

Resolución Parcial Programación Orientada a Objetos.

Jaquelina Lopez Abramo.

1

A) Agregar en el examen la cardinalidad de las relaciones entre las clases. ✓

Para comenzar con la resolución del parcial se agregara la **cardinalidad** de las relaciones que tendrán cada una de las clases. Esta establecerá la multiplicidad con que se relacionen las clases en la asociación.

Cuando hablamos de asociación nos referimos a la dependencia que existirá entre las clases, que de no existir esto serian independientes.

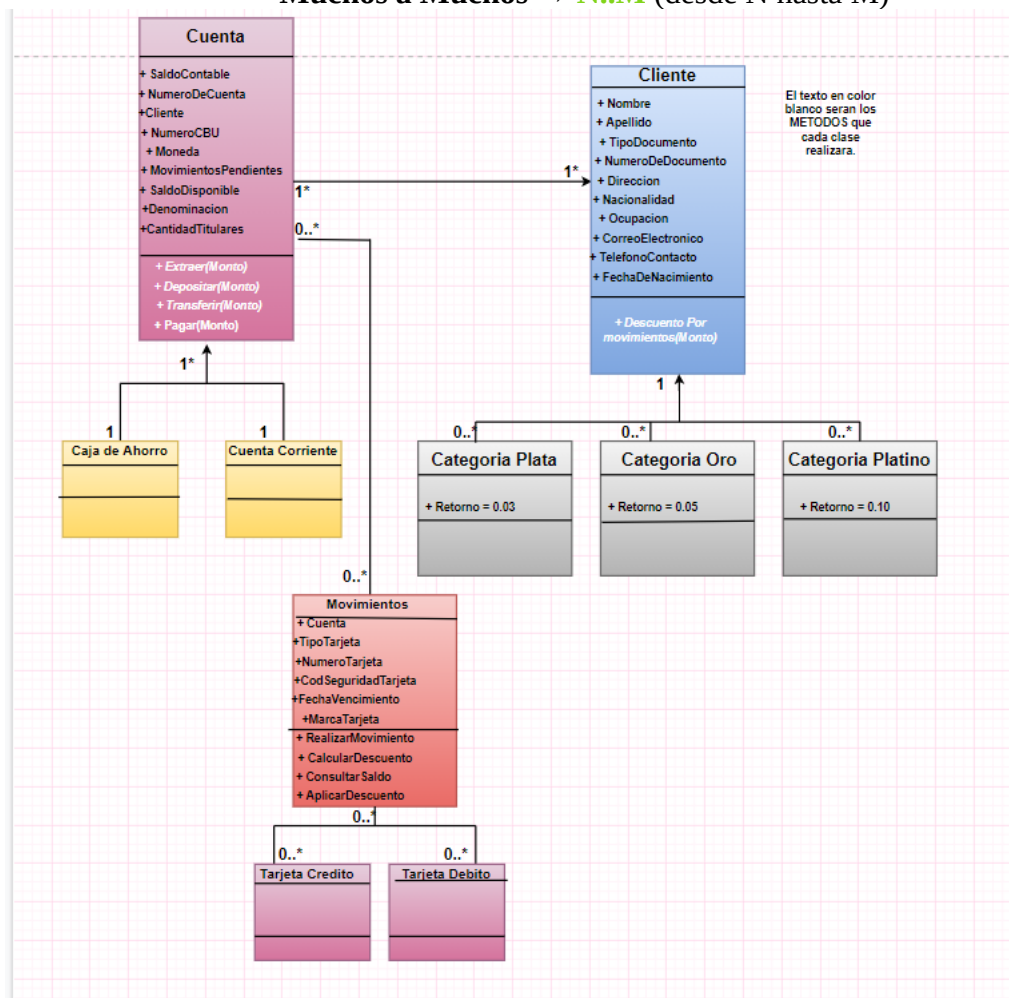
Los tipos de relaciones serán: Uno solo 1 → 1

Cero a Uno → 0..1

Cero a Varios → 0..*

Uno o varios al menos Uno → 1*

Muchos a Muchos → N..M (desde N hasta M)



Explicación:

Cuenta: Una cuenta podrá estar conformada por **UNO o MAS** clientes. Pero al menos deberá tener uno. **'1*'**.

Un cliente podrá tener **UNA o MAS** **'1*'** cuentas. Y a su vez tendrá asociada UNA **'1'** Categoría que sera si pertenece a la categoría Plata, Oro o Platino estas categorías estarán asociadas a **Cero o varios clientes**. Ya que la categoría en la que el cliente este y la bonificación que se retorne de esta dependerá de un cliente. Es decir un cliente no podrá ser de la categoría Oro y Platino a la vez. Pero

a la vez podrá tener varios clientes en la categoría por ejemplo Plata o ningún cliente que cumpla las condiciones para estar en esa categoría, y no tener a ninguno.

