

TAREA DE PROG05

Curso 2018-19

ENUNCIADO

- A lo largo de esta unidad habéis aprendido a crear clases, así como sus distintos miembros (atributos y métodos). Se ha experimentado con la encapsulación y accesibilidad (modificadores de acceso a miembros), se han creado miembros estáticos (de clase) y de instancia (de objeto), se han escrito constructores para las clases, se han sobrecargado métodos y se han utilizado en pequeñas aplicaciones. También se ha tenido un primer encuentro el concepto de herencia, que se desarrollarán en unidades más avanzadas junto con otros conceptos avanzados de la Programación Orientada a Objetos.
- En esta tarea hay que entregar:
 - Un proyecto en NetBeans llamado **Tarea_Prog05_Apellido1Apellido2Nombre** (Nombre es vuestro nombre y primer apellido), en el que se realizará las clases Java que se enumeren mas abajo.
 - Un documento **PDF** llamado: **Tarea_Prog05_Apellido1Apellido2Nombre.pdf**, en el que pondremos **captura de pantallas** en la que aparezca como **fondo** vuestra conexión **al Papas** y como primer plano las capturas del código fuente y los resultados de la ejecución del ejercicio (tal y como se vería con Netbeans), para así mostrar que funcionan (debe aparecer una prueba ejecutada de cada ejercicio). También realizaremos una explicación de lo que has realizado en la tarea.

Clase GestionLibreria

En esta clase realizaremos lo siguiente dentro del método **main**:

1. Solicitar al usuario los siguientes datos:
 - **Título** del libro (con un máximo de 40 caracteres). Se introducirá por teclado el título del libro, para ello llamaremos al método **leerTeclado**, explicado mas abajo. A continuación llamará al método **Libro.validarTitulo** que devolverá true si el título es válido (longitud entre 1 y 40) y false si no lo es. No continuará hasta que sea válido el título.
 - **Código del Libro Completo (CLIBC)** es el código del libro completo. Se introducirá por teclado (llamando al método **leerTeclado**) un String, que será el código completo de 12 caracteres numéricos que estará compuesto por: la estantería (3 caracteres), balda (2 caracteres), número del libro (5 caracteres) y 2 dígitos control.

El programa deberá asegurarse que el **CLIBC** sea válido (no continuará hasta que lo sea), mediante la comprobación de:

- El **formato** :12 dígitos: 3 dígitos de estantería, 2 dígitos para la balda, 5 para el número de libro y 2 dígitos de control. Para ellos llamaremos al método **Libro.validarCLIBC**, al que le pasaremos la cadena introducida por teclado y nos dirá si es correcto o no (longitud debe ser 12 y que lo teclado sea numérico).
- Los **dígitos de control** son válidos. Para comprobar que los dígitos de control sean válidos, tiene que cumplir que sumando: estantería+balda + nº libro y dividiendo por 99, cogemos el resto. El resto serán los dígitos de control. Llamaremos al método **Libro.validarDigitosControl**, al que se le pasa CLIBC y comprueba que los dígitos de control sean correctos.

Si el código no es válido, se debería generar una **excepción** (y por supuesto no almacenar ese código de artículo). No continuará hasta que sea válido CLIBC.

Una vez que los datos introducidos sean correctos crearemos el objeto **libroActual**, pasándole al constructor CLIBC y el título del libro.

2. Mediante un **menú** se podrán realizar las siguientes operaciones:
 1. **Ver el código del libro completo (CLIBC- Código Libro).**
 2. **Ver el título del libro.**
 3. **Ver la estantería.**
 4. **Ver la balda**
 5. **Ver el número del libro** (solamente el número de artículo, sin almacén ni estantería).
 6. **Ver los dígitos de control del libro.**
 7. **Realizar altas de libros.** Habrá que solicitar por teclado las unidades que se desean dar de alta. Llamaremos al método **altaLibros**. Hay que recoger la excepción en caso de que se produzca algún error.
 8. **Realizar ventas de libros.** Habrá que solicitar por teclado las unidades que se desean vender. Llamaremos al método **ventaLibros**. Hay que recoger la excepción en caso de que se produzca algún error.
 9. **Consultar unidades totales.**
 10. **Salir de la aplicación.**

La visualización del Menu lo introduciremos en un método llamado: **visualizarMenu** (será privado) Introduciremos una opción llamando al método **leerOpcion** (explicado después), realizaremos las sentencias para realizar cada una de las opciones y finalmente llamaremos al método **pulsarTecla** (explicado después) , por el cual el programa se parará hasta que pulsemos una tecla.

