1. Notas

Escribe un programa que lea una secuencia de notas que se van introduciendo por el teclado (*valores numéricos que pueden contener decimales*). La secuencia de notas finalizará cuando se introduzca un valor negativo. El programa *informará al final* del número de aprobados, el número de suspensos y la nota media.
El programa debe controlar posibles errores producidos al introducir las notas por teclado de tal forma que si alguna nota introducida fuera errónea la vuelva a solicitar.

2. Calculadora

Diseña un programa que imite a una calculadora capaz de realizar las operaciones básicas (suma, resta, producto y multiplicación) y dos operaciones complejas como la potencia y la raíz cuadrada. La aplicación debe mostrar un menú como este:
Calculadora Evolution
Elige una opción:
1.- Operaciones básicas.
2.- Operaciones complejas.
3.- Salir.

Si seleccionamos la opción 1, deberá mostrar:
a. Suma.
b. Resta.
c. Producto.
d. División.
Para el caso de la opción 2:
a. Potencia.
b. Raíz cuadrada.

La tercera opción, finalizará con la ejecución del programa.

Una vez elegida la opción correspondiente, el programa deberá solicitar los datos necesarios y mostrar por pantalla el resultado obtenido en cada caso.

El programa deberá controlar los posibles errores que se puedan producir a la hora de elegir la opción y de introducir los datos por teclado, de tal forma que, si los datos introducidos fueran erróneos, vuelva a solicitarlos.

Se deberá gestionar adecuadamente la encapsulación y herencia de cada una de las clases necesarias. El diseño del menú correrá bajo tu responsabilidad, ya que esto dependerá en gran medida, de los posibles fallos que puedas tener.

3. Biblioteca

Crea un proyecto que contenga una clase Libro y una clase Biblioteca con las siguientes características:

- La clase **Libro** contendrá tres atributos private de tipo String:
  - ✓ titulo: el título del libro.
  - ✓ autor: el autor del libro.
  - ✓ isbn: el código ISBN del libro.

- Y los siguientes métodos:
  - ✓ `public Libro(String titulo, String autor, String isbn)`: El constructor de la clase inicializa los atributos del objeto con los datos pasados por parámetro. Es importante comprobar que el objeto se genera adecuadamente, por lo que si el ISBN pasado por parámetro no es válido, se lanzará la excepción IllegalArgumentException. Así se genera la excepción:
    - ✦ `throw new IllegalArgumentException("isbn no válido");`
  - ✓ Métodos `consulta_XXX()` y `cambia_XXX()`: se creará un método consulta/cambia para cada uno de los atributos. El método cambia_isbn() deberá comprobar la validez y en caso de que no sea válido generará la excepción IllegalArgumentException (similar al constructor).
  - ✓ `private boolean compruebaIsbn10(String isbn):` Este método se utilizará en el contructor y en cambia_isbn() para comprobar la validez del ISBN. Devolverá true si el parámetro es un ISBN-10 válido, y false en caso contrario.
  - ✓ `private boolean compruebaIsbn13(String isbn):` Este método se utilizará en el contructor de la clase y en cambia_isbn() para comprobar la validez del ISBN. Devolverá true si el parámetro es un ISBN-13 válido, y false en caso contrario.
- La clase **Biblioteca** contendrá la función main que hará uso de los métodos públicos de la clase Libro. Para comprobar si un isbn introducido es válido, habría que capturar la excepción IllegalArgumentException en la llamada al constructor (bloque try-catch).Cómo comprobar la validez de un código ISBN:
  - ✓ El código ISBN de un libro puede contener 10 o 13 caracteres (desde 2007). El ISBN es un código de detección de errores por lo que es posible comprobar la validez del mismo. Ejemplo:
    a) **ISBN-10**: Debemos multiplicar cada dígito (si el último carácter fuera una X, su valor es 10) por la posición que ocupa y calcular la suma de todos ellos. Si la suma obtenida es múltiplo de 11, el isbn es válido. Ejemplo:
       1) Comprueba la validez del siguiente isbn-10: 84-481-2231-3
       2) Suma: 8*1 + 4*2 + 4*3 + 8*4 + 1*5 + 2*6 + 2*7 + 3*8 + 1*9 + 3*10 = 154
       3) Comprobación de que la suma es múltiplo de 11: 154 / 11 = 14 (el resto es 0)

    b) **ISBN-13**: Debemos multiplicar cada número por 1 si la posición que ocupa es impar o por 3 si la posición que ocupa es par y calcular la suma de todos salvo del último que es el dígito de control. Dividimos el resultado de la suma anterior entre 10 y obtenemos el resto de la división. Restamos a 10 el resto obtenido. El isbn será válido si el valor obtenido es igual al dígito de control (último dígito del isbn). Ejemplo:
       1) Comprueba la validez del siguiente isbn-13: 978-84-415-2682-2
       2) Suma: 9*1 + 7*3 + 8*1 + 8*3 + 4*1 + 4*3 + 1*1 + 5*3 + 2*1 + 6*3 + 8*1 + 2*3 = 128
       3) Cálculo del dígito de control:128 mod 10 = 8 (el resto). 10 – 8 = 2 (coincide con el dígito de control).

