Universidad Mariano Gálvez
Ingeniería en Sistemas de Información
Ing. Christian López
Programación 2

Proyecto 2 Sistema de Ventas

Usted ha sido contratado para desarrollar una aplicación para la empresa Auto Partes, esta empresa se dedica a la venta de repuestos de todo tipo de vehículo automotor, además cuenta con el servicio de envío lo cual le ha servido como estrategia de negocio para incrementar a sus clientes, actualmente ellos cuentan con su sistema en consola pero quieren migrar a una aplicación Desktop, por lo que usted será el encargado de realizar esta labor.

Requerimientos funcionales:

1. Debe crear un mecanismo de seguridad para validar usuarios permitidos.
2. La pantalla principal debe tener un menu en la parte superior para ingresar a cada opción.
3. Altas, bajas y cambios de cualquiera de los dos tipos de clientes en las mismas pantallas
4. Altas, bajas y cambios de productos
5. Altas, bajas y cambios para Ordenes de Compra
6. Reporte en un Grid donde muestre todos los clientes disponibles, permitiendo la opción que se muestren solo los clientes individuales, empresas o ambos.
7. Reporte en un Grid donde muestre todos los productos disponibles.
8. Reporte en un Grid donde se ingrese un id de orden de compra y muestre los detalles del cliente, productos, total y subtotales de la orden.

Requerimientos técnicos:

1. **clase Cliente**: Deberá contener las propiedades que usted crea conveniente, estas propiedades deben ser privadas.
    a. Esta clase está obligada a mantener una propiedad para identificar a cada cliente, la cual será de tipo "int" y se deberá llamar id y adicionalmente manejará una propiedad "static" llamada sigIdCliente que se inicializará con el valor 1, la cual será utilizada para asignar el id de cada cliente y será incrementado en 1 cada vez que se agrega un nuevo cliente.
    b. Esta clase debe tener 2 constructores, el primero sin parámetros que implementará una asignación Default, es decir, que deberá asignarle el valor que le corresponda a la propiedad id y esa será su única implementación. El segundo constructor recibirá todos los parámetros que correspondan a cada propiedad definida para esta clase y asignará estos valores a las propiedades así como también el id que le corresponda, pero como la asignación del id ya fue implementada en el primer constructor, para reutilizar este código lo que hará es únicamente su invocación desde el segundo constructor.
    c. Los métodos de esta clase serán los Accessors para cada propiedad, excepto para la propiedad sigIdCliente que no tendrá accessors implementados, además deberá sobreescribir el método toString sin parámetros que retornará el tipo de dato String y que se encargará de retornar la cadena con el nombre y valor de todas sus propiedades y adicionalmente en el inicio encerrado entre corchetes deberá mostrar el nombre de la

clase, pero este nombre no debe ser hard code, deberá mostrarlo de forma dinámica, para eso hará la llamada al método getNombreClase que implementará en otra clase que se llamará Utilerias, este método recibirá un solo parámetro que será la clase y se encargará de retornar un String que mostrará únicamente el nombre de la clase (sin la información del package al que pertenece), un ejemplo de como será la salida del método toString es el siguiente: [Cliente] id=1, nombre=Carlos Martinez, etc.

2. Auto Partes cuenta con clientes individuales (personas) como también empresas, por lo que deberá implementar estas clases, considerando que ambas clases comparten información común que heredan de la super clase Cliente y las subclases que heredarán de la clase Cliente deberá nombrarlas Individual y Empresa.
    a. Las propiedades adicionales para la clase Individual será el DPI de tipo String, y para la clase Empresa será la propiedad contacto de tipo String y descuento de tipo int que representa un porcentaje de descuento que se le aplicarán a los clientes que sean de esta categoría. Se deben implementar los constructores que deberán reutilizar los constructores de su superclase, los accessors de las propiedades y también deberán sobreescribir el método toString para extender la funcionalidad de la clase que heredan y así poder mostrar los valores de las propiedades de su superclase como también adicionar los valores de sus propiedades.
3. **La clase DataSistema** será utilizada para guardar un catálogo de clientes, para lo cual deberá agregar una propiedad que será un arreglo de tipo Cliente, la cual será "public" y "static" y les asignará en la misma línea de definición su espacio de memoria, por default en esta asignación debe crear 4 clientes de tipo Individual y 4 de tipo Empresa con sus respectivos valores para cada uno de los clientes.
4. La **clase Producto**, debe contener las propiedades que usted crea conveniente, estas propiedades deben ser privadas.
    a. Esta clase está obligada a mantener una propiedad para identificar a cada producto, la cual será de tipo "int" y se deberá llamar id y adicionalmente manejará una propiedad "static" llamada sigIdProducto que se inicializará con el valor 2000, la cual será utilizada para asignar el id de cada producto y será incrementado en 1 cada vez que se agrega un nuevo producto.
    b. Esta clase debe tener 2 constructores, el primero sin parámetros que implementará una asignación Default, es decir, que deberá asignarle el valor que le corresponda a la propiedad id y esa será su única implementación. El segundo constructor recibirá todos los parámetros que correspondan a cada propiedad definida para esta clase y asignará estos valores a las propiedades así como también el id que le corresponda, pero como la asignación del id ya fue implementada en el primer constructor, para reutilizar este código únicamente lo invocará desde el segundo constructor.
    c. Los métodos de esta clase serán los Accessors para cada propiedad, excepto para la propiedad sigIdProducto que no tendrá accessors implementados, además deberá sobreescribir el método toString sin parámetros que retornará el tipo de dato String y que se encargará de retornar la cadena con el nombre y valor de todas sus propiedades y adicionalmente en el inicio encerrado entre corchetes deberá mostrar el nombre de la clase, pero este nombre no debe ser hard code, deberá mostrarlo de forma dinámica, para

