es por ello, que se le ha solicitado a ustedes, como estudiantes ICINF, que puedan desarrollar una solución factible a través de Programación Orientada a Objetos, o siguiente:

- (10 pts) Diseño de Diagrama de Clases de la solución.
- (20 pts) Crear una clase llamada **Persona** con atributos: **nombre**, **fecha de nacimiento** y **RUT**. Además considere los siguientes métodos:
  - Un constructor, donde los datos pueden ser inicializados con valores vacíos.
  - Setters y Getters para cada uno de los atributos. Se deben validar todas las entradas de datos. En el caso del rut, debe validar la forma y el dígito verificador con el algoritmo del rut[1]
  - Definir el método **toString()** que organice la información de la persona.
  - Implementar el método **esMayorDeEdad()**, que permite verificar si la persona tiene una edad igual o mayor a 18 años.
- (5 pts) Crear una clase llamada **Cliente**, que tiene el comportamiento de la clase persona, y posee los atributos de **password**, que es un texto de 10 caracteres, y otro llamado **Productos**, que son todas los tipos de productos bancarios que posee el banco ULagosBank.
- (5 pts) Crear una clase llamada **Ejecutivo**, que tiene el comportamiento de la persona, y posee los atributos de **Usuario**, que es un texto de 10 caracteres que identifica al Ejecutivo, y otro llamado **password**, que es un texto de 10 caracteres.
- (20 pts) Crear una clase llamada **Cuenta**, que tiene los atributos: **titular** (que es un Cliente), **saldo** (puede tener decimales) y **movimientos** (pueden ser decimales). El titular es necesario

[1] Recursos relacionados en: `https://es.wikipedia.org/wiki/Rol_%C3%9Anico_Tributario#:~:text=vuelto%20a%20asignar.-,Algoritmo,enga%C3%B1os%20y%20suplantaciones%20de%20identidad.` y `https://es.wikipedia.org/wiki/Anexo:Implementaciones_para_algoritmo_de_rut`

para crear una cuenta, el saldo se refiere al monto que guardará en la cuenta, pudiendo ser vacío en primera instancia, y los movimientos son los registros de todos los depósitos y retiros de dinero. Además, se requieren los siguientes métodos:

- Un constructor que permita crear objetos vacíos.
- Los Setter y Getters para cada uno de los atributos. El atributo **cantidad** no puede ser modificado de forma normal, sino que se debe establecer los métodos de **deposito()** y **retiro()**, con las comprobaciones de dinero; es decir, solamente se pueden hacer depósitos con montos mayores a 0 y no puede retirar montos de dinero mayores a la cantidad que posee en la cuenta.
- Definir el método **toString()** que organice la información de la **Cuenta**.
- Definir el método **registrarMovimiento()** que permite tener registro de cada transacción de dinero en la cuenta.

- (20 pts) Crear una clase llamada **CuentaJoven**, que es una especificación de la clase **Cuenta** destinada para personas jóvenes hasta los 25 años y que deseen ahorrar dinero. Esta clase posee los atributos de la clase anterior y, además. posee un atributo **bonificacion** que almacena un porcentaje de ganancia al ahorrar dinero dentro del banco. Además, se requieren los siguientes métodos:

  - un constructor, que siempre debe solicitar todos los datos de la cuenta.
  - Los Setters y getters para el atributo **Bonificacion**.
  - Definir un método **esTitularValido()** para validar que los titulares de este tipo de cuenta pertenezcan al grupo etario de 18 a 25 años. Este método devolverá un valor booleano y establecerá si se puede crear o no una cuenta de este tipo.
  - Definir un método llamado **bonificar()** que se realiza al realizar un depósito, se suma el monto de dinero del depósito y se le agrega la bonificación dada por el porcentaje de la bonificación (Se deben sobrescribir algunos métodos de la superclase).
  - Defnir el método **toString()** que organice la información de la **CuentaJoven**.

- (20 pts) Desarrollar una clase **Main**, que permita mostrar todas las opciones del programa por mensajes por consola (CLI).

  - Establecer un manú de opciones diferenciado para Clientes y Ejecutivos, permitiendo ingresar solamente a las opciones que puede realizar cada uno.
  - Las opciones del ejecutivo son: Crear cliente y Ejecutivos, Crear cuenta y/o Cuenta Joven.
  - Las opciones del Cliente son: Seleccionar producto bancario, Realizar Depósito y Realizar Retiro()

Observación: Puede utilizar todas las técnicas que requiera, clases, atributos y métodos auxiliares.

Evaluación:

- El desarrollo de este proyecto es en **equipos de 2 integrantes**, indique claramente en el código quienes son los integrantes. Solamente un integrante del equipo debe entregar el avance correspondiente en la plataforma ULagos Virtual.
- La copia será sancionada para todos los equipos involucrados.
- Se debe entregar:
  - Informe con Diagrama de Clases y su descripción (justificación) de la solución. (10 pts)
  - Desarrollo del Código en Repositorio Git/GitHub. (90 pts)
- Plazo de entrega: Lunes 13 de Noviembre hasta las 16:00hrs.