Departamento de Programación
Facultad de Informática
Universidad Nacional del Comahue

Introducción a la Programación Orientada a Objetos

Trabajo Práctico 3 - Herencia

1. Defina e implemente una clase ***Cliente*** que hereda de la clase ***Persona*** (DNI, Nombre y Apellido) con la información básica de un cliente (Nro. de Cliente, DNI, Nombre y Apellido).

2. Defina e implemente una clase "***Cuenta***" que contenga la información de una cuenta de un banco y haga referencia a su dueño. Tener en cuenta que las cuentas pueden ser de 2 tipos: Cuenta Corriente o Caja de Ahorro.
   Nota: La Cuenta Corriente se distingue de una Caja de Ahorro porque su dueño puede girar en descubierto. Es por ello que la clase Cuenta Corriente debe tener un atributo que determine el monto máximo para girar en descubierto.
   Implementar los siguientes métodos en la clase:

   1. ***saldoCuenta****()* : retorna el saldo de la cuenta.

   2. ***realizarDeposito****(monto)*: permite realizar un depósito a la cuenta una cantidad "*monto*" de dinero.

   3. ***realizarRetiro****(monto):* permite realizar un retiro de la cuenta por una cantidad "*monto*" de dinero.

3. Defina una clase ***Banco*** con las siguiente variables instancias:

   1. ***coleccionCuentaCorriente***: variable que contiene una colección de instancias de la clase Cuentas Corrientes.

   2. ***coleccionCajaAhorro***: variable que contiene una colección de instancias de la clase Caja de Ahorro .

   3. ***ultimoValorCuentaAsignado***: variable que contiene el ultimo valor asignado a una cuenta del banco.

   4. ***coleccionCliente***: variable que contiene una colección de instancias de la clase Cliente

4. En la clase Banco defina e implemente los siguientes métodos:

   1. ***incorporarCliente****(objCliente)*: permite agregar un nuevo cliente al Banco

   2. ***incorporarCuentaCorriente****(numeroCliente)*: Agregar una nueva Cuenta a la colección de cuentas, verificando que el cliente dueño de la cuenta es cliente del Banco.

   3. ***incorporarCajaAhorro****(numeroCliente)*:Agregar una nueva Caja de Ahorro a la colección de cajas de ahorro, verificando que el cliente dueño de la cuenta es cliente del Banco.

   4. ***realizarDeposito****(numCuenta,monto)*: Dado un número de Cuenta y un monto, realizar depósito.

   5. ***realizarRetiro****(numCuenta,monto)*: Dado un número de Cuenta y un monto, realizar retiro.

5. Implemente una clase ***TestBanco*** que realice las siguientes operaciones:

   1. Crear un objeto de la clase Banco.

   2. Crear dos nuevos clientes Cliente1 y Cliente2, y agregarlos al banco.

   3. Crear dos Cuentas Corrientes, una asociada al cliente A y otra al cliente B, y agregarlas al Banco .

4. Crear tres Cajas de Ahorro, dos asociadas al cliente A y una asociada al cliente B, y agregarlas al Banco .

5. Depositar $300 en cada una de las Cajas de Ahorro.

6. Transferir $150 de la Cuenta Corriente de Cliente1, a la Caja de Ahorro de Cliente2.

7. Mostrar los datos de todas las cuentas.

6. Se desea sistematizar la venta de un local de productos regionales e importados, la primer etapa de la sistematización involucra la obtención del precio de venta de un producto a partir de su código de barra y el costo de todos los productos disponibles para la venta dentro del local.

   De los productos se almacena el código de barra, la descripción, el stock, el porcentaje de iva, el precio de compra y una referencia al rubro que pertenece. De cada rubro se almacena la descripción y el porcentaje de ganancia aplicado a los productos vinculados a ese rubro.

   El precio de venta de un producto se calcula sobre el precio de compra, más el porcentaje de ganancia en base a su rubro, más el porcentaje de IVA que se aplica al producto.

   Los productos del local pueden ser regionales o importados. Si son importados, el precio de venta se incrementa un 50 % y se agrega un impuesto del 10 % sobre el valor obtenido. Si son regionales se almacena un porcentaje de descuento que será aplicado al precio de venta. Implementar un método *darPrecioVenta()* que retorna el precio de venta de un producto y redefinirlo cuando sea necesario.