Dentro de la clase **GestionLibreia** crearemos los siguientes métodos privados y estáticos:

- String **leerTeclado()**: en el que mediante `InputStreamReader` y `readLine`, nos devuelva un `String` con lo tecleado. Pondremos un `try-catch` y propagaremos la excepción, si salta alguna.
- int **leerOpcion** (): método por el que llamando al anterior nos devuelva una opción válida (entre 0 y 9). Pondremos un `try-catch` y propagaremos la excepción, si salta `NumberFormatException`, `IOException` o cualquier otra.
- int **pulsarTecla()**: método por medio de `InputStreamReader` y `read()` pulsaremos una tecla. Pondremos un `try-catch` y propagaremos la excepción, si salta alguna

Clase Libro

Esta clase debe proponer todas las herramientas necesarias para almacenar y trabajar con la siguiente información sobre los libros vendidos por una librería

- **Atributos:**
 - Código del artículo: se almacenará mediante 3 atributos:
 - **estateria**: estantería donde están almacenado el libro (3 dígitos numéricos), almacenado como cadena.
 - **balda**: balda donde está el libro (2 dígitos numéricos, almacenado como cadena
 - **numeroLibro**: número individual del libro (5 dígitos numéricos), almacenado como cadena.
 - Los dígitos de control no se almacenan porque se pueden calcular.
 - **titulo**: título del libro (cadena de 40 caracteres)
 - **unidadesTotales**: Unidades totales que hay de ese libro en el almacén (entero de 5 caracteres).
- **Métodos Privados:**
 - **establecerLibro**: le pasaremos como entrada una cadena con el CLIBC (código de libro completo) y se separarán los 3 atributos: estantería, balda y `codigoLibro` y los almacenará en los atributos del objeto
 - **establecerTitulo**: pasándole como entrada una cadena con el título, comprobará que sea correcta llamando al método `validarTitulo` y lo almacenará en el atributo `titulo`.

- **Constructor:** le pasaremos el CLIBC y el título y llamando a los métodos **establecerLibro** y **establecerTitulo**, asignará el valor a los atributos.
- **Constantes:** tendremos 4 constantes: DIGITOS_ESTANTERIA, valor 2, DIGITOS_BALDA, valor 3, DIGITOS_NRO_LIBRO valor 5, DIGITOS_DC, valor 2. Se utilizarán donde se necesite.
- **Métodos estáticos y públicos:**
 - **calcularDigitosControl:** al que le pasaremos: estantería, balda y codigoLibro y nos devolverá los dígitos de control como cadena. Para calcular los dígitos de control sumaremos: estantería+balda + nº libro, se divide por 99 y se coge el resto. El resto son los dígitos de control (método estático y público)
 - **validarDigitosControl:** le pasaremos como entrada al método un CLIBC y nos devolverá true si los dígitos son correctos y false si no lo son. Para comprobar que los dígitos de control sean válidos, en primer lugar, mediante el método **CLIBC.substring(inicio,final)**, sacaremos de CLIBC: estantería, balda, nº libro y los dígitos de control. Sumaremos: estantería+balda + nº libro, se divide por 99 y se coge el resto. El resto serán los dígitos de control, que deben coincidir con los dos dígitos de control de CLIBC (método estático y público). Si coinciden, devolverá true, si no coincide false.
 - **validarCLIBC:** le pasaremos una cadena CLIBC y nos devolverá true si es correcto (longitud =12 y que sea numérico) y false si no lo es. (método estático y público)
 - **validarTitulo:** le pasaremos una cadena con el título del libro y nos devolverá true si el título es válido (longitud entre 1 y 40) y false si no lo es.
- **Métodos de Interfaz: públicos**
 - Obtener el valor de los atributos: **getEstanteria**, **getBalda**, **getNumeroLibro**, **getDigitosControl** (Llamará a método calcularDigitosControl) , **getTitulo**, **getUnidadesTotales**, **getCLIBC** (lo compondrá concatenando la estantería, la balda, el nº de libro y lo devuelto por el método calcularDigitosControl.
 - **altaLibros:** se le pasa como entrada las unidades y se les sumarán a las unidades totales. Controlaremos que las unidades sean >0, en caso de no serlo levantaremos una excepción.
 - **ventaLibros** se le pasa como entrada las unidades y se les restarán a las unidades totales. Controlaremos que las unidades sean > 0 y que sean < que las unidades totales en almacén, en cada caso levantaremos una excepción.
- Si crees conveniente introducir algún otro método, justifica su necesidad.