eso hará la llamada al método getNombreClase, un ejemplo de como será la salida del método toString es el siguiente:
[Producto] id=1, nombre=Silvin, precio unitario=Q250

5. **La clase DataSistema** será utilizada para guardar un catálogo de productos, para lo cual deberá agregar una propiedad que será un arreglo de tipo Producto, la cual será "public" y "static" y le asignará en la misma línea de definición su espacio de memoria, debe agregar 8 productos por default en la creación de la propiedad.
6. Por último deberá implementar la orden de compra para lo cual será necesario lo siguiente:
    a. **La clase Orden**, que representa la orden de compra de algún cliente, por lo que debe tener las siguientes propiedades y tipos: int id, Cliente cliente, ItemOrden item1, ItemOrden item2, Date fechaOrden, double precioEnvio, double total, String tipoEnvio, String estado, static int sigIdOrden (llevar el correlativo del id de la orden), int diasEnvio.
    b. Constructores:
    - Orden(): inicializa el id, el total= 0.0 y fecha = new Date();
    - Orden(Date pFecha); //Deberá llamar al constructor Orden() para aprovechar su implementación
    - Orden(int pCliente, Date pFecha);
    Automáticamente deberá calcular el monto total de la orden, así que generará un método llamado getTotalOrden, se calculará en base a los precios de los productos y el descuento si es que aplica.
7. **Clase ItemOrden**, que representa el cálculo de un producto que forma parte de la orden
    a. tendrá las siguientes propiedades:
    - int noLinea; //que será un número correlativo de la orden.
    - int cantidad;
    - Producto producto;
    b. Constructor:
    - ItemOrden(int pNoLinea, int pCantidad, int pIdProducto);
    c. Método:
    - toString(); //tomando las características de lo que se ha dicho anteriormente de otras clases.
    - getTotalItem(); // = precio * cantidad
8. **La clase DataSistema** será utilizada para guardar las ordenes de compra, para lo cual deberá agregar una propiedad que será un arreglo de tipo Orden, la cual será "public" y "static" y le asignará en la misma línea de definición su espacio de memoria, debe agregar 8 productos por default en la creación de la propiedad.
9. Para probar la aplicación será necesario crear una clase Main que dará inicio al sistema.
10. Cuando sea necesario retornar valores para ser desplegados en pantalla, estos deben llevar el formato de moneda por lo que deberá implementar en la clase Util un método que se encargue de realizar este procedimiento.

Consideraciones técnicas adicionales:

1. Todo funcionará en memoria RAM no debe guardar nada en archivos, ni en base de datos.
2. Está permitido agregar clases adicionales para su implementación.
3. La Interfaz gráfica de usuario debe implementarla con JavaFX, no está permitido utilizar ningún otro medio.
4. Debe generar la documentación técnica generada con Javadoc.
5. Puede utilizar de IDE IntelliJ o Netbeans
6. Debe utilizar git y GitHub (crear su cuenta y repositorio para subir su proyecto).

Entregable

1. Debe grabar un video, subirlo a Youtube y adjuntar el link, donde explique (hablado) la funcionalidad de su proyecto ya que esto sirve como guía para la calificación del mismo.
2. En un archivo .zip debe entregar lo siguiente:
    a. Carátula de integrantes
    b. Link del manual de usuario
    c. Link del repositorio en GitHub
    d. documentación técnica en GitHub
    e. Directorio con archivo ejecutable en GitHub

Información:

1. El proyecto se puede trabajar en pareja o individual
2. Valor del proyecto: 15 puntos
3. Fecha máxima de entrega 2 de octubre a las 23:59 horas
4. Los proyectos serán validados, al detectarse copias tendrán 0 todos los involucrados y se enviará un informe a las autoridades de la Universidad

Universidad Mariano Gálvez
Ingeniería en Sistemas de Información
Ing. Christian López
Programación 2

Ponderación:

Evaluar	Ponderación	Nota
Manual de Usuario	**10**	
Documentación Técnica	**10**	
Clase Cliente, Individual, Empresa, Producto y DataSistema	**10**	
Altas, bajas y cambios de Clientes	**10**	
Altas, bajas y cambios de Productos	**10**	
Crear Orden de Compra	**10**	
Reporte de Clientes	**10**	
Reporte de Productos	**10**	
Reporte de Orden de Compra	**10**	
Uso de GitHub	**10**	
TOTAL	**100**	

Tomar en cuenta lo siguiente:

- La nota del manual de usuario y técnico será proporcional al porcentaje de desarrollo entregado.