Universidad Tecnológica Nacional
Facultad Regional Avellaneda

Técnico Superior en Programación - Técnico Superior en Sistemas Informáticos

Materia: Laboratorio de Programación II

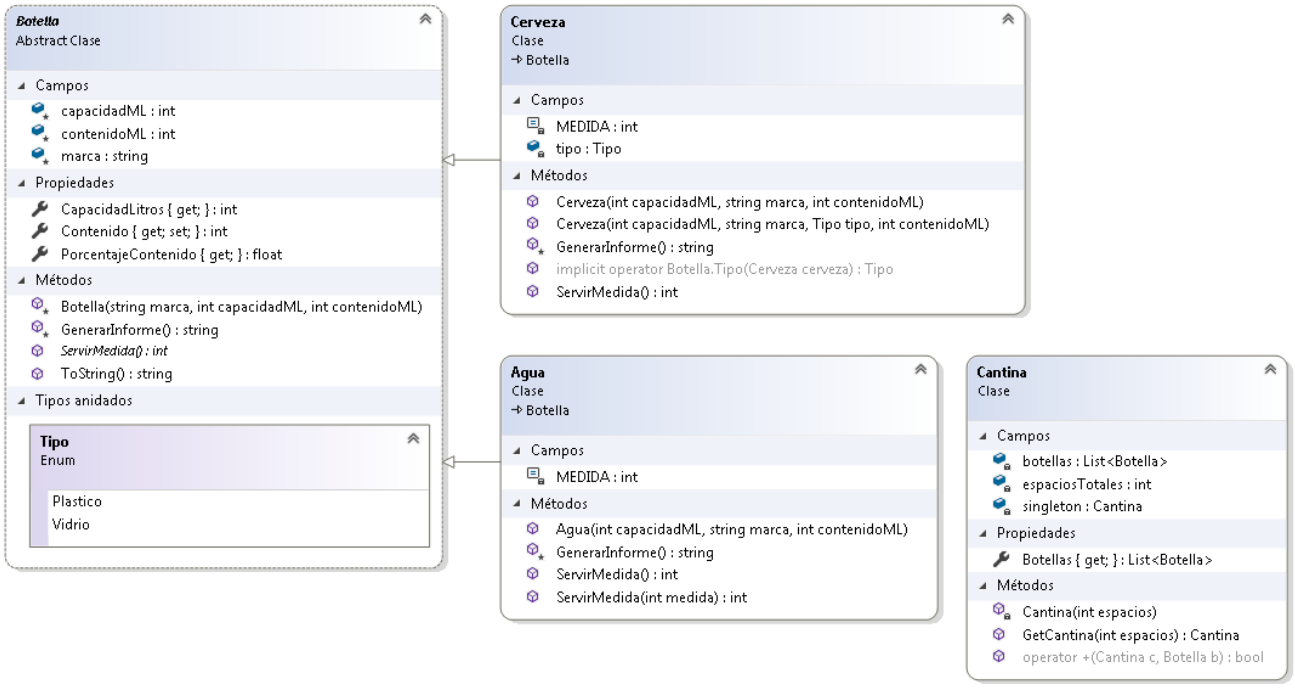
Apellido:					Fecha:		10-10-2019			
Nombre:					Docente(2):					
División:	2°C				Nota(2):					
Legajo:					Firma(2):					
Instancia(1):	**PP**	X	**RPP**		**SP**		**RSP**		**FIN**	

(1) Las instancias validas son: 1er Parcial (**PP**), Recuperatorio 1er Parcial (**RPP**), 2do Parcial (**SP**), Recuperatorio 2do 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 de la carpeta principal y la solución: Apellido.Nombre.Div. Ej: Pérez.Juan.2D. No sé 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 los 4 pilares de la POO.

