7. UD7-PR.01: Utilización avanzada de clases.

7.1 Gestión de cuentas bancarias 1
7.2 CONSIDERACIONES A TENER EN CUENTA 4
7.3 Recomendaciones 5
7.4 Formato de entrega 6
7.5 Recursos necesarios para realizar la tarea 6
7.6 Materiales 7
7.6.1 Recursos didácticos 7

Detalles de la tarea de esta unidad.

A lo largo de esta unidad has terminado de familiarizarte con el resto de los conceptos relacionados con la Programación Orientada a Objetos que faltaban por ver de una manera más formal y con ejemplos explícitos: composición; herencia; clases y métodos abstractos; sobreescritura de métodos; interfaces; polimorfismo; ligadura dinámica, etc.

Has experimentado con todos estos conceptos y los has utilizado en pequeñas aplicaciones para comprobar su funcionamiento y su utilidad.

Una vez finalizada la unidad se puede decir que tienes ya un dominio adecuado del lenguaje Java como un lenguaje que permite aplicar todas las posibilidades de la Programación Orientada a Objetos. Dado ese supuesto, esta tarea tendrá como objetivo escribir una pequeña aplicación en Java empleando algunas de las construcciones que has aprendido a utilizar.

7.1 Gestión de cuentas bancarias

Se trata de desarrollar una aplicación Java, denominada *BancoX* donde X será un nombre o palabra a tu elección.

Nuestro banco va a permitir gestionar varios tipos de cuentas bancarias. Mediante un menú se podrán elegir determinas operaciones:

1) *Abrir una nueva cuenta.*
2) *Ver un listado de las cuentas disponibles (código de cuenta, titular y saldo actual).*
3) *Obtener los datos de una cuenta concreta.*
4) *Realizar un ingreso en una cuenta.*
5) *Retirar efectivo de una cuenta.*
6) *Consultar el saldo actual de una cuenta.*
7) *Salir de la aplicación.*

Las cuentas se irán almacenando en alguna estructura en memoria según vayan siendo creadas. Esta estructura podrá contener un máximo de 100 cuentas bancarias. Cada cuenta será un objeto de una clase que contendrá la siguiente información:

- **Titular de la cuenta** (un objeto de la clase Persona, la cual contendrá información sobre el titular: nombre, apellidos y DNI).
- **Saldo** actual de la cuenta (número real).
- Número de cuenta (**IBAN**).
- **Tipo de interés anual** (si se trata de una cuenta de ahorro).
- **Lista de entidades** autorizadas para cobrar recibos de la cuenta (si se trata de una cuenta corriente).
- **Comisión de mantenimiento** (para el caso de una cuenta corriente personal).
- **Tipo de interés por descubierto** (si es una cuenta corriente de empresa).
- **Máximo descubierto permitido** (si se trata de una cuenta corriente de empresa)

