

PAC DESARROLLO

CFGS Desarrollo de Aplicaciones

Módulo 03B: Programación



2S 2022/2023

INFORMACIÓN IMPORTANTE

Para la correcta realización de la PAC el alumno deberá consultar los contenidos recogidos en el **Tema 1, Tema 2, Tema 3 y Tema 4** del material didáctico.

Requisitos que deben cumplirse en vuestros trabajos:

- Siempre que utilicéis información de Internet para responder / resolver alguna pregunta, tenéis que citar la fuente (la página web) de dónde habéis sacado aquella información.
- No se aceptarán respuestas sacadas de Internet utilizando la metodología de copiar y pegar. Podéis utilizar Internet para localizar información, pero el redactado de las respuestas ha de ser vuestro.
- Las respuestas a las preguntas deben estar bien argumentadas, no se admiten respuestas escuetas o monosílabas.
- La PAC debe entregarse en **formato ZIP**.
- Este ZIP, contendrá el proyecto **realizado en Java**.
- **Deberá entregarse únicamente la carpeta src comprimida con todo el código en su interior sin utilizar ningún package.**
- En el caso de **no** realizarse la entrega en dicho formato **el alumno se hace responsable** de posibles incompatibilidades en la visualización de su entrega y por ende afectará a su calificación.

CRITERIOS DE CORRECCIÓN

1. Todos los programas realizados en la PAC deben realizarse con IDE con que se pueda trabajar con el lenguaje Java
2. Para la realización de esta PAC es necesario que se utilicen las estructuras de control y las estructuras repetitivas siempre que sea posible.
3. Se deben poner comentarios para su mejor comprensión. Estos comentarios explicarán la funcionalidad del código. Se valorarán los comentarios en la parte de presentación.
4. Si se detecta la entrega de una PAC copiada, ya sea de una fuente externa o con un contenido idéntico al de otro alumno/a, serán evaluadas como suspenso, con una calificación de 0.

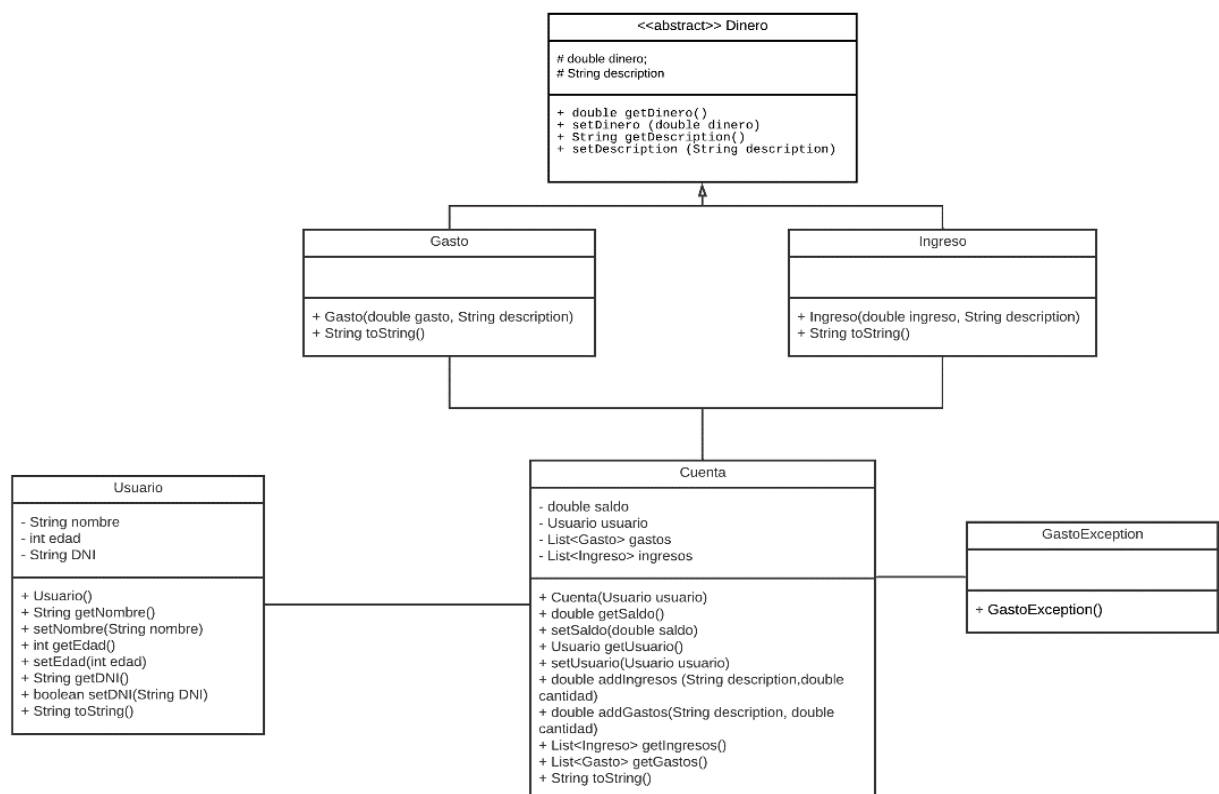
Introducción:

