Unidad

3

DIPLOMATURA EN PROGRAMACION .NET

Ejercicios

Capítulo 5

Clases: Conceptos Avanzados

Ejercicio 1

En este ejercicio se modificará la clase Banco para implementar el patrón de diseño Singleton, según se muestra en los diagramas UML.

C# y VB

Diagrama de paquetes:

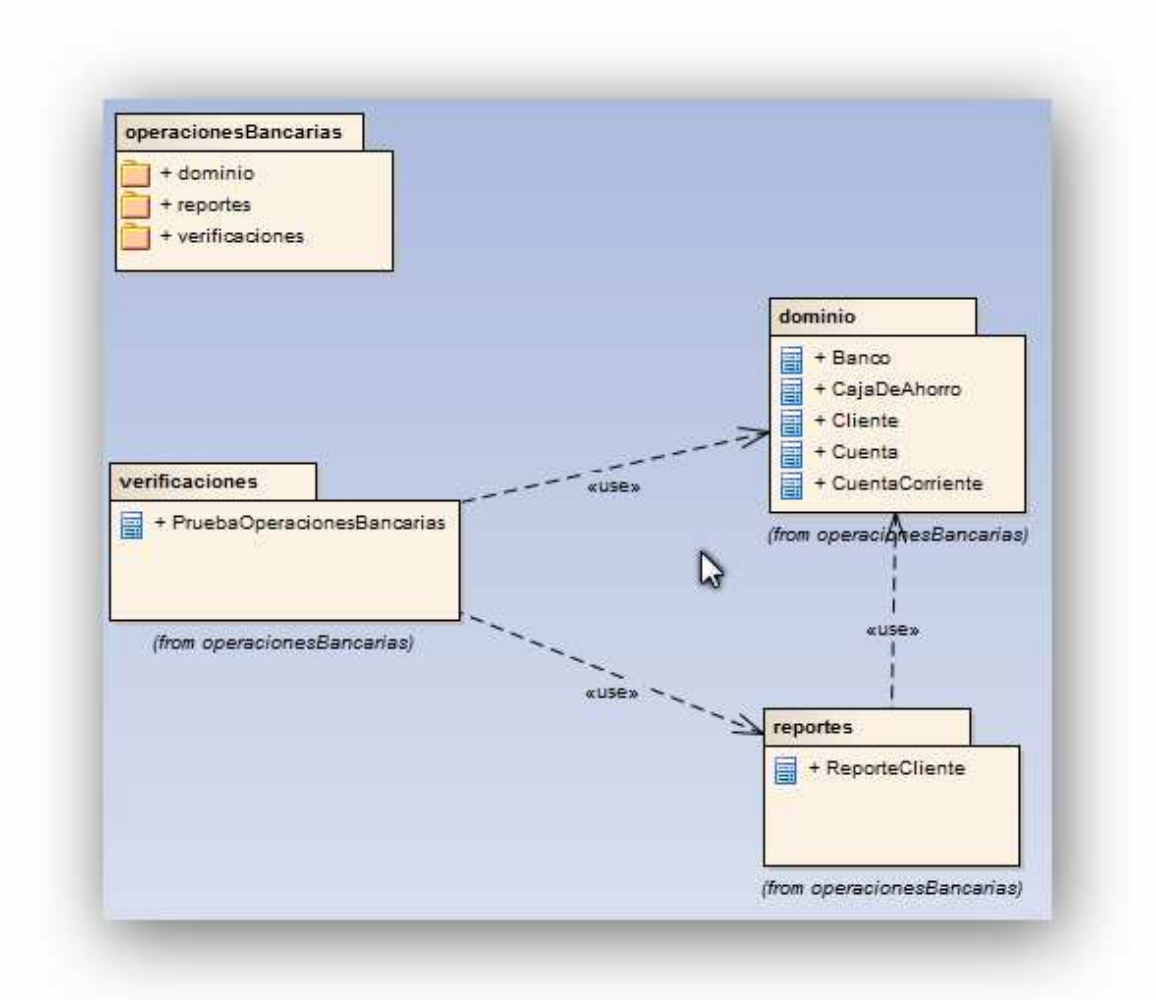
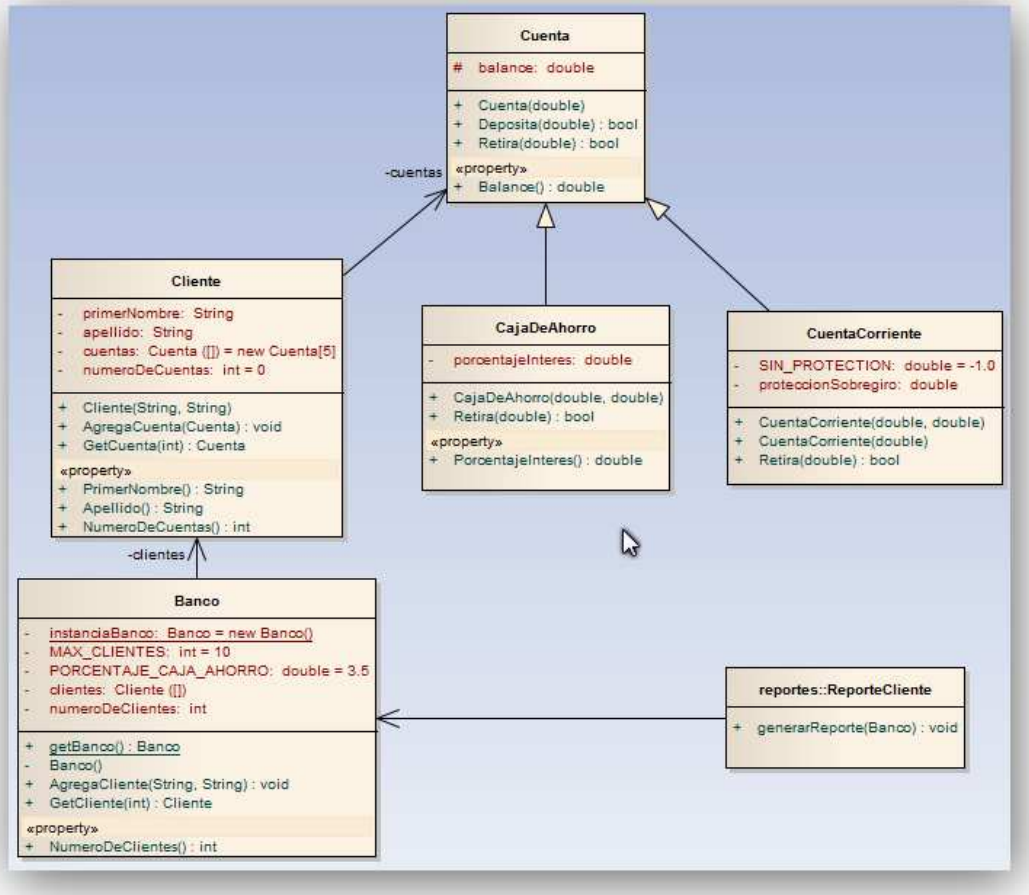
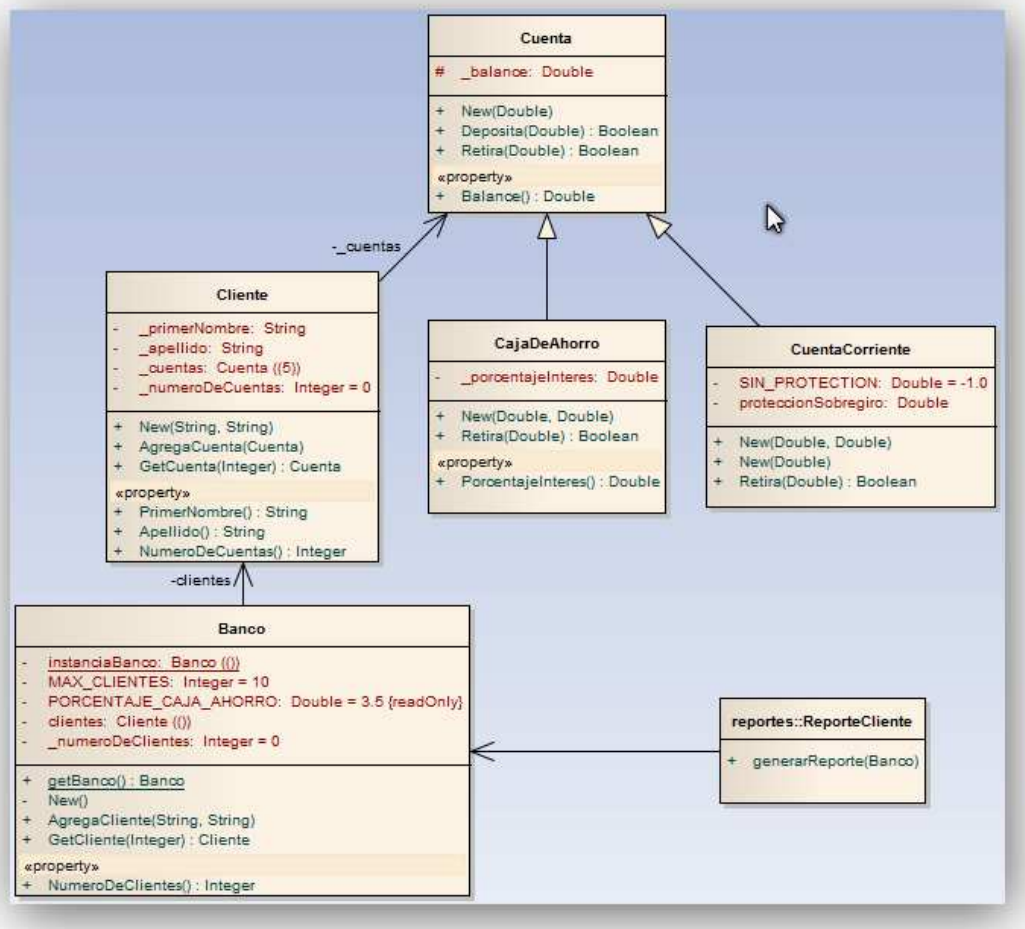