Una cuenta podrá realizar **Cero o Varios** movimientos '0..*', por ejemplo podrá tener una tarjeta de crédito y de débito y realizar varios movimientos con la tarjetas durante un día. O no realizar ninguno.

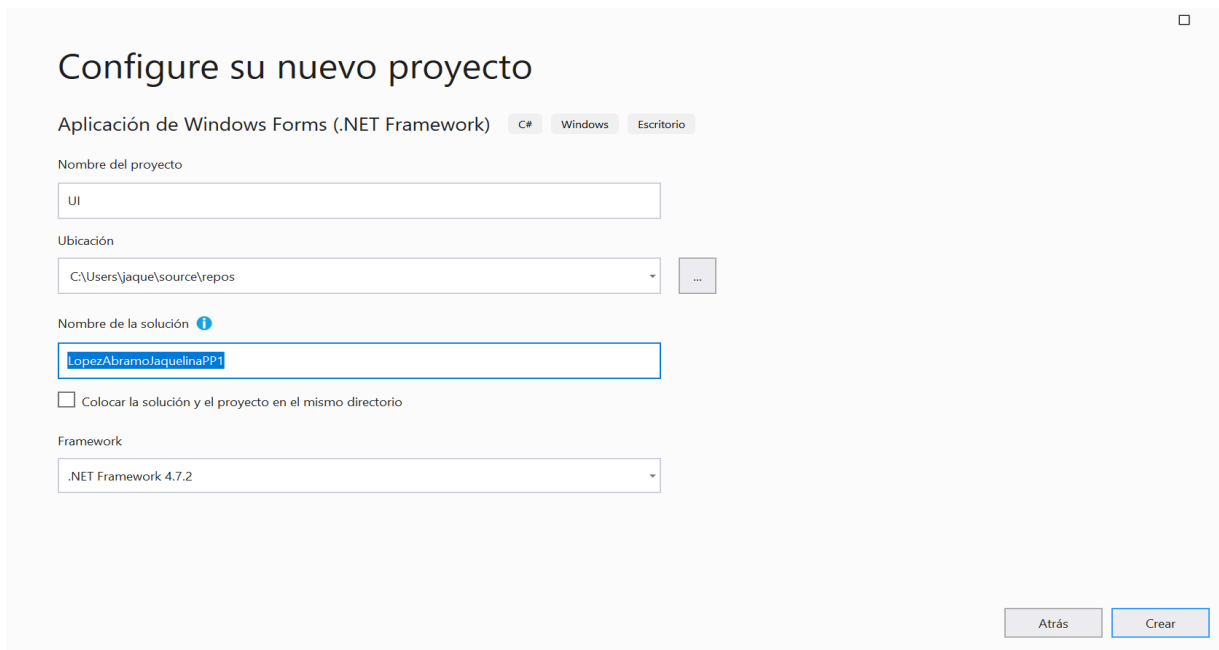
A su vez, una cuenta podrá tener **UNA o MAS** '1*' cuentas asociadas a ese titular por ejemplo puede pasar que el cliente tenga asociados los productos 004 caja de ahorro en dolares, 003 caja de ahorro en peso y 001 cuenta corriente. Todas estas cuentas estarian asociadas al mismo titular.

La caja de ahorro y la cuenta corriente estarán asociadas solamente a una **ÚNICA** '1' cuenta ya que no puedo tener una caja de ahorro asociada a dos cuentas diferentes.

Una persona podrá realizar **CERO o VARIOS** movimientos con sus tarjetas "0..*".

B) Una solución con nombre ApellidoNombrePP1.. ✓

C) Un proyecto Winforms con el nombre "UI" *.. ✓



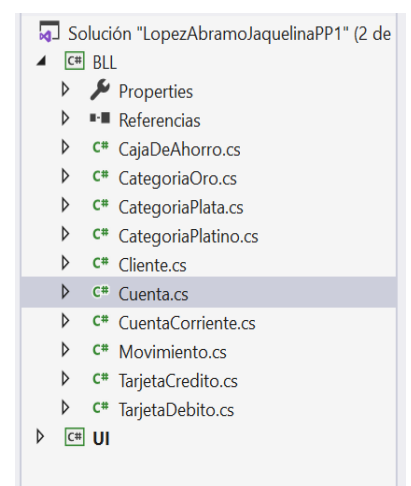
D)
Las

clases debe estar implementadas en otro ensamblado con el nombre BLL.. ✓

Agregare en BLL todas las clases:

Cuenta.cs
CajaDeAhorro.cs
CuentaCorriente.cs
Cliente.cs
Platino.cs
Oro.cs
Plata.cs
Movimiento.cs
TarjetaCredito.cs
TarjetaDebito.cs

E) Implementar todas las clases según el diagrama de clases adjunto en el examen. ✓



F) Implementar correctamente las relaciones entre las clases. ✓

En la clase Cuenta, heredaré los atributos y métodos de la clase Cliente ya que necesito tener la información del cliente que está asociada a esa cuenta. Creo un constructor para poder tomar todos los campos que necesito del cliente.

