TRABAJO PRÁCTICO Nº3

Resolvé los siguientes ejercicios

1. Crear una clase llamada Persona. La misma estará compuesta por:

· Atributos privados: nombre (String), apellido (String), direccion (String).

· Métodos públicos: setters y getters para cada atributo, y el método toString().

Codificar la función main() en una clase denominada Test, creando dos objetos Persona p1, Persona p2, asignando valores en sus atributos mediante los métodos setters.

A continuación, deberás mostrar por pantalla los valores de los atributos de cada objeto, utilizando los métodos getters. A su vez, imprimí por pantalla sus atributos utilizando el método toString().

Obs: El método toString debe ser igual al siguiente ejemplo:

```
@Override
public String toString() {
	return "[Nombre:" + nombre + ", Apellido:" + apellido + ", Dirección: " + direccion ]";
}
```

Por ejemplo, si mostramos por pantalla la llamada al método toString, por ejemplo, será:

persona1 [Nombre:Diego, Apellido:Diaz, Direccion: Roosevelt 3313]

persona2 [Nombre:Pablo, Apellido:Gomez, Direccion: Virrey del pino 2284]

2. Crear una clase llamada TarjetaDeCredito.

· Atributos privados: numero (String), titular (String), limiteDeCompra (double), disponible (double)

· Métodos públicos: setters y getters para cada atributo, y el método toString().

· Constructor por parámetros que asigne número, titular y limiteDeCompra. El disponible, en un inicio, será igual al límite.

Crear los métodos:

- puedoComprar, método privado que según un monto devuelve true si puede hacer la compra, o false en caso contrario.

- realizarCompra, método público que dado un monto devuelve un booleano si lo pudo realizar, actualizando los atributos que corresponda.

- actualizarLimite, método público que recibe solamente el nuevo límite de compra y actualiza a la vez el límite y el disponible según este criterio: Si el límite de compra nuevo es menor al anterior debe decrementar el disponible.

Veamos algunos ejemplos: imagínate que tu límite anterior era de $5000, y ya habías realizado una compra por $500. En ese entonces, tu disponible era de $4500. Ahora, te dicen que tu nuevo limite de compra es de $3000, por lo cual, tu disponible actual pasa a ser de $2500 (resulta de hacer 3000 – (5000-4500)).
Por otro lado, si el nuevo límite de compra es menor a lo que ya habíamos gastado, nuestro nuevo disponible será 0, ya que gastaste más de lo permitido.

En la clase Test, se creará un objeto tarjeta, con los valores correspondientes, y se deberán probar los métodos desarrollados con la siguiente información:

- Numero 4145414122221111
- Titular Juan Perez
- Limite 10000

Hacer una compra de $4000 → Ver la información llamando al método toString. → Chequear que el disponible debería ser de $6000. → Bajar el límite a $3000 → Ver la información llamando nuevamente al método toString. → Chequear que el disponible debería ser $0.

3. Crear una clase Superhéroe con los siguientes atributos: nombre (String), fuerza (int), resistencia (int), superpoderes (int)

Tendrá:

· Constructor que asigne todos los datos por parámetros

· Métodos públicos: setters y getters para cada atributo, toString() y jugar.

El método jugar recibirá un superhéroe por parámetro, y devolverá 1 si ganó, 2 si perdió o 3 si empató. Para triunfar un superhéroe debe ganar en al menos 2 de los 3 ítems.

Realizar una clase Test que contenga el main para probar el correcto funcionamiento de los métodos de la clase previamente realizada con el siguiente ejemplo.

superHeroe1: Nombre: Batman, Fuerza: 90, Resistencia: 70, Superpoderes: 0

superHeroe2: Nombre: Superman, Fuerza: 95, Resistencia: 60, Superpoderes: 70

Hacer jugar al superheroe2 contra el superheroe1 y mostrar el resultado por pantalla. Chequear que el resultado debería ser 2(Perdió).

Hacer jugar al superheroe1 contra el superheroe2 y mostrar el resultado por pantalla. Chequear que el resultado debería ser 1(ganó)

Crear más superhéroes con distintos valores y jugar.

4. Crear una clase Cafetera con los siguientes atributos: capacidadMaxima (la cantidad máxima de café que puede contener la cafetera), cantidadActual (la cantidad actual de café que hay en la cafetera).