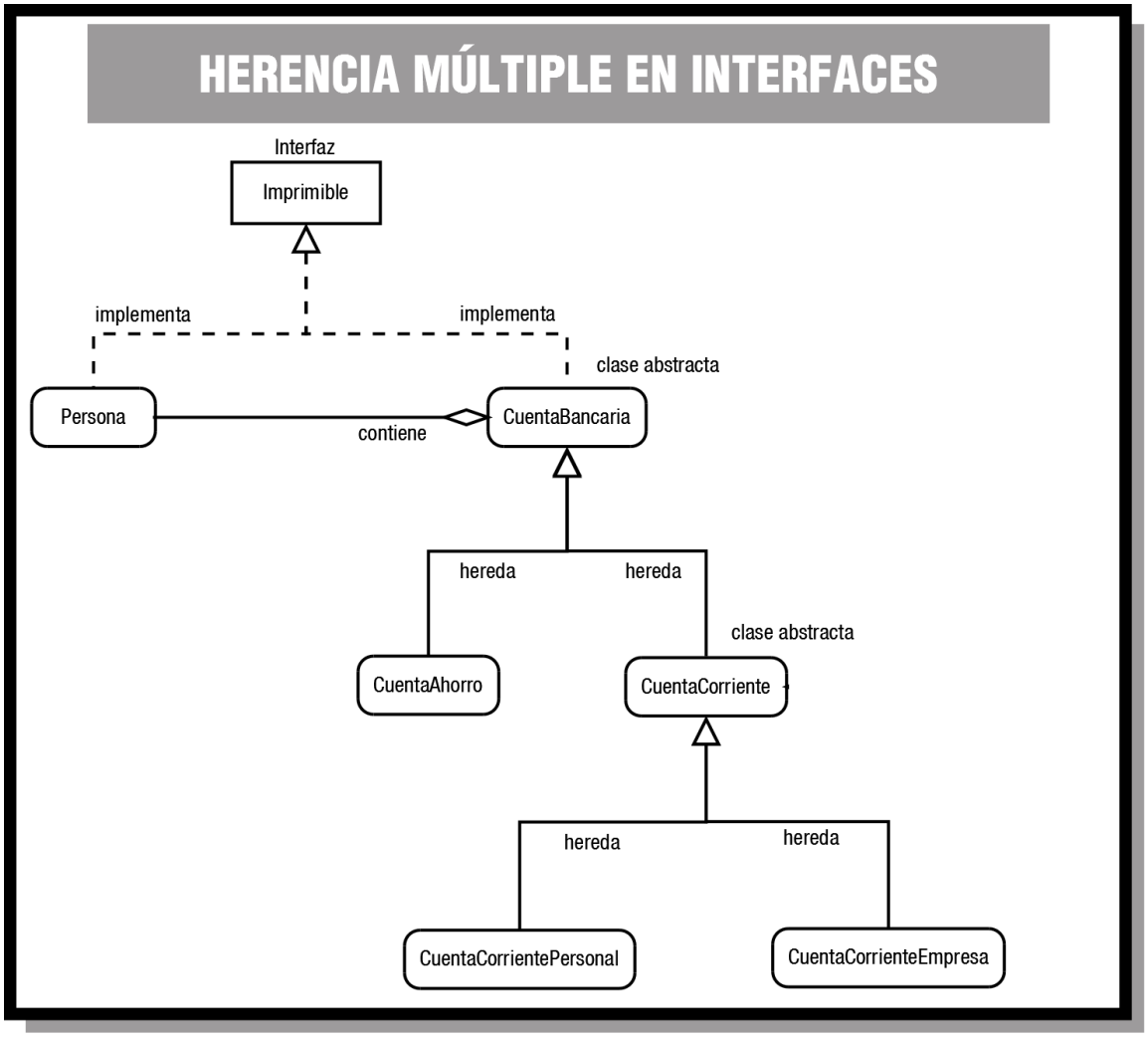
Las cuentas bancarias pueden ser de dos tipos:

- **cuentas de ahorro**: son remuneradas y tienen un determinado tipo de interés.
- **cuentas corrientes**: no son remuneradas, pero tienen asociada una lista de entidades autorizadas para cobrar recibos domiciliados en la cuenta.

  Dentro de las cuentas corrientes podemos encontrar a su vez otros dos tipos:

- *Las cuentas corrientes personales*, que tienen una comisión de mantenimiento (una cantidad fija anual).
- *Las cuentas corrientes de empresa*, no tienen comisión de mantenimiento. Además, las cuentas de empresa permiten tener una cierta cantidad de descubierto (máximo descubierto permitido) y por tanto un tipo de interés por descubierto y una comisión fija por cada descubierto que se tenga. Es el único tipo de cuenta que permite tener descubiertos.

Aquí tienes un ejemplo de una posible estructura de clases para llevar a cabo la aplicación:



Cuando se vaya a abrir una nueva cuenta bancaria, el usuario de la aplicación (empleado del banco) tendrá que solicitar al cliente:

- Datos personales: nombre, apellidos y DNI.

- **Tipo de cuenta** que desea abrir: cuenta de ahorro, cuenta corriente personal o cuenta corriente de empresa.
- **Saldo inicial**.

Además de esa información, el usuario de la aplicación deberá también incluir:

- **Número de cuenta (IBAN)** de la nueva cuenta, el cual se validará con una expresión regular y deberá tener formato ESNNNNNNNNNNNNNNNNNN, donde N es un dígito del 0 al 9.
- **Tipo de interés de remuneración**, si se trata de una cuenta de ahorro.
- **Comisión de mantenimiento**, si es una cuenta corriente personal.
- **Máximo descubierto permitido**, si se trata de una cuenta corriente de empresa.
- **Tipo de interés por descubierto**, en el caso de una cuenta corriente de empresa.
- **Comisión fija por cada descubierto**, también para el caso de una cuenta corriente de empresa.

7.2 CONSIDERACIONES A TENER EN CUENTA

El programa que escribas debe cumplir al menos los siguientes requisitos:

- ✓ Para almacenar los objetos de tipo cuenta deberás utilizar un array.
- ✓ Para trabajar con los datos personales, debes utilizar una clase Persona que contenga la información sobre los datos personales básicos del cliente (nombre, apellidos, DNI).
- ✓ Para guardar las entidades autorizadas en las cuentas corrientes utiliza una cadena de caracteres.