   De las Ventas se almacena la fecha, la referencia al producto o los productos y el cliente al que se le ha realizado la venta. El importe final de una venta normal se calcula en base a la cantidad de productos, por el importe del producto.

   El local almacena los productos que tiene disponible para la venta e implementa los siguientes métodos:

   - *incorporarProductoTienda(objProducto)*: método que recibe por parámetro un objeto Producto y verifica que el código de barra no se encuentre dentro de la lista. Si el producto ya existe no es incorporado dentro de la lista de productos de la tienda. El método retorna verdadero o falso según corresponda.

   - *retornarImporteProducto(codProducto)*: método que recibe por parámetro el código de un producto y retorna el precio de venta.

   - *retornarCostoProductoTienda()*: recorre todos los productos de la tienda y retorna la suma de los costos de cada producto teniendo en cuenta el stock de cada uno.

   Definir la jerarquía de clases junto a los métodos y variables instancias de cada una de ellas. No olvidar realizar el diagrama de clases.

   Implementar cada una de las clases definidas con sus métodos de acceso para cada variable instancia y los métodos constructores según la definición de cada clase, y los métodos *__toString*.

   Implementar el método *darImporteVenta()* que retorna el valor que debe ser abonado por el cliente.

   Implementar los métodos mencionados en el enunciado en la clase que corresponda y redefinirlos de ser necesario:*darPrecioVenta()*, *incorporarProductoTienda(objProducto)*, *retornarImporteProducto(codProducto)*, *retornarCostoProductoTienda()*.

   Implementar la clase Local que contiene una colección de productos importados y una colección de productos regionales, la colección de ventas realizadas por la agencia. La clase cuenta con los siguientes métodos:

   - ***productoMasEcomomico****(rubro)*: método que retorna el producto más económico de un rubro.
   - ***informarProductosMasVendidos****(anio, n)*: retorna los n productos más vendidos en el año recibido por parámetro.

- ***promedioVentasImportados()***: método que retorna el promedio de ventas de productos importados realizadas.
- ***informarConsumoCliente****(tipoDoc,numDoc)*: método que retorna todos los productos que fueron comprados por la persona identificada con el *tipoDoc y numDoc* recibidos por parámetro.

Implementar un script *Test_Tienda* donde:

- Se crean 2 rubros: conservas con un 35 % de ganancia, regalos con un 55 % de ganancia.
- Se crean 4 instancias de la clase Producto y se vinculan a los rubros creados en el inciso anterior.
- Se incorpora cada instancia de la clase Producto a la Tienda.
- Se incorpora nuevamente la última instancia de producto incorporada a la tienda y visualizar el resultado.
- Se retorna el precio de venta de cada uno de los productos creados.
- Se retorna el costo en productos que tiene hasta el momento la tienda.

7. Una agencia de viaje desea sistematizar la venta de paquetes turísticos. Para ello desea almacenar información de los destinos, paquetes turísticos y de las ventas realizadas. Las ventas de los paquetes puede ser a partir de la agencia o por web.

   De los ***Destinos*** se almacena una identificación, el nombre del lugar y el valor por día en ese destino por pasajero.

   De los ***Paquetes Turísticos*** se almacena fecha desde, cantidad de días, destino, cantidad total de plazas y cantidad disponible de plazas. El constructor de la clase paquete turístico no recibe como parámetro la cantidad de plazas disponibles, debe ser un valor que se setea con el valor recibido para la cantidad total de plazas del paquete.

   De las ***Ventas*** se almacena la fecha, la referencia al paquete, la cantidad de personas, y el cliente al que se le ha realizado la venta. Por otro lado hay ventas **On-Line**, de estas ventas se almacena el porcentaje de descuento y tienen un cálculo de venta diferente. El importe final de una venta normal se calcula en base a la cantidad de días, por el importe del día del paquete, por la cantidad de personas de la venta. Si se trata de una venta On-Line al importe de la venta se aplica el porcentaje de descuento cuyo valor por defecto es de un 20% (este valor puede variar). Definir la jerarquía de clases junto a los métodos y variables instancias de cada una de ellas. No olvidar realizar el diagrama de clases.

   7.1. Implementar cada una de las clases definidas con sus métodos de acceso para cada variable instancia y los métodos constructores según la definición de cada clase, y los métodos *toString*.