Utilizaremos el método **Integer.parseInt(String cadena)** para convertir una cadena en un entero.

El código fuente Java debe 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 **Libro**

CRITERIOS DE PUNTUACIÓN

Para poder empezar a aplicar estos criterios es necesario que la aplicación compile y se ejecute correctamente en un emulador. En caso contrario la puntuación será directamente de 1,00.

Clase Libro : 4.5 puntos

Clase: GestionLibreria: 4,5 puntos

Aspectos generales : 1 puntos: presentación del informe , buen diseño proyecto, bien estructurado,etc.

En la puntuación total se podrá descontar por lo siguiente:

- **-0,5 puntos: No** se han incluido **comentarios** describiendo el funcionamiento de todos los métodos y atributos en ambas clases, como se ha pedido en el enunciado
- **-1 punto: No** se ha entregado el **informe** explicativo o se trata de un informe explicativo insuficiente
- **- 1 punto:** Alguna de las **opciones de menú** pedidas en el enunciado (menú del programa principal) **no** funciona correctamente: **por cada opción que no funcione.**

Dado que algunos criterios de puntuación son negativos, podría suceder que el balance final fuera negativo. En tal caso la puntuación final será simplemente de 1,00

RECURSOS NECESARIOS PARA REALIZAR LA TAREA.

- Ordenador personal.
- JDK y JRE de Java SE.
- Entorno de desarrollo NetBeans con las funcionalidades necesarias para desarrollar y emular midlets.

CONSEJOS Y RECOMENDACIONES.

Para realizar la aplicación te realizamos la siguiente serie de recomendaciones:

- Básate en los diferentes ejemplos que has tenido que probar durante el estudio de la unidad. Algunos de ellos te podrán servir de mucha ayuda, así que aprovéchalos.
- El ejercicio resuelto de la clase DNI, en el cual se hacen comprobaciones de entrada, puede servirte de base para la comprobación de la validez de un **CLIBC**.

INDICACIONES DE ENTREGA.

- Lo que debes entregar:
 - Un proyecto en **NetBeans** llamado: **Tarea_Prog05_Apellido1Apellido2Nombre**
 - Además del proyecto crearemos un documento **PDF** llamado: **Tarea_Prog05_Apellido1Apellido2Nombre.pdf** en el que pondremos **captura de pantallas** en la que aparezca como **fondo** vuestra conexión **al Papas** y como primer plano con las capturas del código fuente y la ejecución del ejercicio (tal y como se vería con Netbeans), para así mostrar que funcionan (debe aparecer una prueba ejecutada del ejercicio).
- Comprimir el proyecto NetBeans y el documento pdf creados en un fichero llamado: **Tarea_Prog05_Apellido1Apellido2Nombre.zip**
- Asegúrate que el nombre no contenga la letra ñ, tildes ni caracteres especiales extraños
- En caso de tener que realizar un segundo envío, le daremos el siguiente nombre: **Tarea_Prog05_Apellido1Apellido2Nombre_ENVIO2.zip**