TAREA PARA PROG06
Curso 2023-24
ALMACENANDO DATOS

ENUNCIADO

- **La tarea no se dará por entregada si el proyecto tiene errores de sintaxis, no compila, no se ejecuta correctamente, si lo entregado no se corresponde con lo pedido o si no se ha realizado un mínimo de la tarea, es decir, si está vacía o casi.**

- En esta tarea hay que entregar:
  - Un proyecto en NetBeans llamado **Tarea_Prog06_Apellido1Apellido2Nombre** (Nombre es vuestro nombre y primer apellido), que gestione la **venta de muebles de un almacén**.
  - Un documento **PDF** llamado: **Tarea_Prog06_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 del ejercicio).
- Se trata de realizar un proyecto Java llamado: **Tarea_Prog06_Apellido1_Apellido2_Nombre** que gestione la **venta de muebles de un almacén**.
- Esos datos se almacenarán en un **fichero de objetos serializado**, denominado **MUEBLES_OBJ.DAT**.

- Los datos que necesitamos almacenar de los **muebles** son:
  - **idMueble:** número entero de 5 caracteres
  - **descripMueble:** cadena de 30 caracteres
  - **precioUniario:** número real.
  - **unidadesAlmacen:** número entero de 5 caracteres.(unidades que hay de ese mueble en el almacén)
  - **unidadesMinimas:** número entero de 5 caracteres (unidades mínimas que debe de haber del mueble)
  - **tipoMueble**: 1 carácter (Valores corretos:H: Hogar/D:Despacho/C:Colegios)

- Crearemos 3 paquete:
  - **controlador:** que tendrá una clase llamada **Utilidades**.
  - **modelos:** tendrá una clase llamada **Mueble**
  - **app:** que tendrá una clase llamada **AppGesionMueblesNombre** (Nombre es vuestro nombre)

Clase Utilidades:

Tendremos los siguientes métodos estáticos y públicos:

- void **visualizarMensaje(**String mensaje**)**: que le demos como entrada el mesaje que queremos visualizar.
- void **pulsarReturn()**, el programa se parará hasta que pulsemos la tecla Return (Enter) para seguir. Llamaremos al método visualizarMensaje con el mensaje "Pulse Return para seguir...."
- String **leerTeclado(**String mensaje**)** : visualizamos el mensaje mediante **visualizarMensaje** y nos permitirá introducir una cadena por teclado y devolverla como salida. Se utilizará BufferedReader y InputStreamReader.
- int **leerOpcion(**int limite**)**: dándole como entrada un límite, nos devuelve una opción que tecleemos entre 1 y el límite que le indiquemos. No saldrá hasta que lo tecleado esté entre 1 y límite. Llamaremos al método **leerTeclado** con el mensaje "Teclee una opción (1...6):"

Clase: Mueble

Realizaremos una clase llamada **Mueble, serializable** que tendrá los atributos para poder almacenar los datos de los muebles (idMueble,descipMueble,precioUnitario, unidadesAlmacen, unidadesMinimas y tipoMueble. Serán privados) y los getters y setters de todos los atributos. Que nos servirá para grabar los datos en el fichero de objetos.

Clase: AppGesionMueblesNombre

Incluiremos una clase llamada **AppGesionMueblesNombre** (Nombre es vuestro nombre), en la que mediante un menú se podrán realizar las siguientes operaciones:

1. Añadir Mueble
2. Listar todos los muebles
3. Buscar un mueble
4. Borrar un mueble
5. Borrar todos los datos del fichero
6. Salir.

Se creará un método llamado **visualizaMenu**, para visualizar las opciones por pantalla.
Cada opción del menú se creará en un método:

insertarMueble()
listarMuebles()
buscarMueble()
eliminarMueble()
eliminarFichero()

Después de visualizar el menú, pediremos la opción llamando al método **leeOpcion** poniéndole como límite=6, ejecutaremos la opción correspondiente y llamaremos al método **pulsarReturn**.

Esta clase tendrá dos métodos estáticos y privados:

- boolean **existeRegistro(**int idMueble**)**: le daremos como entrada un **idMueble**, lo buscaremos en el fichero MUEBLES.DAT, si existe devolverá **true** y si no existe devolverá **false**.
- Mueble **buscarRegistro(**int idMueble**)**: le daremos como entrada un idMueble, si existe devolverá el obejeto de tipo mueble del idMueble introducido, si no existe devuelve null.

1. **Añadir Mueble insertarMueble():** Esta opción pedirá los datos de la venta y añadirá el registro correspondiente en el fichero.
   Antes de añadir los datos hay que controlar:
   - Para cada atributo se creará un método para introducirlo, por ejemplo: **idMueble=introducirIdMueble();**
   - En primer lugar, se tecleará el **idMueble** y se controlará que no esté en el fichero MUEBLES.DAT llamando al método **existeRegistro**. En caso de que exista, visualizaremos el mensaje ESE REGISTRO YA EXISTE y volveremos a teclear el **idMueble** hasta que no exista o pulsemos 0 para volver a menú.
   - El **idMueble**, **descripMueble** y **precioUnitario** no pueden estar vacíos, si es así visualizaremos un mensaje y no dejaremos seguir hasta que sean correctos.
   - **tipoMueble**: aparecerá el siguiente mensaje: Tipo Mueble: :H: Hogar/D:Despacho/C:Colegios, teclearemos el tipo. Se almacenará en el fichero H, D o C (en mayúsculas). Si lo tecleado no es correcto, visualizaremos el mensaje: TIPO CORRECTO: H: HOGAR /D:DESPACHO /C:COLEGIOS y no dejaremos seguir hasta que sea correcto.
   - Las **unidadesAlmacen** deben ser mayores o iguales que **unidadesMininas**, si no es así visualizaremos un mensaje y volveremos a introducir las unidadesAlmacen y unidadesMinimas.
   - Se controlará para cada campo que **tipo de datos y la longitud** sean correctos.