En esta práctica se evaluarán vuestros conocimientos en programación orientada a objetos. Leed bien el enunciado ya que debéis crear las clases y las funciones con los nombres que se indican en este documento.

La práctica consiste en crear una aplicación de gestión de gastos personales. A través de un menú interactivo por la consola, introducir ingresos y gastos para así poder llevar un pequeño control de nuestra economía.

El siguiente **diagrama de clases** representa la estructura que **debéis realizar (OBLIGATORIO SEGUIR LA ESPECIFICACIONES)**:

DIAGRAMA DE CLASES
PAC Desarrollo M03



Importante: Recordad que es obligatorio que **todas las funciones y métodos tengan el mismo nombre y parámetros** que se indican en el diagrama de clases anterior. Estas clases no se pueden añadir ni más métodos ni más atributos, ya que no se puede modificar el diagrama de clases.

Clase Usuario

Esta clase será la encargada de gestionar un único usuario. Éste se creará al inicio del programa leyendo datos por el teclado.

Ejemplo de datos correctos:
 Nombre: Alberto
 Edad: 23

El DNI deberá tener un formato concreto, está **comprobación la realizará en la función setter**, la cual devolverá un booleano conforme es correcto o no. Si el DNI es correcto quedará asignado. Es obligatorio utilizar **una expresión regular** para la comprobación del DNI.

Formato correcto:

- ✓ Los primeros 8 caracteres solo podrán ser numéricos.
- ✓ El ultimo caracteres deberá ser una letra MAYÚSCULA entre la A y la Z.
- ✓ El guion entre los números y la letra es opcional, **admitiendo ambas posibilidades.**
 - ✓ DNI: 78844112L
 - ✓ DNI: 78844112-L

Tendrá una función toString con la que devolver su contenido.

En la clase Usuario no está permitido el uso de ningún import.

Clases Gasto e Ingreso:

Las clases Gasto e Ingreso heredarán de Dinero y tendrán un único constructor en el que se inicializarán los valores recibidos por parámetros.

Además, tendrán una función toString con la que devolver su contenido.

En la clase Gasto e Ingreso no está permitido el uso de ningún import.

Clase Cuenta:

Clase donde se gestionarán todos los movimientos de dinero tanto ingresos como gastos.

Inicialmente (en el constructor) se recibirá el usuario que es dueño de la cuenta y el saldo inicial será de 0€.

Al añadir un nuevo ingreso se sumará al saldo de la cuenta teniendo en esta variable nuestro dinero real, la función devolverá el saldo de la cuenta.

Al añadir un nuevo gasto se debe comprobar si se dispone de saldo suficiente, en caso contrario se deberá lanzar una nueva excepción del tipo `GastoException()`, pero el programa no debe finalizar. Si se dispone de saldo suficiente se restará el importe del gasto y se devolverá el saldo de la cuenta.

Las funciones `getGastos` y `getIngresos` nos devolverán **todos** los movimientos de un tipo u otro.

La clase `Cuenta` tendrá una función `toString` con la que devolverá el usuario y su saldo.