Diagrama de clases:

C#



VB



En este punto el proyecto necesita una jerarquía de paquetes más compleja. Por lo tanto, las clases del dominio (principales) necesitan estar en el espacio de nombres operacionesBancarias.dominio, razón por la cual se necesita cambiar la declaración.

Realizar los siguientes pasos:

1. Abrir la solución del directorio Capitulo5. Los errores que figuran luego de importar los archivos se deben a las modificaciones que deberán realizarse.
2. Modificar la declaración del espacio de nombres en todas las clases, que correspondan según el diagrama UML, para que estas pertenezcan ahora a operacionesBancarias.dominio. Respetar el diagrama para los espacios de nombres de las clases que no pertenecen a dominio.

Modificar la clase Banco para implementar el patrón de diseño Singleton.

3. Modificar la clase Banco para crear un atributo estático con un modificador de acceso (visibilidad) privada. De igual modo, el constructor de Banco deberá ser privado.
4. Esta instancia única deberá ser retornada por un método estático público llamado GetBanco() que deberá agregarse para cumplir dicho cometido.
5. Modificar el atributo MAX_CLIENTES para que sea constante.

Modificar la clase ReporteCliente

En el proyecto del ejercicio anterior, se creaba un reporte de clientes que estaba embebido dentro del método Main() de la clase PruebaOperacionesBancarias. En este ejercicio, ese código se puso dentro de la clase ReporteCliente en el espacio de nombres operacionesBancarias.reportes. Se deberá modificar esta clase para utilizar el objeto del tipo Banco que es un Singleton para que lo utilice esta clase.

6. Buscar la línea de código marcada como un bloque de comentario /***…***/ en CS y '***…'*** en VB
7. Obtener la referencia a un objeto de tipo Banco que es un Singleton y quitar la definición de la referencia recibida como parámetro en el método.
8. Compilar y ejecutar el programa
9. Verificar que la salida sea la siguiente:

```
Creando al cliente Juan Perez
Creando una caja de ahorros con 500.00 de balance con el 3% de
interés.
Creando al cliente Pedro García.
Creando una cuenta corriente con 500.00 de balance y sin protección
de sobregiro
.
Creando al cliente Oscar Toma.
Creando una cuenta corriente con 500.00 de balance y 500.00 de
protección de sob
regiro.
Creando a la cliente Maria Soley.
María comparte su caja de ahorro con su esposo Oscar.

Recuperando al cliente Juan Perez y su caja de ahorro.
Retira 150.00: True
Deposita 22.50: True
Retira 47.62: True
Retira 400.00: False
Cliente [Pérez, Juan] tiene un balance de 222,5

Recuperando al cliente Pedro García y su cuenta corriente.
Retira 150.00: True
Deposita 22.50: True
Retira 47.62: True
```