Cuenta corriente y Caja de ahorro heredarán de Cuenta.

Las categorías Oro, Plata y Platino, heredarán la información del cliente.

G). Implementar correctamente los métodos y propiedades de cada uno de ellas. ✓

Clase Cliente:

La clase cliente tendrá las siguientes propiedades:

```
public String Nombre
{
    get { return nombre; }
    set { nombre = value; }
} // Ya así sucesivamente con los demás campos...
```

Tendrá a su vez un constructor:

```
public Cliente(String cNombre, String cApellido, String cTipoDeDocumento, long
cNumeroDocumento, String cDireccion, String cNacionalidad, String cOcupacion, String
cCorreoElectronico, long cTelefono, String cFechaDeNacimiento)
```

Y un método que será:

```
public double DescuentoPorMovimientos { get; set; }
```

Clase Cuenta:

La declarare como una clase **Abstracta**, la cual heredará de la clase Clientes que esto lo realizare para poner traerme los atributos del cliente y los métodos. A su vez esta clase contendrá también a la clase Movimientos que para realizar la instancia (Clase movimiento es abstracta) deberé crear otra clase que me sirva de base para poder realizar esto.

Esta clase tendrá métodos que serán también abstractos:

```
public abstract double Depositar { get; set; }
public abstract double Extraer { get; set; }
public abstract double Transferir { get; set; }
public abstract double Pagar { get; set; }
```

Clase Movimiento:

La declarare como una clase **Abstracta**, la cual heredará de la clase **cuenta**.

Métodos:

```
public abstract double realizarMovimiento { get; set; }
public abstract double calcularMovimiento { get; set; }
public abstract double consultarSaldo { get; set; }
public abstract double AplicarDescuento{ get; set; }
```

Propiedades

```
public long Monto
{
    get { return monto; }
    set { monto = value; }
}
```

H) Implementar la propiedad Monto a la clase movimiento. * ✓

```
public long Monto
{
    get { return monto; }
    set { monto = value; }
}
```

J) Dado un movimiento el mismo conoce la cuenta sobre la que se realice al movimiento y a su vez la cuenta conoce cual es el tipo de cliente dueño de esa cuenta y cual es el porcentaje de descuento (bonificación) según la cuenta. * ✓

- i. Plata : % 3
- ii. Oro : % 5
- iii. Platino : 10%

Clase CategoriaOro:

```
//Declaro Atributo
float aplicarDescuento = 0.03f;

//Declaro Propiedad
public float AplicarDescuento
{
    get { return aplicarDescuento; }
    set { aplicarDescuento = value; }
}
```

Clase CategoriaPlata:

```
//Declaro Atributo
float aplicarDescuento = 0.05f;

//Declaro Propiedad
public float AplicarDescuento
{
    get { return aplicarDescuento; }
    set { aplicarDescuento = value; }
}
```

Clase CategoriaPlatino:

```
//Declaro Atributo
float aplicarDescuento = 0.10f;

//Declaro Propiedad
public float AplicarDescuento
{
    get { return aplicarDescuento; }
    set { aplicarDescuento = value; }
}
```

2)

Requerimientos

La clase **cuenta** debe ser Abstracta. ✓

Los métodos **extraer** y **depositar** deben ser abstractos ✓

La clase **movimiento** debe ser Abstracta. ✓

Los métodos **RealizarMovimiento** y **AplicarDescuento**. ✓

El método Calcular descuento debe ser implementado en la clase Movimiento. ✓

Se copian las clases:

Clases:

clase Cliente:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace BLL
{
    public class Cliente
    {
        //Declaro los atributos
        public string nombre;
        public string apellido;
        public string tipoDocumento;
        public long numeroDeDocumento;
        public string direccion;
        public string nacionalidad;
        public string ocupacion;
        public string correoElectronico;
        public long telefono;
        public string fechaDeNacimiento;

        //Detallo todas las propiedades
        public string Nombre
        {
            get { return nombre; }
            set { nombre = value; }
        }

        public string Apellido
        {
            get { return apellido; }
            set { apellido = value; }
        }

        public string TipoDeDocumento
        {
            get { return tipoDocumento; }
            set { tipoDocumento = value; }
        }
        public long NumeroDeDocumento
        {
            get { return numeroDeDocumento; }
            set { numeroDeDocumento = value; }
        }
        public string Direccion
        {
            get { return direccion; }
            set { direccion = value; }
        }
        public string Nacionalidad
        {
            get { return nacionalidad; }
            set { nacionalidad = value; }
        }
    }
}
```

