Universidad Tecnológica Nacional
Facultad Regional Avellaneda

Técnico Superior en Programación - Técnico Superior en Sistemas Informáticos										
Materia: Laboratorio de Programación II										
Apellido:				Fecha:		24-10-2019				
Nombre:				Docente(2):						
División:	2°C			Nota(2):						
Legajo:				Firma(2):						
Instancia(1):	**PP**		**RPP**	X	**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:

- Guardar el proyecto en el **disco D:**. Ante un corte de energía o problema con el archivo de corrección, el alumno será responsable de que el proyecto sea recuperable.
- **2 (dos) errores en el mismo tema anulan su puntaje.**
- **Errores de conceptos de POO anulan el punto.**
- **Cada tema vale 1 (un) punto (Herencia, Generics, Test Unitarios, etc.). La correcta documentación también será evaluada.**
- **Se deberán tener al menos el 60% bien de los temas a evaluar según la instancia para lograr la aprobación.**
- Colocar sus datos personales en el nombre del proyecto principal, colocando: Apellido.Nombre.AñoCursada. Ej: Pérez.Juan.2019. No sé corregirán proyectos que no sea identificable su autor.
- **Salvo que se indique lo contrario, TODAS** las clases deberán ir en una Biblioteca de Clases llamada Entidades.
- No se corregirán exámenes que no compilen.
- **Reutilizar** tanto código como crean necesario.

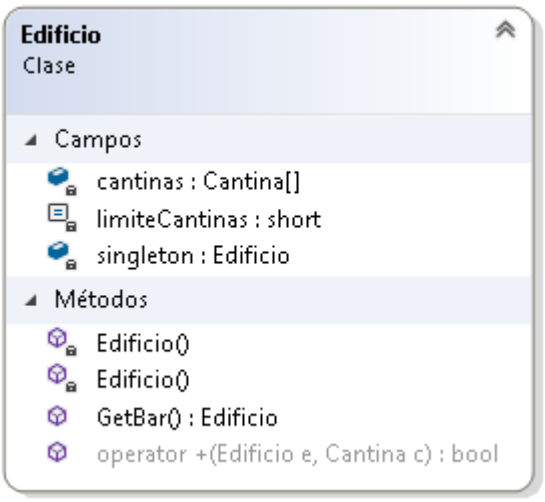
Al finalizar, colocar la carpeta de la Solución completa en un archivo ZIP que deberá tener como nombre Apellido.Nombre.AñoCursada.zip y dejar este último en el Escritorio de la máquina. Luego presionar el botón  de la barra superior, **colocar un mensaje** y presionar *Aceptar*. **Aguardar a que el profesor indique que el examen fue copiado de forma correcta.** Luego retirarse del aula.

---

*TIEMPO MÁXIMO PARA RESOLVER EL EXAMEN **110** MINUTOS.*

---

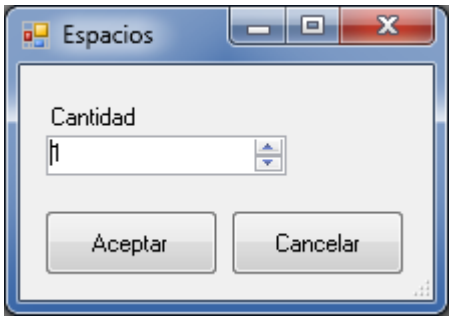
1. Agregar dentro de la biblioteca la siguiente clase:



