Práctica 2 – Primera parte - POO en Java

Objetivo:
Definir clases para representar objetos del mundo real. Concepto de clase (estado –v.i.s - y comportamiento -métodos). Instanciación. Envío de mensajes.

NOTA: Trabajar sobre la carpeta "tema3" del proyecto

1- A- Definir una clase para representar triángulos. Un triángulo se caracteriza por el tamaño de sus 3 lados (double), el color de relleno (String) y el color de línea (String). El triángulo debe saber:

- Devolver/modificar el valor de cada uno de sus atributos (métodos *get#* y *set#*)
- Calcular el área y devolverla (método *calcularArea*)
- Calcular el perímetro y devolverlo (método *calcularPerimetro*)

NOTA: Calcular el área con la fórmula $Área = \sqrt{s(s-a)(s-b)(s-c)}$, donde a,b y c son los lados y $s = \frac{a+b+c}{2}$. La función raíz cuadrada es Math.sqrt(#)

B- Realizar un programa principal que instancie un triángulo, le cargue información leída desde teclado e informe en consola el perímetro y el área.

2- A – Definir una clase para representar balanzas comerciales (para ser utilizadas en verdulerías, carnicerías, etc). Una balanza comercial sólo mantiene el monto y la cantidad de items correspondientes a la compra actual (es decir, no almacena los ítems de la compra). La balanza debe responder a los siguientes mensajes:

- iniciarCompra(): inicializa el monto y cantidad de ítems de la compra actual.
- registrarProducto(*pesoEnKg*, *precioPorKg*): recibe el peso en kg del ítem comprado y su precio por kg, debiendo realizar las actualizaciones en el estado de la balanza.
- devolverMontoAPagar(): retorna el monto de la compra actual.
- devolverResumenDeCompra(): retorna un String del siguiente estilo "Total a pagar **X** por la compra de **Y** productos" , donde **X** es el monto e **Y** es la cantidad de ítems de la compra.

B - Genere un programa principal que cree una balanza e inicie una compra. Lea información desde teclado correspondiente a los ítems comprados (peso en kg y precio por kg) hasta que se ingresa uno con peso 0. Registre cada producto en la balanza. Al finalizar, informe el resumen de la compra.

3- A- Definir una clase para representar *entrenadores* de un club de fútbol. Un *entrenador* se caracteriza por su nombre, sueldo básico y la cantidad de campeonatos ganados.

- Defina métodos para obtener/modificar el valor de cada atributo.
- Defina el método *calcularSueldoACobrar* que calcula y devuelve el sueldo a cobrar por el *entrenador*. El sueldo se compone del sueldo básico, al cual se le adiciona un plus por campeonatos ganados (5000$ si ha ganado entre 1 y 4 campeonatos; $30.000 si ha ganado entre 5 y 10 campeonatos; 50.000$ si ha ganado más de 10 campeonatos).

B- Realizar un programa principal que instancie un *entrenador,* cargándole datos leídos desde teclado. Pruebe el correcto funcionamiento de cada método implementado.

4-A- Generar una clase para representar círculos. Los círculos se caracterizan por su radio (double), el color de relleno (String) y el color de línea (String). El círculo debe saber:

- Devolver/modificar el valor de cada uno de sus atributos (get# y set#)
- Calcular el área y devolverla. (método *calcularArea*)
- Calcular el perímetro y devolverlo. (método *calcularPerimetro*)

NOTA: la constante PI es *Math.PI*

B- Realizar un programa principal que instancie un círculo, le cargue información leída de teclado e informe en consola el perímetro y el área.

5-A- Modifique el ejercicio 2-A. Ahora la balanza debe poder generar un resumen de compra más completo. Para eso agregue a la balanza la característica *resumen* (String). Modifique los métodos:

- iniciarCompra para que además inicie el *resumen* en el String vacío.
- registrarProducto para que reciba un objeto Producto (que se caracteriza por peso en kg y descripción) y su precio por kg. La operación debe realizar las actualizaciones en monto /cantidad de ítems y adicionar al *resumen* (string) la descripción y el monto pagado por este producto.
- devolverResumenDeCompra() para que retorne un String del siguiente estilo: "Naranja 100 pesos – Banana 40 pesos – Lechuga 50 pesos – Total a pagar 190 pesos por la compra de 3 productos" . La sección subrayada es el contenido de *resumen*.

Realice las modificaciones necesarias en el programa principal solicitado en 2-B para corroborar el funcionamiento de la balanza.

NOTA: dispone en la carpeta tema 3 de la clase Producto ya implementada. Para adicionar la información del producto recibido al *resumen* use concatenación de Strings (operación +).

Práctica 2 – Segunda parte - POO en JAVA

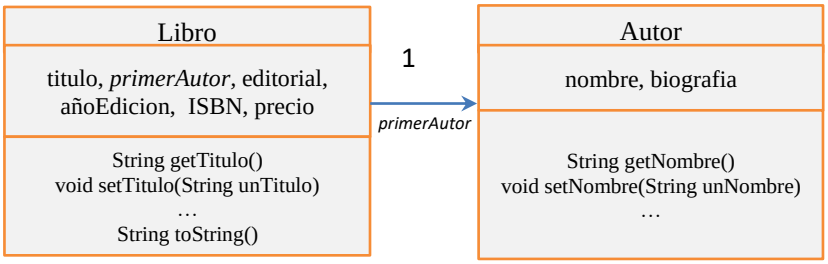
Objetivo:
Continuar trabajando los conceptos de la POO. Inicializar objetos a partir de sus constructores.