En la clase `Cuenta`, solo está permitido importar la clase `List`; y la clase que hayamos elegido para esa `List`. No está permitido el uso de ningún otro import.

Main:

La clase Main será la que se ejecute al iniciar el programa, se valorará la organización y comprensibilidad del código; tendrá seguirá unos pasos definidos:

1. Creación del usuario y sus datos, el DNI no se establecerá hasta que se introduzca uno correcto, el orden de los datos será:
 - a. Nombre
 - b. Edad
 - c. DNI
2. Creación de la cuenta
3. Visualización del menú con las instrucciones tal y como se muestra en la siguiente figura (NO se permite notación “” ”” ””):

```
Realiza una nueva acción
1 Introduce un nuevo gasto
2 Introduce un nuevo ingreso
3 Mostrar gastos
4 Mostrar ingresos
5 Mostrar saldo
0 Salir
```

4. Cada acción realizará una operación donde se deberán de solicitar los datos si los necesitase.
5. Los Valores Double con coma en lugar de punto para parte decimal; siguiendo la notación europea: **1000,35€** (para mostrarlos por consola). No es posible hacer import, en ninguna clase.
6. Al finalizar la aplicación se deberá mostrar el mensaje (**importante que sea igual al que se indica**):

Fin del programa.

Gracias por utilizar la aplicación de M03B en el curso 2s2223.

- ✓ En la clase Main, solo se puede importar la clase Scanner; no está permitido el uso de ningún otro import.
- ✓ En la clase Main, se recomienda generar métodos propios, para mejorar la comprensibilidad y la organización del código; pero NO está permitido crear métodos que ya existan en el diagrama de clases; lo que conllevaría una penalización.


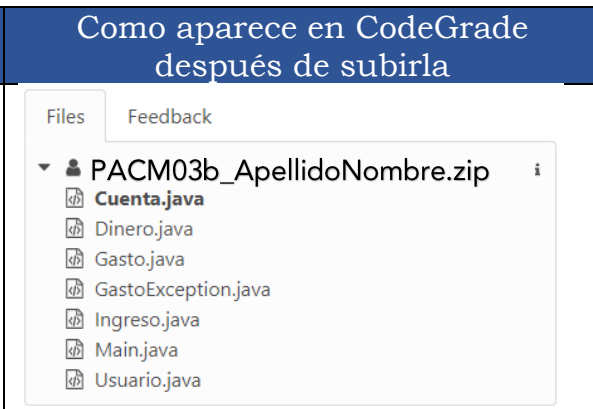
Especificaciones de entrega:

Subir fichero PACM03b_ApellidoNombreOnliner.ZIP

La entrega tiene que mantener estructura indicada en el enunciado: src/ (todas las clases)

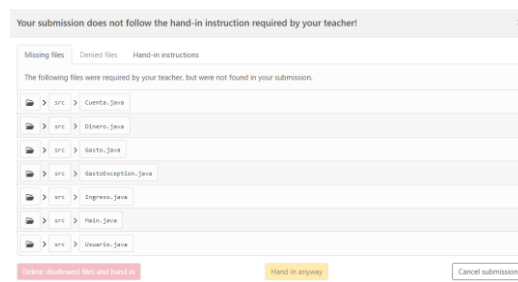
No incluir otras carpetas ni archivos, ya que podría penalizar, incluso llegar a suspender la PAC, si no se pueden pasar las pruebas unitarias.

El contenido de la carpeta *.ZIP debe ser el siguiente:

Contenido carpeta *.zip	Como aparece en CodeGrade después de subirla
	

A LA HORA DE ENTREGAR LA TAREA:

En caso de que los archivos no sean exactamente esos, te saldrá este mensaje:



Te está diciendo que no estás entregando los archivos solicitados; revisa la videotutoría de la pac de Desarrollo, ahí se explica que es lo que hay que entregar, y como solucionarlo.

ÁNIMO CON LA PAC DE DESARROLLO

- No lo dejes para el último día por los posibles problemas que puedan surgir.
- Las fechas son fijas y no se pueden modificar por normativa.
- Solo se aceptarán entregas, realizadas en el lugar habilitado para ello. No se aceptarán entregas ni por mensaje ni por comentarios en la tarea.

