# Tarea 5 programación

#### Enunciado

|  |
| --- |
| Se trata de desarrollar una aplicación Java en consola que permita gestionar una cuenta bancaria. Mediante un menú se podrán realizar determinas operaciones:   1. Ver el número de cuenta completo (CCC - Código Cuenta Cliente). 2. Ver el titular de la cuenta. 3. Ver el código de la entidad. 4. Ver el código de la oficina. 5. Ver el número de la cuenta (solamente el número de cuenta, sin entidad, oficina ni dígitos de control). 6. Ver los dígitos de control de la cuenta. 7. Realizar un ingreso. Habrá que solicitar por teclado la cantidad que se desea ingresar. 8. Retirar efectivo. Habrá que solicitar por teclado la cantidad que se desea retirar. 9. Consultar saldo. 10. Salir de la aplicación.   Antes de que aparezca este menú, el programa tendrá que solicitar al usuario los siguientes datos:  Nombre del titular de la cuenta (con un máximo de caracteres).  DNI del titular de la cuenta.  Código cuenta cliente (CCC) de la cuenta completo (entidad-oficina-dígitos de control-cuenta).  El programa deberá asegurarse que el CCC es válido mediante la comprobación de:  El formato (cuatro dígitos de entidad, cuatro dígitos de oficina, dos dígitos de control y diez dígitos de número de cuenta).  Los dígitos de control son válidos.  Además del programa principal de la aplicación (clase con una función main), habrá que escribir una clase CuentaBancaria que proporcione todas las herramientas necesarias para trabajar con este tipo de información:  Constructor (o constructores) adecuados.  Almacenamiento del nombre del titular (atributos).  Almacenamiento del DNI del titular (atributos).  Almacenamiento del código de cuenta (atributos).  Almacenamiento del saldo actual (atributos).  Gestión de ingresos y depósitos (métodos de interfaz pública).  Obtención del saldo (métodos de interfaz pública).  Obtención de información sobre la cuenta: número de la cuenta, entidad, oficina, titular, etc. (métodos de interfaz pública).  Aquellas herramientas auxiliares necesarias para poder trabajar cómodamente con el objeto. Algunas de esas herramientas podrán ser públicos y otras quizá no. Algunas podrán ser específicas de clase y otras podrán ser de objeto (métodos de objeto privados, métodos estáticos públicos, etc.).  Para trabajar con el número de cuenta debes utilizar el modelo de Código Cuenta Cliente (CCC), que está formado por cuatro campos: entidad - sucursal - dígito de control - número de cuenta. La idea es que puedas introducir el código de cuenta completo y que la clase disponga de un mecanismo para comprobar que ese código es válido. Si el código no es válido, se debería generar una excepción (y por supuesto no almacenar ese código de cuenta). Para ello podrías tener, por ejemplo, un método estático que permita validar códigos de cuenta.  En general, deberías incluir excepciones para controlar aquellos casos en los que el uso de un método no sea posible (intentar sacar más dinero del que hay en el saldo, intentar introducir un titular con más caracteres de los permitidos, intentar ingresar o retirar una cantidad negativa, etc.).  El código fuente Java de esta 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. El programa principal también debería incluir algunos comentarios explicativos sobre su funcionamiento y la utilización de objetos de la clase CuentaBancaria.  Además del programa deberás escribir también un informe con todas las consideraciones oportunas que se necesiten para entender cómo has realizado la tarea.  El proyecto deberá contener al menos dos archivos fuente Java:  Programa principal (clase con método main: AplicacionCuentaBancaria.java).  La clase CuentaBancaria (CuentaBancaria.java).  El documento que contendrá el informe lo elaborarás con un procesador de texto. Debe ser de tipo ".doc" (Microsoft Word) o de tipo ".odt" (OpenOffice.org). Debe tener tamaño de página A4, estilo de letra Times New Roman, tamaño 12 e interlineado normal. |

### Explicación

#### CuentaBancaria

La clase CuentaBancaria se ha optado por incluir entre sus variables de clase los datos que tiene que introducir el cliente: nombre y el DNI, además de su cuenta bancaria, pero como la cuenta bancaria se forma de varias partes como el código de la entidad, de la oficina, dígito de control y número de cuenta que serán pedidas por el usuario de la aplicación se ha decidido ponerlas como variables separadas.