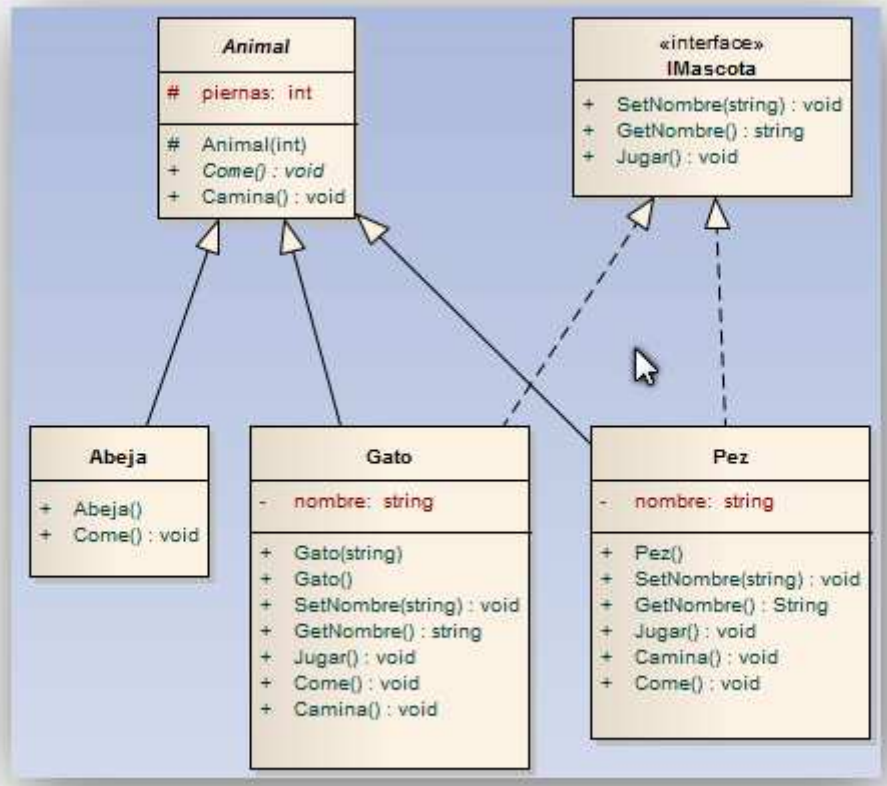
```
Retira 400.00: False
Cliente [García, Pedro] tiene un balance de 222,5

Recuperando al cliente Oscar Toma y su cuenta corriente.
Retira 150.00: True
Deposita 22.50: True
Retira 47.62: True
Retira 400.00: True
Cliente [Toma, Oscar] tiene un balance de 0

Recuperando a la cliente Maria Soley y su caja de ahorro unida a su
esposo Oscar
.
Deposita 150.00: True
Retira 750.00: False
Cliente [Soley, Maria] tiene un balance de 150

                        REPORTE DE CLIENTES
                        ===================

Cliente: Pérez, Juan
    Caja de Ahorro: el balance actual es 222,5

Cliente: García, Pedro
    Cuenta Corriente: el balance actual es 222,5

Cliente: Toma, Oscar
    Cuenta Corriente: el balance actual es 0
    Caja de Ahorro: el balance actual es 150

Cliente: Soley, Maria
    Caja de Ahorro: el balance actual es 150
```