   Se preguntará si se desea grabar los datos, si responde que si, se grabarán en el fichero y si se responde que no, no se grabarán. (Controlaremos que lo tecleado sea correcto S,s, N o n.)
   Finalmente, se preguntará si se desea introducir otro registro, si responde que si, se volverá a introducir todos los datos de nuevo y si se responde que no volverá al menú.

2. **Listar Muebles listarMuebles()**. Recorrerá el fichero mostrando por pantalla todos registros almacenados en el mismo.
   En TIPO MUEBLE, en vez poner H, D o C (que es lo que está almacenado en el fichero), visualizaremos HOGAR, DESPACHO o COLEGIOS

```
ID MUEBLE  NOMBRE               TIPO     PRECIO UNITARIO  UNID.ALMACEN  TOTAL
=========  ==================== ======== ===============  ============ ==============
1          MESA COMEDOR         HOGAR      120,45           100          12045
2          MESA REUNIONES       DESPACHO   350,00            50          17500
```

   El TOTAL será el resultado de multiplicar las unidades en almacén por el precio unitario.

3. **Buscar Mueble: buscarMueble()**. Pedirá al usuario que teclee el **idMueble** del mueble que queremos visualizar, comprobará si existe llamando al método **buscarRegistro** (al que le daremos como entrada el idMueble tecleado: si existe el idMueble, nos devolverá un objeto de tipo **Mueble** y si no existe un objeto nulo).
   Si no existe, visualizaremos el mensaje NO EXISTE EL MUEBLE BUSCADO y volveremos a introducir otro idMueble, hasta que lo encuentre o pulse 0 para finalizar.
   Si existe, visualizaremos todos los atributos del Mueble.

4. **Eliminar Mueble: eliminarMueble()**. Se pedirá por teclado el **idMueble** del mueble que se desee eliminar, comprobará si existe llamando al método **existeRegistro** (que devolverá true si idMueble existe y false si no existe).
   Si no existe, visualizaremos el mensaje: NO EXISTE EL MUEBLE QUE QUIERES ELIMINAR y volveremos a introducir otro idMueble, hasta que lo encuentre o pulse 0 para finalizar.
   Si existe, preguntaremos si realmente deseamos eliminarlo(DESEA ELIMINAR EL MUEBLE (S/N)) y si se teclea que S, se elimina y si se teclea N, vuelve a introducir otro idVino.

   Para eliminar un mueble, utilizaremos un fichero temporal, llamado TEMPORAL.TMP, el cual lo abriremos para grabar todos los datos del fichero de MUEBLES.DAT menos el que queremos elimiar.
   Iremos leyendo del fichero de objetos MUEBLES.DAT, cada objeto leído que no sea el objeto a eliminar, lo iremos grabando en el fichero temporal.
   Cuando encontremos el objeto a eliminar **no lo grabaremos** en el fichero temporal.
   Cuando se acabe el fichero MUEBLES.DAT, cerraremos ambos ficheros.
   Finalmente tendremos que renombrar el fichero TEMPORAL.DAT y darle el nuevo nombre MUEBLES.DAT

5. **Eliminar todos los datos del fichero: eliminarFichero():** Si existe eliminará el fichero MUEBLES.DAT del disco y si no existe visualizará un mensaje diciendo que no existe.

6. **Salir de la aplicación**.

Tienes que controlar todas las excepciones que consideres oportunas.
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.

CRITERIOS DE PUNTUACIÓN. TOTAL 10 PUNTOS.

Clase Utilidades	1 puntos
Clase Mueble	0,5 puntos
AppGestiónMuebles: Menú	0,5 puntos
AppGestiónMuebles: Añadir Mueble	3 puntos
AppGestiónMuebles: Listar Mueble	1 puntos
AppGestiónMuebles: Buscar Mueble	1 puntos
AppGestiónMuebles: Eliminar Mueble	1,5 puntos
AppGestiónMuebles: Borrar fichero	0,5 puntos
Aspectos generales (Presentación del informe, buen diseño proyecto, bien estructurado, comentarios, etc)	1 puntos

RECURSOS NECESARIOS PARA REALIZAR LA TAREA.

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

CONSEJOS Y 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.
- Ejercicios de apoyo dejados en el aula virtual de la unidad PROG06 en Recuros Aportados por el profesorla:
    - Ejercicios Resueltos de Ficheros
    - Ejercicio Resuelto de Ficheros de Objetos: Artículos
    - Ejercicio Resuelto de Ficheros de Objetos de Personas con Menú

INDICACIONES DE ENTREGA.

- Lo que debes entregar:
    - Un proyecto en **NetBeans** llamado: **Tarea_Prog06_Apellido1Apellido2Nombre**
    - Además del proyecto crearemos un documento **PDF** llamado: **Tarea_Prog06_ 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 de las diferentes opciones del ejercicio).
- Comprimir el proyecto NetBeans y el documento pdf creados en un fichero llamado: **Tarea_Prog06_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_Prog06_Apellido1Apellido2Nombre_ENVIO2.zip**