Por otra parte, necesitaremos una serie de datos para poder verificar que el DNI y la cuenta bancaria que introduce el cliente es verdadera por lo que usaremos “LETRAS\_DNI” para calcular la letra de DNI es la misma que ha introducido el cliente y una lista de números que nos sirve para calcular los dígitos de control, al igual que el importe mínimo y máximo que puede sacar el cliente, que nos servirá a la hora de lanzar excepciones. Estas cuatro variables mencionadas anteriormente serán static y privadas para que solamente sean accesibles desde la clase CuentaBancaria ya que son usadas únicamente por métodos creados en la misma clase.

Para simplificar el trabajo al cliente le pediremos el nombre, DNI y número de cuenta (este último sin separar sus diferentes partes ya que esto lo haremos llamando a otro constructor que recibe el codigoEntidad, codigoOficina, digitoControl y numeroCuenta como parámetros que se los pasaremos haciendo un substring del número de cuenta que ha metido el cliente)

En los métodos de ingresar y retirar dinero se tienen en cuenta una serie de posibles excepciones que se pueden dar como que el cliente introduzca cifras vengativas, números iguales a cero o negativos o cifras muy altas (más de 10.000 euros en el caso de ingresar y 1.000 de retirar) para asemejarse a los límites que pone un banco. Además, en el caso de retirar, solo se podrán hacer con cifras que sean múltiplos de 10 y el mínimo deberán ser 10 euros porque los cajeros solo tienen billetes en su gran mayoría.

#### AplicaciónCuentaBancaria

Esta aplicación se divide en dos partes. La primera donde el cliente introduce sus datos. Donde en el caso de nombre se contempla que la longitud sea entre 3 y 100 caracteres y se valida con un método que únicamente devolverá true o false si cumple este requisito.

Entrada de DNI y número de cuenta. Para estos dos se ha optado por un mecanismo similar donde se realiza un do-while para pedir al cliente los datos hasta que los introduzca adecuadamente y a su vez estos se han puesto dentro de un do-while que se repite hasta que el booleano volver sea false. Este booleano se ha declarado al inicio de la clase y se pone a false antes de que el cliente introduzca los datos, si se produce alguna excepción en alguna de las validaciones. Por ejemplo, la letra del DNI calculado no coincide con la introducida por el cliente o el dígito de control del número de cuenta no coincide con el introducido para que no se corte la ejecución del programa dentro de los catch se pone al booleano en true para que vuelva a pedir los datos al cliente hasta que sean correctos. De esta forma ahorramos al cliente que vuelva a meter todos los datos por poner un dato erróneo.

Validaciones

* DNI. La letra del DNI se calcula cogiendo los 8 primeros dígitos y dividiéndolo entre 23 y el resto de esa división corresponde a una de las letras que tenemos especificadas dentro de la variable estática de la clase Cuenta Bancaria LETRAS\_DNI
* Número de cuenta. Los dígitos de control se calculan cogiendo los números antes y después del dígito de control. La primera parte serán 8 números por lo que se concatenan dos delante y cada número se multiplica por cada número de control y después se sumarían. Por ejemplo *4651 0444 26 4435436057*

La primera parte es 4651 0444 se concatenarían dos ceros quedando 00 4651 0444 y los factores son 1, 2, 4, 8, 5, 10, 9, 7, 3, 6 por lo que se multiplica 1 \* 0 + 2 \* 0 + 4 \* 4 + 6 \* 8 + 5 \* 5 + 10 \* 1 + 9 \* 0 + 7 \* 4 + 3 \* 4 + 6 \* 4 = 163 y ese número se divide entre 11 quedando de resto 9. Y el dígito de control sería 11 – 9 = 2 y ese sería el primer número de control. Si el resto tiene más de una cifra como sería 10 el dígito de control sería 1 y para la segunda parte del número de cuenta, es decir 4435436057 se realizaría el mismo proceso para tener los dos números.

El resto de la aplicación funciona como si fuera un cajero automático donde con un switch se consigue que cada opción que meta realice una función y en el caso de que meta una opción no válida entre en el default donde le pedirá una opción hasta que introduzca una válida