

**Universidad Tecnológica Nacional**  
**Facultad Regional Avellaneda**



Técnico Superior en Programación - Técnico Superior en Sistemas Informáticos

**Materia: Laboratorio de Programación II**

Apellido:					Fecha:					
Nombre:					Docente <sup>(2)</sup> :	F. Dávila / M. Cerizza				
División:	2°C				Nota <sup>(2)</sup> :					
Legajo:					Firma <sup>(2)</sup> :					
Instancia <sup>(1)</sup> :	<b>PP</b>	<input type="checkbox"/>	<b>RPP</b>	<input type="checkbox"/>	<b>SP</b>	<input type="checkbox"/>	<b>RSP</b>	<input type="checkbox"/>	<b>FIN</b>	<input type="checkbox"/>

**(1)** Las instancias validas son: 1<sup>er</sup> Parcial (**PP**), Recuperatorio 1<sup>er</sup> Parcial (**RPP**), 2<sup>do</sup> Parcial (**SP**), Recuperatorio 2<sup>do</sup> Parcial (**RSP**), Final (**FIN**). Marque con una cruz.

**(2)** Campos a ser completados por el docente.

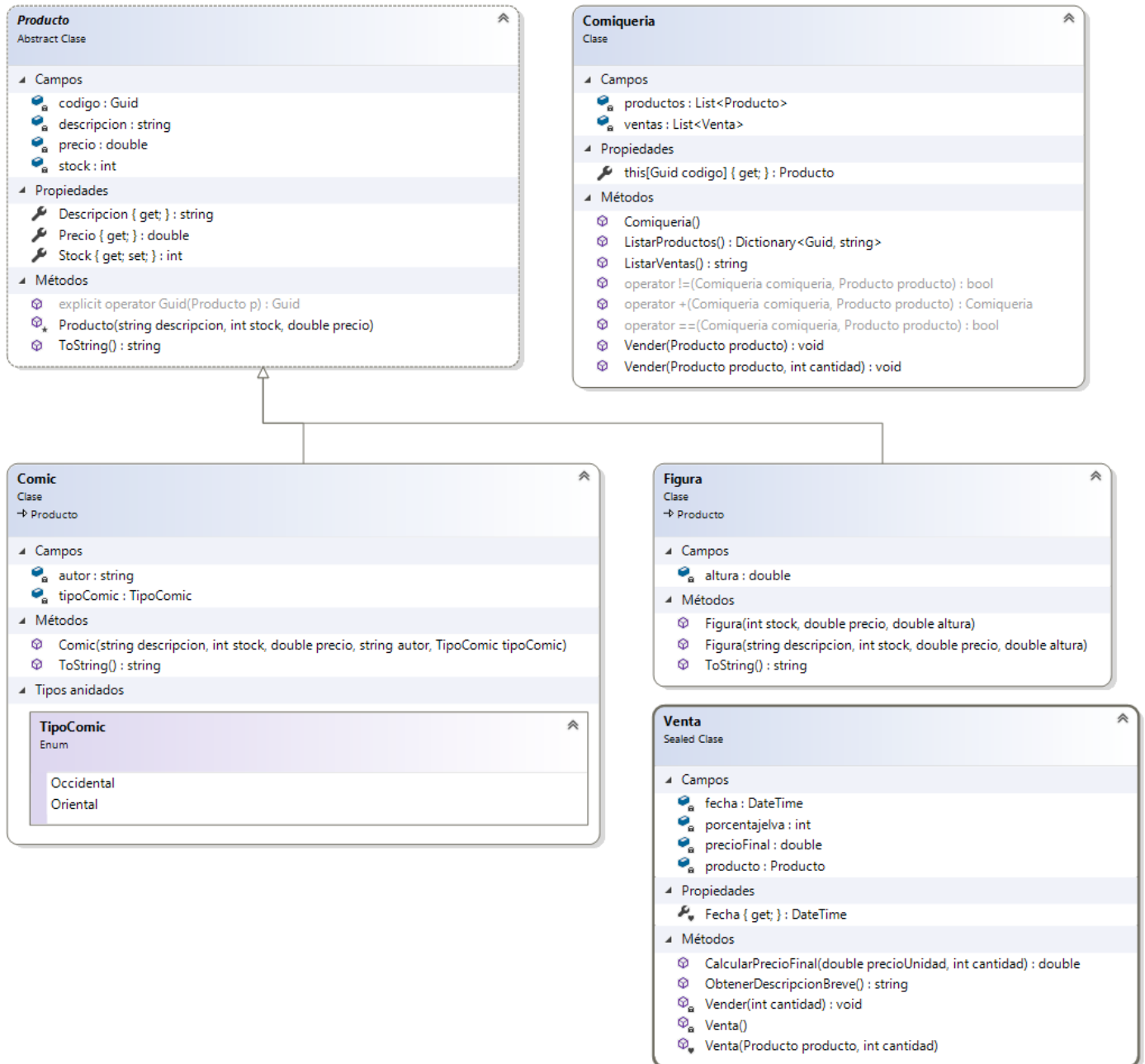
**IMPORTANTE:**

- **2 (dos) errores en el mismo tema anulan su puntaje.**
- La correcta documentación y reglas de estilo de la cátedra serán evaluadas.
- Colocar sus datos personales en el nombre del proyecto principal, colocando: Apellido.Nombre.Division. Ej: Pérez.Juan.2D. No se corregirán proyectos que no sea identificable su autor.
- No se corregirán exámenes que no compilen.
- **Reutilizar** tanto código como crean necesario.
- Colocar nombre de la clase (en estáticos), **this** o **base** en todos los casos que corresponda.
- Aplicar los principios de la encapsulación. Los datos sólo deben ser accesibles mediante propiedades y/o métodos cuando sea necesario y realizando las validaciones y cálculos correspondientes. Se indicará cómo y cuándo hacerlo en las siguientes instrucciones.

Se desarrollará una aplicación para una comiquería.

En una primera etapa se podrá visualizar la lista de productos disponibles, el detalle de cada producto y el historial de ventas realizadas. Además, se informará el precio final de los productos indicando la cantidad que se desea vender.

1. Trabajar sobre la solución "ComiqueriaApp" que acompaña al examen. No tocar el código que viene con la solución.
2. Crear la biblioteca de clases "ComiqueriaLogic" y programar las siguientes clases:



### 3. Clase Producto:

- Clase abstracta.
- Constructor protegido. Inicializará el campo “codigo” con el método estático NewGuid de la clase Guid. Iniciar el resto de los campos con los parámetros de entrada.

*El identificador único global, en inglés: globally unique identifier (GUID) es un número pseudoaleatorio empleado en aplicaciones de software. El GUID es una implementación de Microsoft de un estándar llamado universally unique identifier (UUID), especificado por la Open Software Foundation (OSF).*

- La propiedad Stock de lectura y escritura. Validará que el stock ingresado sea mayor o igual a cero, sino no lo actualizará.
- Las propiedades Precio y Descripción son de sólo lectura.
- Tiene un método de conversión **explícito** transforma un producto en su código.