Recursos:

- Gaussianos
- Monografias

4. **Cuenta bancaria**

Desarrollar una aplicación 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).
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.

Para ello, se crearán dos clases:

Clase CuentaBancaria: Una cuenta bancaria se compondrá de una persona titular de la cuenta, el saldo en euros, y un código de cuenta (el CCC o Código Cuenta Cliente). El CCC consta de 20 dígitos: 4 dígitos que representan el código de la entidad, 4 dígitos que representan el código de la oficina, 2 dígitos de control y 10 dígitos con el número de la cuenta. Los dos dígitos de control se calculan con los valores de la entidad, la oficina y el número de cuenta por lo que no será necesario almacenarlos.

También deseamos que el nombre del titular tenga una longitud máxima de 100 caracteres y una longitud mínima de 10 caracteres.

Por tanto, el diseño de la clase se ajustará a:

Atributos:

Privados: titular, saldo, entidad, oficina y numCuenta.

De clase (static) públicos: constantes con el tamaño máximo y mínimo del nombre del titular.

Métodos:

Públicos:

Constructor con los parámetros básicos de la cuenta corriente. Si alguno de los parámetros y/o el CCC no es válido se lanzará la excepción IllegalArgumentException. public CuentaBancaria(String titular, String entidad, String oficina, String DC, String numCuenta).

Constructor con los parámetros titular y CCC sin descomponer. Este constructor descompondrá el CCC y llamará al constructor básico mostrado anteriormente.public CuentaBancaria(String titular, String CCC).

Getters para los atributos: titular, saldo, entidad, oficina y numCuenta.

Setters para los atributos: titular. Se debe comprobar la validez del nuevo titular y lanzar la excepción IllegalArgumentException si no se verifica.

Método ingresar. Ingresará el dinero correspondiente al saldo de la cuenta. Este valor debe ser positivo, en caso contrario se lanzará la excepción IllegalArgumentException. public void ingresar(double cantidad).

Método retirar. Se restará la cantidad indicada del saldo de la cuenta. Este valor debe ser positivo y nunca superior al saldo de la cuenta. De no ser así se lanzará la excepción IllegalArgumentException.public void retirar(double cantidad).

Método comprobar la validez de un CCC. El método devolverá true o false si el CCC indicando si el CCC es válido o no.public static boolean comprobarCCC(String CCC).

Método para calcular los dígitos de control de un CCC. El método devuelve los dos dígitos de control dado los códigos de entidad, oficina y número de

cuenta. public static String obtenerDigitosControl(String entidad, String oficina, String numCuenta).
Método `String toString()`. Devolverá una cadena con los datos del titular, CCC y saldo de la cuenta.

Todos aquellos métodos que implementados que puedas pasar a privado, mejorarán la encapsulación, hazlo con aquellos casos que creas y, evita la duplicación del código.

Relativo al método cálculo de los dígitos de control de un CCC:

El primer dígito de control se calcula con los datos de la entidad y la oficina mientras que el segundo dígito de control se calcula con el número de cuenta. El procedimiento es el mismo en ambos casos.

Supongamos que vamos a calcular el CCC para la cuenta 1234 5678 - - 1234567890
Debemos multiplicar los dígitos por unos pesos en función de la posición que ocupan. Los pesos son los siguiente: 1, 2, 4, 8, 5, 10, 9, 7, 3, 6 y sumar los productos obtenidos. Para el caso del primer dígito, como la entidad y la oficina tienen 8 dígitos, se rellena con ceros por la izquierda. Calculemos los productos y la suma de los mismos:

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| 1 | 2 | 4 | 8 | 5 | 10 | 9 | 7 | 3 | 6 |
| **0** | **0** | **4** | **16** | **15** | **40** | **45** | **42** | **21** | **48** |

4+16+15+40+45+42+21+48 = 231

La suma obtenida la dividimos por 11 y nos quedamos con el resto. Posteriormente, a 11 le restamos el resto anterior y ese será el dígito de control. En caso de que el resultado fuera 10, el dígito sería 1 y si fuera 11, tomaríamos como dígito el 0.

231 % 11 = 0
11 – 0 = 11

Por tanto, el dígito de control sería 0.
De igual forma calcularíamos el segundo dígito de control que debe ser 6.

El programa principal, se encarga de gestionar la solicitud de datos al usuario y la creación de una cuenta bancaria. Las opciones disponibles se mostrarán en un menú como el mostrado al principio de esta tarea.
Se deben comprobar y manejar los errores de forma adecuada para evitar que el programa termine de forma inesperada. Por tanto, se comprobará que el usuario introduzca valores correctos en todo momento.

Recursos:

- Código cuentas clientes
- CCC wikipedia