Implementar, al menos, los siguientes métodos:

a) Constructor predeterminado: establece la capacidad máxima en 1000 y la actual en cero (cafetera vacía).

b) Constructor con la capacidad máxima de la cafetera: inicializa la cantidad actual de café igual a la capacidad máxima.

c) Constructor con la capacidad máxima y la cantidad actual. Si la cantidad actual es mayor que la capacidad máxima de la cafetera, la ajustará al máximo.

d) Setters y Getters.

e) llenarCafetera(): la cantidad actual de la cafetera será igual a la capacidad máxima.

f) servirTaza(int): simula la acción de servir una taza con la capacidad indicada por parámetro. Si la cantidad actual de café “no alcanza” para llenar la taza, se sirve lo que quede.

g) vaciarCafetera(): setea la cantidad de café actual en cero.

h) agregarCafe(int): añade a la cafetera la cantidad de café indicada, en el caso de ser posible.

Realizar una clase Test que contenga el main para probar el correcto funcionamiento de los métodos de la clase previamente realizada.

5. Clase una clase Hotel con los siguientes atributos: nombre (String), localidad (String), habitacionesTotales (int), habitacionesDisponibles (int).

Implementar los siguientes métodos y constructores:

a) Constructores: vacío y parametrizado completo

b) ocuparHabitaciones: recibe por parámetro la cantidad de habitaciones a ocupar. Valida que sea posible realizar la acción y devuelve true o false según corresponda.

c) desocuparHabitaciones: recibe por parámetro la cantidad de habitaciones a desocupar. Valida que sea posible realizar la acción y devuelve true o false según corresponda.

d) getters, setters y toString

Crear el siguiente objeto en la clase Test y realizar las acciones correspondientes:

"Hilton", "CABA", 200, 15.

Ocupar 18 habitaciones → chequear que no se pueda realizar dicha acción. → ocupar 3 habitaciones → chequear que resten 12 habitaciones disponibles. → desocupar 10 habitaciones → chequear que resten 22 habitaciones disponibles.

6. Crear una clase Vuelo con los siguientes atributos: origen (String), destino (String), fecha (String), numero (int), capacidadTotal (int), asientosOcupados (int).

Implementar los siguientes métodos y constructores:

a) Constructor por parámetros con origen, destino, fecha, número, capacidadTotal.

b) Método público reservar: recibe la cantidad de personas que quieren reservar asientos. De ser posible, actualiza la cantidad de asientos ocupados. Devuelve true o false.

c) Método público liberarAsientos: recibe la cantidad de asientos que se desea liberar. De ser posible, actualiza la cantidad de asientos ocupados. Devuelve true o false.

c) getters, setters y toString

Crear el objeto, mediante el contructor, en la clase Test con la siguiente informacion "Aeroparque","Miami","01/06/2018",1234, 250.

Asientos ocupados:15 (con setter)

Probar el método reservar con distintos casos para que devuelva true o false.

Idem con el método liberarAsientos.

7. Crear una clase Automovil con los siguientes atributos: marca (String), modelo (String), patente (String), capacidadTotalCombustible (double), cantidadCombustible (double), kmPorLitro(double. representa cuantos kilometros recorre con un litro de combustible).

Implementar los siguientes métodos y constructores:

a) Constructor por parámetros: recibe marca, modelo y patente.

b) Método público viajar: recibe la cantidad de kilómetros. Actualiza la cantidad de Nafta, devuelve true o false.

c) Método privado verificarCantidadCombustible:recibe por parámetro la cantidad que se quiere verificar. Chequea si se puede o no cargar al tanque la cantidad deseada. Retorna true o false según corresponda.

d) Método público cargarCombustible: recibe por parámetro la cantidad que se quiere cargar. Invoca al método verificarCantidad para ver si puede cargar dicha cantidad. Si puede, actualiza el atributo correspondiente. Devuelve true o false,

e) getters, setters y toString

En la clase test, crear el objeto a través del constructor. "Ford","Fiesta", "ABCD123", Capacidad:40, Cantidad: 20, Kilometros por litro: 10.

Probar todos los métodos

8. Crear una clase Gato y una clase Raton Ambos tienen como atributo energía (int).

Un gato puede correr mientras le de la energía, descontando un punto de energía por metro corrido. Lo mismo el ratón pero se le descuentan 2 puntos de energía por cada metro corrido. Es decir, un gato o un ratón corren x cantidad de metros según la energía.

Implementar un método en Gato que se llame alcanzar y reciba como parámetro un ratón y la distancia que debe recorrer para alcanzarlo, si lo alcanza devuelve true, caso contrario false.

Ejemplo, un gato con 100 de energía puede correr 100 metros, un ratón con 100 de energía puede correr 50 metros. O sea que si el ratón está a más de 50 metros del gato y ambos tienen 100 de energía, no lo va a alcanzar.

Crear un gato y un ratón en una clase Test y jugar