NOTA: Las modificaciones planteadas en los ejercicios 1, 2 y 6 deben realizarse sobre los archivos .java de las clases definidas en carpeta "tema 3". Para el resto de los ejercicios, trabajar sobre la carpeta "tema4" del proyecto.

1-A– Defina constructores para la clase Triángulo (definida en la Act. 3): el *primer constructor* debe recibir un valor para cada lado y para los colores de línea y relleno; el *segundo constructor* no debe poseer parámetros ni código (constructor nulo).
B- Realice un programa que instancie un triángulo mediante los distintos constructores.

2-A– Defina un constructor para la clase Entrenador (definida en la Act. 3) que reciba los datos necesarios (nombre, sueldo básico, cantidad de campeonatos ganados). Además defina un constructor nulo.
B- Realice un programa que instancie un entrenador mediante el primer constructor.

3-A- Modifique la clase Libro (carpeta tema 4) para ahora considerar que el primer autor es un objeto instancia de la clase Autor. Implemente la clase Autor, considerando que éstos se caracterizan por nombre y biografía. El autor debe poder devolver/modificar el valor de sus atributos.



B- Modifique el programa ppal. (carpeta tema 4) para instanciar un libro con su autor, considerando las modificaciones realizadas en A). Los datos se ingresan por teclado.

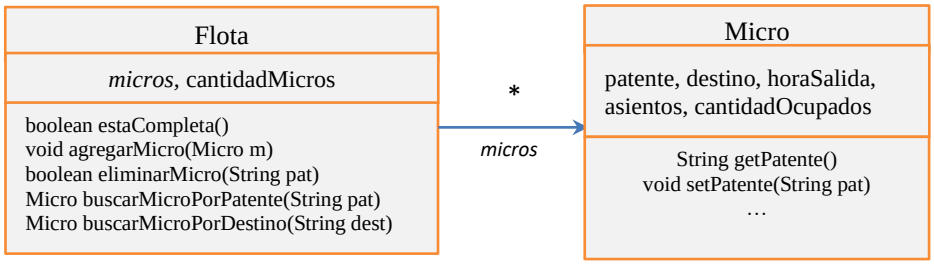
4-A- Definir una clase para representar micros. Un micro conoce su patente, destino, hora salida, el estado de sus 20 asientos (es decir si está o no ocupado) y la cantidad de asientos ocupados al momento. Lea detenidamente a) y b) y luego implemente.

a) Implemente un constructor que permita iniciar el micro con una patente, un destino y una hora de salida (recibidas por parámetro) y sin pasajeros.
b) Implemente métodos para:
   i. devolver/modificar patente, destino y hora de salida
   ii. devolver la cantidad de asientos ocupados
   iii. devolver si el micro está lleno
   iv. validar un número de asiento recibido como parámetro (es decir, devolver si está en rango o no)
   v. devolver el estado de un nro. de asiento válido recibido como parámetro
   vi. ocupar un nro. de asiento válido recibido como parámetro
   vii. liberar un nro. de asiento válido recibido como parámetro
   viii. devolver el nro. del primer asiento libre

B- Realice un programa que cree un micro con patente "ABC123", destino "Mar del Plata" y hora de salida 5:00. Cargue pasajeros al micro de la siguiente manera. Leer nros. de asientos desde teclado que corresponden a pedidos de personas. La lectura finaliza cuando se ingresa el nro. de asiento -1 o cuando se llenó el micro. Para cada nro. de asiento leído debe: validar el nro; en caso que esté libre, ocuparlo e informar a la persona el éxito de la operación; en caso que esté ocupado informar a la persona la situación y mostrar el nro. del primer asiento libre. Al finalizar, informe la cantidad de asientos ocupados del micro.

5-A- Definir una clase para representar flotas de micros. Una flota se caracteriza por conocer a los micros que la componen (a lo sumo 15). Defina un constructor para crear una flota vacía (sin micros).
Implemente métodos para:

i. devolver si la flota está completa (es decir, si tiene 15 micros o no).
ii. agregar a la flota un micro recibido como parámetro.
iii. eliminar de la flota el micro con patente igual a una recibida como parámetro, y retornar si la operación fue exitosa.
iv. buscar en la flota un micro con patente igual a una recibida como parámetro y retornarlo (en caso de no existir dicho micro, retornar null).
v. buscar en la flota un micro con destino igual a uno recibido como parámetro y retornarlo (en caso de no existir dicho micro, retornar null).



B- Genere un programa que cree una flota vacía. Cargue micros (sin pasajeros) a la flota con información leída desde teclado (hasta que se ingresa la patente "ZZZ000" o hasta completar la flota). Luego lea una patente y elimine de la flota el micro con esa patente; busque el micro con dicha patente para comprobar que ya no está en la flota. Para finalizar, lea un destino e informe la patente del micro que va a dicho destino.

6-A– Defina constructores para la clase Círculo (definida en la Act. 3): el *primer constructor* debe recibir un valor para el radio y para los colores de línea y relleno; el *segundo constructor* no debe poseer parámetros ni código (constructor nulo).

B- Realice un programa que instancie un círculo mediante los distintos constructores.