Programación Orientada a Objetos

Práctica 05.

Objetivos:

- Practicar los conceptos de Herencia y Polimorfismo.
- Proseguir con el desarrollo del sistema bancario.

Antes de iniciar los ejercicios, baje de la página del curso, sección Material, el archivo Practica05.zip al escritorio y descomprímalo para obtener los archivos necesarios para esta práctica.

Ejercicio 1. Crear subclases de cuentas en el Sistema Bancario.

1) Abra el proyecto de netbeans BankLtd que se ha venido desarrollando hasta ahora.
2) Borre la clase TestBanking de la versión anterior del proyecto.
3) Cambie el modificador de acceso del atributo balance de la clase Account a **protected**.
4) Consulte el diagrama de clases adjunto e implemente la clase SavingsAccount como sigue:
   a) debe ser subclase de Account.
   b) debe incluir un atributo privado interestRate de tipo double.
   c) debe incluir un constructor que tome dos parámetros: balance e interestRate; debe inicializar el atributo interestRate y pasar el argumento balance a la superclase.
5) Consulte el diagrama de clases adjunto e implemente la clase CheckingAccount como sigue:
   a) debe ser subclase de Account.
   b) debe incluir el atributo overdraftProtection de tipo double.
   c) debe incluir un constructor que tome un parámetro: balance, el cual debe ser pasado a la superclase.
   d) debe incluir otro constructor que tome dos parámetros: balance y protect; balance se debe pasar a la superclase y protect debe inicializar el atributo overdraftProtection.

e) debe substituir (override) el método withdraw de la superclase, de tal manera que verifique si el saldo actual de la cuenta es adecuado para cubrir el monto del retiro. Si no lo es, pero hay suficiente dinero en el atributo overDraftProtection, debe aceptar la transacción, afectando tanto el saldo de la cuenta como el overDraftProtection. Si aun tomando en cuenta el overDraftProtection no se puede cubrir el monto del retiro, el método deberá regresar false, dejando sin afectar los atributos.

6) Copie la nueva versión del archivo TestBanking.java proporcionado en el directorio Practica05/ejercicio1 al proyecto BankLtd, paquete banking.

7) Estudie el nuevo programa TestBanking, ejecútelo y observe el resultado.

Ejercicio 2. Crear una colección heterogénea para representar las cuentas de los clientes en el Sistema Bancario.

1) Modifique la clase Customer para manejar la asociación de cuentas a clientes, de la misma manera que se hizo en el ejercicio 2 de la práctica 04 en la clase Bank. Consulte el diagrama de clases para agregar los siguientes elementos:

   a) atributo accounts (protected) que es un arreglo de objetos de clase Account que debe ser instanciado en el constructor de la clase.

   b) atributo entero numOfAccounts.

   c) método público addAccount(Account account), que construya un nuevo objeto account y lo añada al final del arreglo accounts. También debe incrementar en 1 el atributo numOfAccounts.

   d) método público getNumberOfAccounts() que entregue el valor del atributo numOfAccounts.

   e) método público getAccount(int n) que entregue la cuenta n del cliente.

   Note que ya existía el método getAccount() pero ahora tiene un argumento y note también que no se requiere más el método setAccount().

2) Borre el programa TestBanking que se utilizó en el ejercicio 1.

3) Copie la nueva versión del archivo TestBanking.java proporcionado en el directorio Practica05/ejercicio2 al proyecto BankLtd, paquete banking.

4) En el nuevo programa TestBanking proporcionado hay comentarios que empiezan y terminan con /*** … ***/. Estos comentarios indican los lugares del programa donde se deben agregar postulados.

5) En TestBanking use el operador instanceof para probar que tipo de cuenta tiene el cliente y poner en la variable accountType el valor apropiado como “Savings Account” o “Checking Account”.

6) Imprima el tipo de cuenta y el saldo. Utilice métodos de la clase NumberFormat para formatear el campo del saldo con signo de pesos. punto decimal y centavos.

7) Ejecute TestBanking y observe el resultado cuidadosamente.

Diagrama de Clases.