- Sobrescribir el método ToString para devolver un string con los datos de un producto: descripción, código, precio y stock. **Se deberá utilizar StringBuilder y/o métodos de la clase String, no utilizar operador + y derivados (no concatenar).** El texto devuelto debe verse como en la siguiente imagen de ejemplo.

Descripción: Figura 20 cm  
Código: c4733096-76b6-4709-b3b8-14f3a0e4d7fc  
Precio: \$349,58  
Stock: 1 unidades

#### 4. Clase Comic:

- a. Hereda de la clase Producto.
- b. Tiene un enumerado TipoComic con dos posibles valores: Occidental, Oriental.
- c. El constructor inicializa todos los campos con los parámetros de entrada.
- d. Sobrescribir el método ToString. Agregar a lo devuelto por el ToString de la clase base los datos del Comic (Autor y TipoComic). Seguir el mismo formato y metodología utilizado en la clase base.

#### 5. Clase Figura:

- a. Hereda de la clase Producto.
- b. Los constructores inicializan todos los campos con los parámetros de entrada.
- c. Tiene una sobrecarga del constructor que no recibe descripción. En su lugar inicializará ese campo con el siguiente texto "Figura \*altura\* cm", donde \*altura\* corresponde al valor de la altura de la figura. Reutilizar código en las sobrecargas, no repetir.
- d. Sobrescribir el método ToString. Agregar a lo devuelto por el ToString de la clase base los datos de la Figura (Altura). Seguir el mismo formato y metodología utilizado en la clase base.

#### 6. Clase Venta:

- a. Es una clase sellada (sealed).
- b. El campo porcentajelva es estático.
- c. Tiene un constructor estático que inicializará el campo porcentajelva en 21.
- d. El constructor de instancia sólo debe ser accesible desde el mismo ensamblado (assembly, .dll). Utilice el modificador de acceso que corresponda.
- e. El constructor de instancia inicializará el campo producto y llamará al método Vender pasándole la cantidad indicada como argumento.
- f. El método Vender **es privado** y de instancia.
  - i. Le restará al stock del producto la cantidad que le pasaron por parámetro.
  - ii. Inicializará el campo fecha con la fecha actual completa.
  - iii. Inicializará el campo precioFinal con el valor retornado por el método CalcularPrecioFinal, al cual se le pasará el precio unitario del producto y la cantidad.
- g. El método CalcularPrecioFinal **es estático y público**. Calculará el precio final multiplicando el precio unitario por la cantidad comprada y a este resultado le aplicará el porcentaje de IVA que esté indicado en el campo porcentajelva.
- h. El método ObtenerDescripcionBreve devuelve un string breve y en una sola línea indicando fecha, descripción del producto y precio final. De esta forma: "*fecha – descripción – precioFinal*". El precio deberá mostrarse con 2 decimales.
- i. Tiene una propiedad que devuelve la fecha de la venta. Es de sólo lectura y tiene el mismo modificador de acceso que el constructor de instancia.