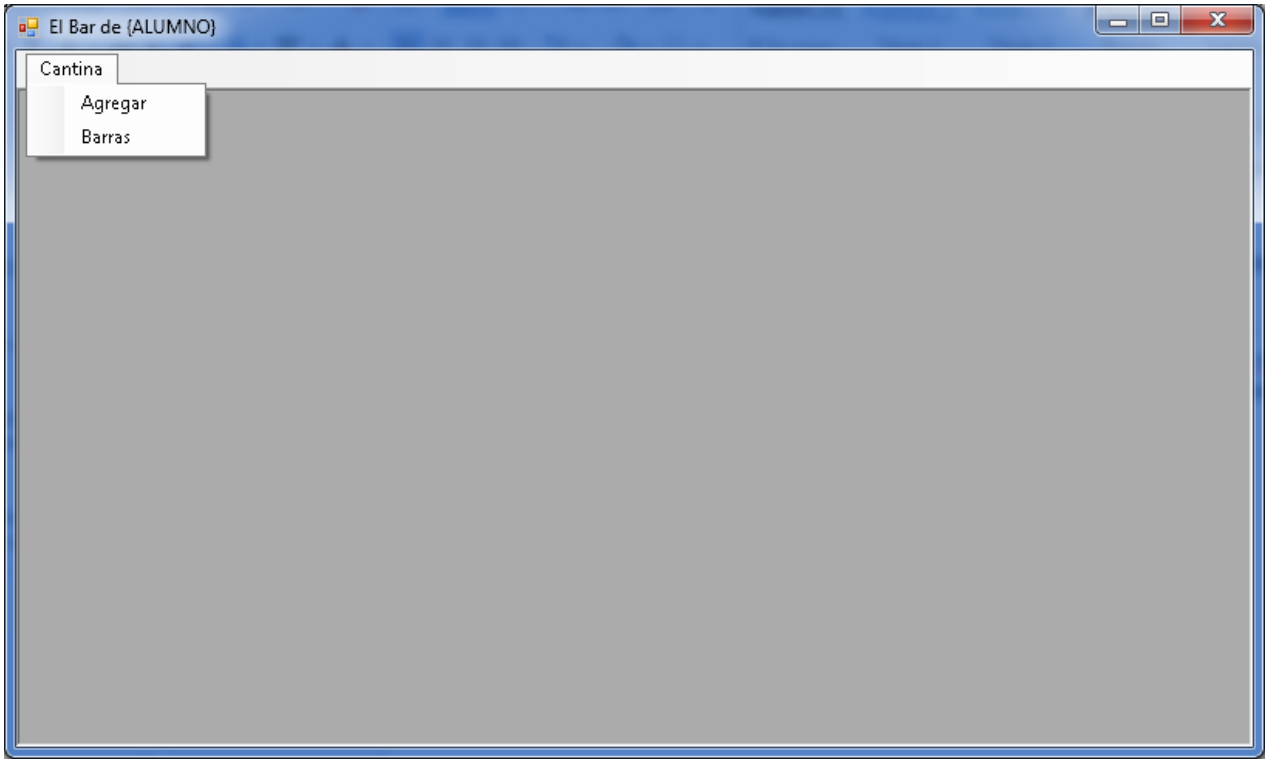
2. Clase **Edificio**:
    a. No se deberá poder heredar de ella.
    b. *limiteCantinas* será una constante con valor 2.
    c. Tendrá un constructor privado que instanciará el array *cantinas* con un tamaño de *limiteCantinas*.
    d. Tendrá un constructor de clase que inicializará la variable *singleton*.
    e. GetBar retornará *singleton*.
    f. El operador + buscará una posición libre en el array:
        i. Si la encuentra, cargará la cantina en ese lugar y retornará true.
        ii. Si se encuentra completo, retornará false.
3. Clase **Cantina**:
    a. Quitar patrón singleton y hacer público su constructor.
    b. Agregar un operador == entre Cantina y Botella para encontrar si dentro de la lista presente en Cantina se encuentra la Botella en cuestión. Dos botellas serán iguales según lo que retorne el == entre ellas.
4. Clase **Botella**:
    a. Convertir la propiedad `PorcentajeContenido` en abstracta:
        i. La botella de Agua calculará su porcentaje con la fórmula: `(this.contenidoML * 100) / this.capacidadML`.
        ii. La botella de Cerveza lo hará con: `(this.contenidoML * 90) / this.capacidadML`.
    b. Quitar sobreescritura de ToString. Hacer la misma operación dentro de un conversor implícito.
    c. Dos botellas serán iguales (==) si y sólo si comparten la misma Marca.
    d. Dentro del método GenerarInforme se deberá utilizar StringBuilder y string.Format.
5. En ningún caso se podrá utilizar concatenaciones de texto. O sea, queda prohibido el uso de **"Texto" + variable** o similares.
6. Clase **FrmCantina**:
    a. Crear la propiedad de sólo lectura *GetInforme* que retornará la información de todas las botellas.
    b. Agregar el evento FormClosing. Cancelar el cierre sólo si se cumple la condición `e.CloseReason == CloseReason.UserClosing`. Para cancelar el evento de cierre se deberá utilizar e.Cancel.

c. Agregar la propiedad GetCantina que retorne `this.barra.Cantina`.

7. Agregar el formulario *FrmCantidadEspaciosCantina*:
   a. Deberá iniciar centrado en la pantalla.
   b. Al presionar btnCancelar retornar DialogResult.Cancel y al presionar btnAceptar DialogResult.OK. Ambos botones cerrarán el formulario.
   c. Agregar la propiedad de solo lectura CantidadEspacios que retornará el **short** el contenido del nudCantidad.



8. Agregar el formulario *FrmBar*:



   a. El formulario será del tipo MDI.
   b. Colocar su nombre en el título.
   c. Implementar el StripMenu como se muestra en la imagen.
   d. Menú Agregar abrirá el *FrmCantidadEspaciosCantina*. Si retorna OK como respuesta:
      i. Consultar la cantidad de espacios que se cargó en *FrmCantidadEspaciosCantina.*
      ii. Instanciará un nuevo *FrmCantina*.
      iii. Si la operación `this.edificio + frmCantina.GetCantina` retorna true:

```
frmCantina.MdiParent = this;
frmCantina.Show();
```

e. Menú Barras utilizará la propiedad `this.MdiChildren` para llamar al método *GetInforme* de cada *FrmCantina* instanciado.