En cuanto a la organización del código, se deberán crear las siguientes clases:

- **Principal**: Contendrá el método main y todo el código de interacción con el usuario (lectura de teclado y salida por pantalla).
- **Banco**: mantendrá como atributo la estructura que almacena las cuentas. Dispondrá de los siguientes métodos:
  - → **Constructor o constructores**.
  - → **abrirCuenta**: recibe por parámetro un objeto CuentaBancaria y lo almacena en la estructura. Devuelve true o false indicando si la operación se realizó con éxito.
  - → **listadoCuentas**: no recibe parámetro y devuelve un array donde cada elemento es

una cadena que representa la información de una cuenta.

→ **informacionCuenta**: recibe un iban por parámetro y devuelve una cadena con la información de la cuenta o null si la cuenta no existe.

→ **ingresoCuenta**: recibe un iban por parámetro y una cantidad e ingresa la cantidad en la cuenta. Devuelve true o false indicando si la operación se realizó con éxito.

→ **retiradaCuenta**: recibe un iban por parámetro y una cantidad y trata de retirar la cantidad de la cuenta. Devuelve true o false indicando si la operación se realizó con éxito.

→ **obtenerSaldo**: Recibe un iban por parámetro y devuelve el saldo de la cuenta si existe. En caso contrario devuelve -1.

- Todas clases e interfaces necesarias para representan la jerarquía de cuentas.
  - → La interfaz Imprimible tan solo declarará el devolverInfoString, que devolverá la información de una cuenta como una cadena de caracteres.
- Otras clases que pudieran ser de utilidad.

El código fuente Java de cada clase debería incluir **comentarios** en cada **atributo** (o en cada conjunto de atributos) y **método** (o en cada conjunto de métodos del mismo tipo) indicando su utilidad.

Se debe entregar el proyecto de Netbeans completo. Para empaquetar un proyecto en Netbeans, utiliza la opción File -> Export Project de Netbeans: generarás un fichero .zip con el contenido completo del proyecto.

7.3 Recomendaciones

- Antes de comenzar a escribir código, lee con detenimiento varias veces el enunciado.
- Trata de entender la jerarquía de clases que se muestra en la especificación, es fundamental para comprender de qué clases se compone la aplicación.
- Ve probando la aplicación a medida que implementas. Es fundamental asegurarnos de que las partes que implementamos funcionan para poder hacer comprobaciones en nuevas partes de la aplicación.

- Un buen punto de partida es implementar la estructura de clases, comenzando por la parte superior.

7.4 Formato de entrega

Se deben entregar el proyecto de Netbeans completo. Para empaquetar un proyecto en Netbeans, utiliza la opción **File - Export Project de Netbeans**: generarás un fichero .zip con el contenido completo del proyecto.

Para la entrega de esta tarea debes generar un único fichero .zip/.rar que contenga todos los ficheros anteriormente indicados y que debes nombrar de la siguiente forma:

- Ciclo-módulo: DAM-PROG-
- número de unidad (UD7)
- identificación de la práctica (PR.01)
- apellidos, seguidos de tu nombre, separados por guión bajo.

"DAM-PROG-UD7-PR.01-apellido1_apellido2_nombre.xxx"

Ejemplo: DAM-PROG-UD7-PR.01-Fernandez_Lopez_Maria.pdf

Nota: Asegúrate que el nombre no contenga la letra ñ, tildes ni caracteres especiales extraños. Así por ejemplo la alumna Begoña Sánchez Mañas sería:

DAM-PROG-UD7-PR.01-Sanchez_Manas_Begona

7.5 Recursos necesarios para realizar la tarea

- Ordenador personal
- Sistema operativo Windows o Linux
- Conexión a Internet
- JDK y JRE de Java

- Entorno NetBeans

7.6 Materiales

7.6.1 Recursos didácticos

- Apuntes en el aula virtual.
- Ordenador personal y conexión a internet
- Manual Java IES San Clemente: https://manuais.iessanclemente.net/index.php/Java
- Aprenda Java como si estuviera en primero (Universidad de Navarra): http://ocw.uc3m.es/cursos-archivados/programacion-java/manuales/java2-U-Navarra.pdf/view
- Libro "Thinking in Java 4th edition" Bruce Eckel (Disponible en recursos del módulo)