Ejercicio 2

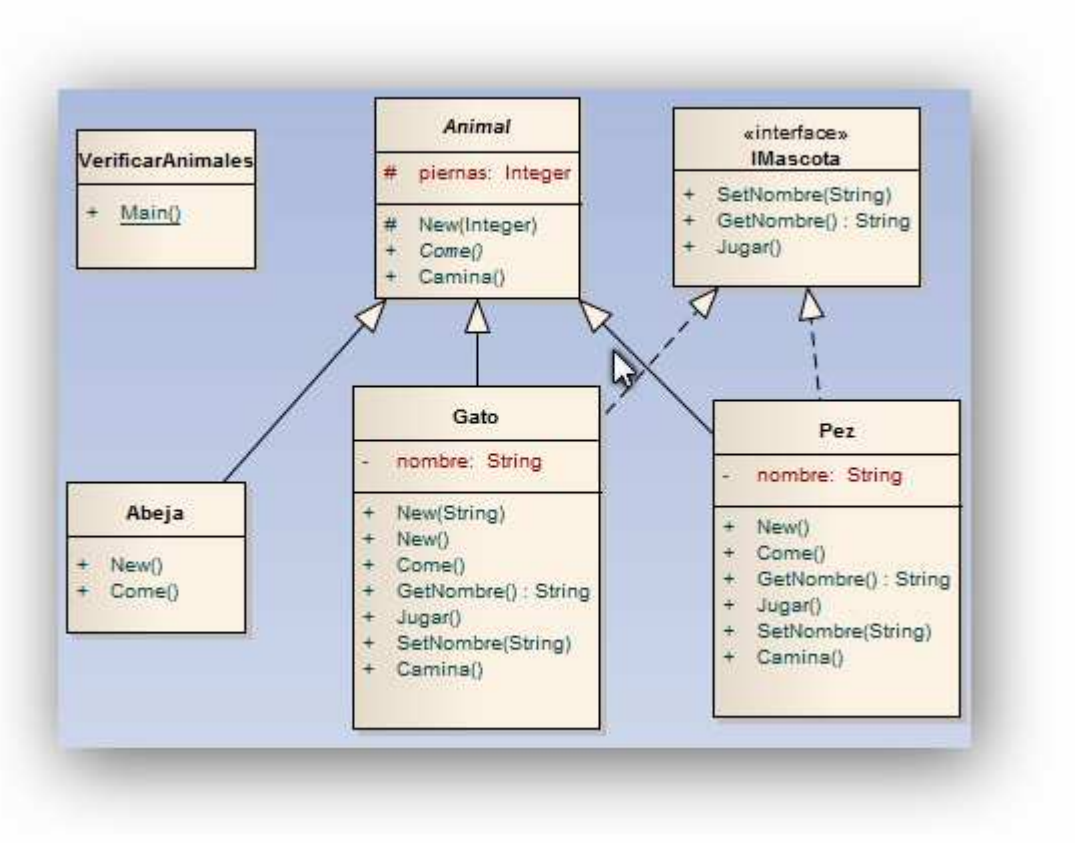
En este ejercicio se creará una jerarquía de animales la cual tiene una clase abstracta como base de su cadena de herencia. Varias de las clases “que son” animales implementarán una interfaz llamada IMascota.

En el gráfico UML se muestra el resultado final de la jerarquía a establecer

C#



VB



Realizar los siguientes pasos:

1. Crear un proyecto llamado "ejercicio2" en la solución Capítulo 5
2. En este ejercicio crear las clases en el espacio de nombres "animales".
3. Crear una clase Animal, la cual es una superclase abstracta de todos los animales
4. Declarar un atributo entero con visibilidad protegida llamado patas, el cual almacena el número de pata de cada animal.
5. Definir un constructor protegido que inicialice el atributo patas
6. Declarar un método abstracto llamado Come()
7. Declarar un método concreto (que tenga un cuerpo de sentencias aunque este vacío) llamado Camina() que muestre por pantalla una cadena que indique si el animal camina y con cuantas patas

Crear la clase Abeja

8. La clase Abeja hereda de Animal
9. Definir el constructor por defecto (el que no recibe parámetros) que llamará al constructor de la superclase que especifique que todas las abejas tienen 6 patas.
10. Implementar el método Come() (sobrescribirlo y hacerlo concreto).

Crear la interfaz IMascota

Como se especifica en el diagrama UML. Asegurarse que ninguno de los métodos tiene un cuerpo de sentencias.

Crear la clase Gato

11. Esta hereda de Animal e implementa la interfaz IMascota
12. Esta clase deberá incluir un atributo de tipo string para almacenar el nombre de la mascota.
13. Definir un constructor que reciba un parámetro de tipo string y especifique el nombre del gato. Este constructor deberá llamar también al de la superclase para especificar que todos los gatos tienen cuatro patas.
14. Definir otro constructor que no reciba parámetros. Este constructor deberá llamar al anterior (utilizar la palabra reservada `this` o `Me` según el caso) y pasar un string vacío como argumento
15. Implementar los métodos de la interfaz IMascota.
16. Implementar el método Come()

Crear la clase Pez

17. Sobrescribir los métodos de la clase Animal para especificar que un pez no puede caminar y no tiene patas.

Crear un programa VerificarAnimales

18. En el método Main() crear y manipular instancias de las clases creadas anteriormente. Un ejemplo podría ser:

C#

```
Pez p = new Pez();
Gato g = new Gato("Manchas");
Animal a = new Pez();
Animal e = new Abeja();
IMascota g2 = new Gato();
```

VB

```
Dim p As New Pez()
Dim g As New Gato("Manchas")
Dim a As New Pez()
Dim e As New Abeja()
Dim g2 As IMascota = New Gato()
```

19. Experimentar con:
20. Llamar los métodos de cada objeto
21. Modificar los tipo en las asignaciones (casting)
22. Utilizar polimorfismo para llamar a los métodos
23. Utilizar la palabra reservada `base` o `Mybase` (C# o VB) para llamar a los métodos de la superclase.