Práctica 4. Diseñar clases para modelar el estado y comportamiento aplicando encapsulamiento

Objetivo

El alumno diseñara clases y métodos para modelar la inicialización del estado y comportamiento de entidades del mundo real, encapsulando los atributos y métodos, precisando el protocolo con pase de parámetros y la asignación de valores, con una actitud crítica, propositiva y creativa.

🛠 Equipo y material de apoyo:

- Computadora Personal (PC)
- Ambiente de desarrollo o IDE (VSCODE)
- JDK Java Development Kit

DESARROLLO DE LA PRACTICA:

Diseñar el diagrama de clases e implementarlo en un programa de Java considerando lo siguiente:

- Modelar la estructura de clases usando paradigma orientado a objetos.
- Diseñar los métodos accesadores y mutadores (precisar el protocolo de los métodos con pase de parametros y asignación de valores)
- Codifica los métodos correspondientes para la implementación del comportamiento de los ejercicios propuestos (utilizar las capacidades del lenguaje y el manejo adecuado de estructuras de selección y control).
- Compila, ejecuta y prueba con diferentes valores para validar el programa que realizaste.

Programa 1: Creando la clase CellPhone

Wireless Soluciones, Inc., es una empresa que vende teléfonos celulares y servicios inalámbricos. Usted es un programador del departamento de tecnología de la información (TI) de la empresa y su equipo está diseñando un programa para administrar todos los teléfonos celulares que están en inventario. Se le ha pedido que diseñe una clase que represente un teléfono celular. Cree la clase CellPhone con sus respectivos atributos y métodos aplicando encapsulamiento de datos. Además, diseñe el diagrama UML de la clase.

La clase CellPhone será utilizada por varios programas que su equipo está desarrollando. Para realizar una prueba simple de la clase, escriba el programa respectivo solicitando al usuario los elementos correspondientes.

Programa 2: Diseñe e implemente un objeto de libreta de direcciones que contenga el nombre, la dirección, número de teléfono de una persona, la dirección y el número de teléfono de su trabajo y los números de su máquina de fax, teléfono celular y buscapersonas. Escriba un plan de prueba para el objeto e implemente una clase que lo pruebe. Diseñe el diagrama UML de la clase, aplique encapsulamiento e imprime los datos personales y del trabajo de tal manera que se vea presentable.

Programa 3: Cree una clase llamada **Factura** que una Ferretería podría usar para representar una factura por un artículo vendido en la tienda. Una factura debe incluir cuatro piezas de información como **variables de instancia**: un número de pieza (tipo String), una descripción de la pieza (tipo String), la cantidad de artículos de ese tipo que se van a comprar (tipo int) y el precio por articulo (tipo double). Proporcione un método obtener y un método establecer para cada variable de instancia. Además, proporcione un método llamado **obtenerMontoFactura** que calcule el monto de la factura (es decir, multiplique la cantidad por el precio por articulo) y luego devuelva el monto como un valor doble. Si la cantidad no es positiva, debe establecerse en 0. Si el precio por articulo no es positivo, debe establecerse en 0.0. Escriba una clase de prueba llamada **FacturaPrueba** que demuestre las capacidades de la clase Factura.

Programa 4: Cree una clase llamada Empleado, que incluya tres variables de instancia: un primer nombre (tipo String), un apellido paterno (tipo String) y un salario mensual (double). Su clase debe tener un constructor que inicialice las tres variables de instancia. Proporcione un método *establecer* y un método *obtener* para cada variable de instancia. Si el salario mensual no es positivo, no establezca su valor. Escriba una aplicación de prueba llamada PruebaEmpleado, que demuestre las capacidades de la clase Empleado. Cree dos objetos Empleado y muestre el salario *anual* de cada objeto. Después, proporcione a cada Empleado un aumento del 10% y muestre el salario anual de cada Empleado otra vez.

Adjunte el diagrama de clase y el código fuente, también deberá hacer una demostración de su ejecución en la clase de laboratorio.

Entrega:

- La entrega se realizará mediante una actividad en BB. Se han de entregar los archivos fuentes desarrollados, más el reporte de la practica adjuntos adjuntos en un fichero zip cuyo nombre esté formado por el número de práctica y nombre del alumno. Ejemplo Practica4_JuanLopez.pdf
- Hacer una clase por cada ejercicio y adjuntarlos
- Entregar un reporte de práctica, formato reporte técnico y electrónico, podrá utilizar este formato y agregar el procedimiento correspondiente), que conceptos fundamentales se aplicaron en esta practica, junto con resultados y conclusiones.

Análisis de Resultados

Conclusiones

Referencias

Criterios para evaluar:

Código (70 puntos)

- Debe incluir comentarios en su programa, sin llegar al uso exagerado de ellos.
- Debe utilizar sangrías en los lugares apropiados.
- Poner como encabezado del programa en comentarios, titulo, la descripción de la práctica, nombre del alumno que realizo la práctica.
- Funcionamiento correcto del programa

Reporte (30 puntos):

- Portada, competencia u objetivo 2.5 puntos
- Pegar Código del programa (anexar archivos .java) 2.5 puntos
- Explicación completa del código y su funcionamiento 15 puntos
- Resultados (análisis de resultados) 5 puntos
- Conclusiones: 5 puntos

Nota: No se aceptarán trabajos fuera de la fecha de entrega o que no cumplan con las especificaciones de entrega requeridas. Los trabajos que se detecten copiados, serán anulados automáticamente para ambos alumnos.

El reporte debe contener lo anterior mencionado y quitar las instrucciones, el reporte si es leido por cualquier persona, esta debe ser capaz de repetir todo el proceso sin errores (por lo que debes de incluir toda la información).

La conclusión no debe ser describir el proceso que se siguio, investigar que debe contener una conclusión.