#### 7. Clase Comiqueria:

- a. El constructor instanciará los campos de tipo List.
- b. Tendrá un indexador que recibirá como parámetro un código Guid y devolverá el producto de la lista de productos que corresponda con ese código. Si no encuentra ningún producto devolverá null.
- c. La sobrecarga del operador == debe verificar si el producto se encuentra en la lista de productos comparando la descripción. Si dos productos tienen la misma descripción son el mismo producto.
- d. La sobrecarga del operador != debe reutilizar código, no repetir.

- e. La sobrecarga del operador + debe agregar un producto a la lista de producto siempre que el mismo ya no exista en la lista.
- f. El método Vender agrega una nueva venta a la lista de ventas.
- g. El método Vender tiene una sobrecarga que sólo recibe un producto (sin cantidad). Llamará a la sobrecarga más compleja pasándole como argumento a la cantidad el valor de 1.
- h. ListarVentas devuelve un string conteniendo la descripción breve de cada venta en la lista de ventas. Una descripción por línea. Utilizar las herramientas que expone la clase Venta.  
**La lista deberá estar ordenada por fecha de la más reciente a la más antigua** (Para esto utilice el mecanismo que conozca y prefiera. Si lo necesita, agregue un método extra o lo que requiera).
- i. ListarProductos devuelve un Dictionary<Guid, string>. Cada elemento del diccionario corresponderá con cada producto en la lista de productos. La key será el código del producto y el valor la descripción del producto.

## 8. PrincipalForm:

Es el formulario que se ejecutará por defecto. Utilizar la estructura que se entrega en el proyecto del examen.

- a. El manejador del evento “OnVenderClick” del evento click del btnVender (ya se encuentra asociado) deberá abrir una nueva ventana del formulario VentasForm **de forma MODAL**. Agregar el código necesario, use como guía los comentarios que acompañan al método.


## 9. VentasForm:

- a. Crear un nuevo formulario llamado “VentasForm” en el proyecto de ComiqueriaApp.  
**Recuerde que la clase del formulario sigue las mismas reglas que el resto de las clases. Modifique el constructor, agregue campos, utilice sus eventos según crea necesario. Va a necesitar acceso a la instancia de la comiquería y al producto seleccionado en el principalForm.**
- b. Cambiar el título a “Nueva Venta” (Propiedad Text).
- c. Agregar los siguientes controles:

- i. **lblDescripcion:** Tipo label que deberá mostrar la descripción del producto que se está comprando. **En negrita.** (Investigar propiedad Font).
  - ii. **lblCantidad:** Tipo label con un texto fijo que diga "Cantidad:".
  - iii. **numericUpDownCantidad:** Tipo NumericUpDown. El número mínimo debe ser 1 (Propiedad Minimum).
  - iv. **lblPrecioFinal:** Tipo label con un texto que diga "Precio Final: \$X.XX". Las X deben ser reemplazadas por el precio final correspondiente a la cantidad que indique el control NumericUpDown. Se debe actualizar siempre que cambie la cantidad. Si lo prefiere, utilice 2 label.
  - v. **btnVender:** Agregar un control de tipo button con un texto que diga "Vender" (Propiedad Text).
  - vi. **btnCancelar:** Agregar otro button de "**Cancelar**".
- d. Todos los controles deben tener nombres descriptivos (Propiedad Name). En el punto anterior se sugieren posibles nombres correctos.
- e. El aspecto del formulario debe ser similar al de la imagen:

- f. Cada vez que se indica una nueva cantidad se actualiza el precio final. Asociar el evento ValueChanged del NumericUpDown.
- g. Al presionar el botón cancelar se debe cerrar el formulario.
- h. Al presionar el botón vender se validará que la cantidad sea menor o igual al stock disponible.
  - i. Si es mayor se deberá mostrar un MessageBox informando al usuario que superó el stock disponible y debe disminuir la cantidad de unidades a vender. **El mensaje debe tener un título, un ícono de error y sólo un botón de aceptar.**
  - ii. Si es menor o igual se venderá el producto. Utilizar el método de la comiquería que corresponda. Además, establecer la propiedad DialogResult del formulario con el valor DialogResult.OK. Por último, cerrar el formulario.

Al finalizar, colocar la carpeta de la Solución completa en un archivo ZIP que deberá tener como nombre Apellido.Nombre.division.zip y dejar este último en el Escritorio de la máquina.

Luego presionar el botón  de la barra superior, colocar un mensaje y apretar **Aceptar**. Finalmente retirarse del aula y aguardar por la corrección.