1. Crear un proyecto del tipo Biblioteca de Clases y colocar el siguiente esquema de clases:



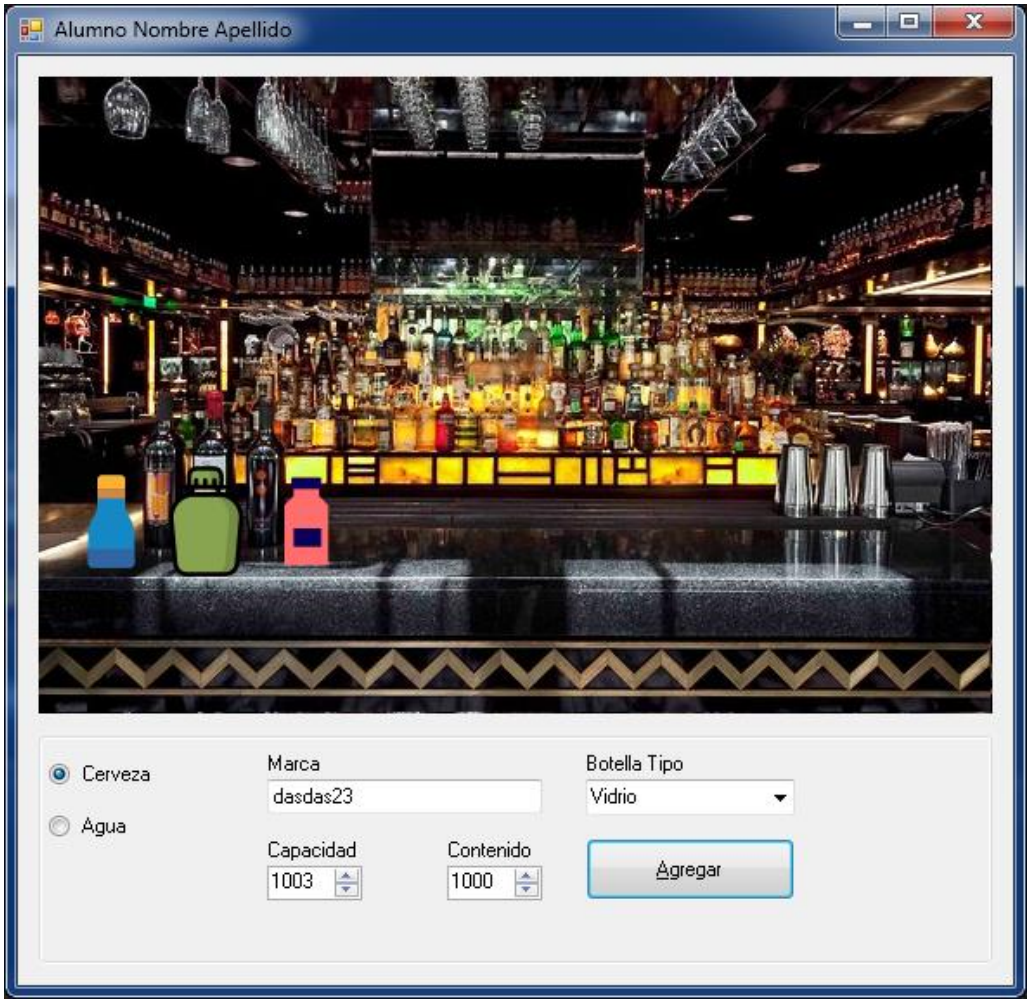
2. Clase **Botella**:
    a. Será abstracta.
    b. Tanto la capacidad como el contenido están expresados en mililitros.
    c. Tendrá un único constructor, en el cual se validará que si la capacidad es menor al contenido, el contenido será reemplazado por la por la variable de capacidad. Así nunca se podrá tener más contenido en una botella que la capacidad que ésta es capaz de guardar.
    d. La propiedad `CapacidadLitros` retornará la capacidad convertida a litros (dividir por 1000).
    e. Utilizar regla de 3 simple para retornar el valor de la propiedad `PorcentajeContenido`.
    f. `ServirMedida` será abstracto.
    g. `GenerarInforme` utilizará StringBuilder para retornar todos los datos de la botella.
    h. `ToString` retornará `GenerarInforme`.

3. Clase **Agua**:
    a. MEDIDA será una constante con el valor 400.
    b. `ServirMedida` gastará unidades de contenido con la siguiente lógica:
        i. Si la constante MEDIDA es menor o igual al contenido, gastará MEDIDA.
        ii. Si MEDIDA es mayor al contenido, tomará contenido.
    c. Generar una sobrecarga que reciba una cantidad a gastar que pueda ser diferente de MEDIDA. No repetir código.
    d. `GenerarInforme` utilizará StringBuilder para retornar todos los datos de la botella de agua.

4. Clase **Cerveza**:
    a. MEDIDA será una constante con el valor 330.
    b. En los constructores, si no se asigna ningún Tipo de Botella, se asignará Vidrio. No repetir código.
    c. `ServirMedida` gastará unidades de contenido con la misma lógica que agua, sólo que servirá el 80% del valor indicado en la variable MEDIDA, para dejar espacio para la espuma.
    d. `GenerarInforme` utilizará StringBuilder para retornar todos los datos de la botella de cerveza.

5. Clase **Cantina**:
    a. El único constructor será privado y se encargará tanto de inicializar la lista como de asignar la cantidad de espacios disponibles para guardar botellas.
    b. `GetCantina` será de clase e implementará un patrón Singleton, para lo cual deberá:
        i. Si la variable de clase `singleton` es null, instanciará el objeto.
        ii. Si no es null, modificará la cantidad de espacios en la cantina.
        iii. En ambos casos, su última acción será retornar el objeto `singleton`.
    c. El operador + agregará, siempre y cuando aún no se hayan ocupado todos los espacios disponibles, retornando true si agregó y false en caso contrario.

6. Clase **FrmCantina**:
    a. Agregar los controles del tipo Label, TextBox, NumericUpDown, ComboBox, Button y RadioButton y Barra para lograr el siguiente formulario:



b. Los NumericUpDown aceptarán valores entre 1 y 5000, siendo su valor inicial 1000.

c. En el evento Load del Formulario se deberá hacer:

```
this.barra.SetCantina = Cantina.GetCantina(10);
```

d. El ComboBox se llenará en el evento Load del Formulario con:

```
cmbBotellaTipo.DataSource = Enum.GetValues(typeof(Botella.Tipo));
```

e. El enumerado del ComboBox se leerá con:

```
Botella.Tipo tipo;
Enum.TryParse<Botella.Tipo>(cmbBotellaTipo.SelectedValue.ToString(), out tipo);
```

f. Al presionar el botón Agregar se agregará una nueva botella al objeto Barra teniendo en cuenta los RadioButton (si se encuentra seleccionado Cerveza se creará una nueva botella de ese tipo, y si se encuentra Agua una botella de agua).

7. Control de usuario **Barra**:
   a. Modificar el método `ServirCopa` para que:
      i. Si `ServirMedida` retorna 0, muestre el mensaje "No queda más líquido!" en vez de "SIRVIENDO!".
   b. No modificar ningún otro aspecto del **ControlesUsuario**. Este proyecto utilizará la biblioteca de clases y será utilizado en el proyecto de formularios que se crearán a continuación.