   7.2. Implementar el método **darImporteVenta***()* que retorna el valor que debe ser abonado por el cliente y redefinirlo según sea necesario.

   7.3. Implementar la clase *Agencia* que contiene una colección de paquetes turísticos, la colección de ventas realizadas por la agencia y la colección de ventas On-Line. La clase cuenta con los siguientes métodos:

   a) **incorporarPaquete***(objPaqueteTuristico) :*que incorpora a la colección de paquetes turísticos un nuevo paquete a la agencia siempre y cuando no haya un paquete en la misma fecha al mismo destino. Si el paquete pudo ser ingresado el método debe retornar true y false en caso contrario.

   b) ***incorporarVenta****(objPaquete,tipoDoc,numDoc,cantPer, esOnLine):* método que recibe por parámetro el paquete, tipo documento, número de documento, la cantidad de personas que van a realizar el paquete turístico y si se trata o no de una venta on-line (valor true o false). El método retorna el importe final que debe ser abanado en caso que la venta pudo concretarse con éxito y -1 en caso contrario.

   c) ***informarPaquetesTuristicos****(fecha,destino)*: método que retorna una colección con todos los paquetes que se realizan en una fecha y a un destino.

   d) ***paqueteMasEcomomico****(fecha,destino)*: método que retorna el paquete turístico mas económico para una fecha y un destino.

   e) ***informarConsumoCliente****(tipoDoc,numDoc)*: método que retorna todos los paquetes que fueron utilizados por la persona identificada con el *tipoDoc y numDoc* recibidos por parámetro.

   f) ***informarPaquetesMasVendido****(anio, n)*: retorna los n paquetes turísticos mas vendido en el año recibido por parámetro.

g) ***promedioVentasOnLine()***: método que retorna el promedio de ventas on-line realizadas por la agencia.

h) ***promedioVentasAgencia()***: método que retorna el promedio de ventas on-line realizadas por la agencia.

7.4. Implementar y redefinir el método ***darImporteVenta***, según corresponda.

7.5. Implementar un script ***Test_Agencia*** donde:

a) se crea una instancia de la clase Destino con los siguientes valores: lugar =“Bariloche”, valorDia =250.

b) se crea una instancia de la clase ***PaqueteTuristico*** referenciando al destino creado en (a) con los siguientes valores: fdesde = 23/05/2014 cantDias= 3, totalPlazas = 25.

c) se crea instancia de la clase Agencia.

d) se invoca al método ***incorporarPaquete*** de la agencia con el paquete creado en (b).

e) se invoca nuevamente al método ***incorporarPaquete*** de la agencia con el paquete creado en (b).

f) se invoca al método ***incorporarVenta*** con los siguientes parámetros: el paquete creado en (b), con los siguientes datos del cliente: número de documento 27898654, y tipo de documento DNI, cantidad de personas que viaja: 5, y no es una venta online.

g) se invoca al método ***incorporarVenta*** con los siguientes parámetros: el paquete creado en (b), con un cliente con número de documento 27898654, tipo de documento DNI, cantidad de personas que viaja: 4, y es una venta online.

h) se invoca al método ***incorporarVenta*** con los siguientes parámetros: el paquete creado en (b), con un cliente con número de documento 27898654, tipo de documento DNI, cantidad de personas que viaja: 15, y es una venta online.

8. Modificar la clase Teatro (Ejercicio 8 TP 2) para que ahora las actividades del teatro puedan ser ahora: Teatro, Cine y Musicales. Es decir, las funciones ahora son actividades que tienen nombre, horario de inicio, duración y precio. Sin embargo, para las proyecciones de Cine hay que agregar el género y el país de origen de la película. Si en cambio se trata de un Musical hay que agregar el director del mismo y la cantidad de personas en escena. El teatro ahora requiere administrar el costo de usar el teatro para poder utilizar las instalaciones. Para ello se solicita implementar el método ***darCostos***, el cual determina según las actividades del teatro cuál debería ser el cobro obtenido. Para obtener el mismo, hay que tener en cuenta que se deben sumar los precios de cada tipo de actividad programada para un mes dado, y aplicar un incremento por actividad según se detalle:

8.1. Si es una obra de teatro: 45%

8.2. Si es un musical: 12%

8.3. Si es un película: 65%.