```

    public string Ocupacion
    {
        get { return ocupacion; }
        set { ocupacion= value; }
    }

    public string CorreoElectronico
    {
        get { return correoElectronico; }
        set { correoElectronico = value; }
    }

    public long Telefono
    {
        get { return telefono; }
        set { telefono = value; }
    }

    public string FechaDeNacimiento
    {
        get { return fechaDeNacimiento; }
        set { fechaDeNacimiento = value; }
    }

    /*Creo el constructor para pasarle estos campos del cliente a la clase
    Cuentas*/
    public Cliente(string cNombre, string cApellido, string cTipoDeDocumento, long
cNumeroDocumento, string cDireccion, string cNacionalidad, string cOcupacion, string
cCorreoEletronico, long cTelefono, string cFechaDeNacimiento)
    {
        Nombre = cNombre;
        Apellido = cApellido;
        TipoDeDocumento = cTipoDeDocumento;
        NumeroDeDocumento = cNumeroDocumento;
        Direccion = cDireccion;
        Nacionalidad = cNacionalidad;
        Ocupacion = cOcupacion;
        CorreoElectronico = cCorreoEletronico;
        Telefono = cTelefono;
        FechaDeNacimiento = cFechaDeNacimiento;
    }

    //Declaro Metodo: DescuentoPorMovimientos
    public double DescuentoPorMovimientos { get; set; }
}
}

```

Clase Cuenta:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace BLL
{
    /*Cuenta sera una clase Asociacion
    La clase Cuenta va a HEREDAR de la clase cliente esto lo hago con los ':'
    No pode heredar de la clase movimientos ya que en C# solo se permite que
    se herede de una SOLA clase.*/
    public abstract class Cuenta : Cliente
    {

```

```

        //Declaro los atributos
        private double saldoContable;
        private int numeroCuenta;
        private long numeroCbu;
        private string moneda;
        private double movimientosPendientes;
        private string denominacion;
        private int cantidadTitulares;

        //Con este constructor tomare los atributos de la clase Cliente
        protected Cuenta(string cNombre, string cApellido, string cTipoDeDocumento, long
cNumeroDocumento, string cDireccion, string cNacionalidad, string cOcupacion, string
cCorreoEletronico, long cTelefono, string cFechaDeNacimiento)
            : base(cNombre, cApellido, cTipoDeDocumento, cNumeroDocumento, cDireccion,
cNacionalidad, cOcupacion, cCorreoEletronico, cTelefono, cFechaDeNacimiento)
        {

        }

        //Declaro los metodos Abstractos
        public abstract double Depositar { get; set; }
        public abstract double Extraer { get; set; }
        public abstract double Transferir { get; set; }
        public abstract double Pagar { get; set; }

    }
}

```

Clase Movimiento:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace BLL
{
    public abstract class Movimiento : Cuenta
    {
        //Declaro los atributos
        private string tipoTarjeta;
        private long numeroTarjeta;
        private int codSeguridadTarjeta;
        private string fechaVencimiento;
        private string marcaTarjeta;
        private long monto;
        public string tipoMovimiento;

        protected Movimiento(string cNombre, string cApellido, string cTipoDeDocumento, long
cNumeroDocumento, string cDireccion, string cNacionalidad, string cOcupacion, string
cCorreoEletronico, long cTelefono, string cFechaDeNacimiento) : base(cNombre, cApellido,
cTipoDeDocumento, cNumeroDocumento, cDireccion, cNacionalidad, cOcupacion,
cCorreoEletronico, cTelefono, cFechaDeNacimiento)
        {
        }

        //Declaro la propiedad Monto:
        public long Monto
        {
            get { return monto; }
            set { monto = value; }
        }

        //Declaro los metodos que seran tambien abstractos
    }
}

```

```

        public abstract double realizarMovimiento { get; set; }
        public abstract double calcularMovimiento { get; set; }
        public abstract double consultarSaldo { get; set; }
        public abstract double AplicarDescuento { get; set; }
    }
}

```

Clase CategoriaOro:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace BLL
{
    internal class CategoriaOroDescuento : Cliente
    {
        //Declaro Atributo
        float aplicarDescuento = 0.03f;

        //Declaro Propiedad
        public float AplicarDescuento
        {
            get { return aplicarDescuento; }
            set { aplicarDescuento = value; }
        }

        public CategoriaOroDescuento(string cNombre, string cApellido, string
cTipoDeDocumento, long cNumeroDocumento, string cDireccion, string cNacionalidad, string
cOcupacion, string cCorreoEletronico, long cTelefono, string cFechaDeNacimiento) :
base(cNombre, cApellido, cTipoDeDocumento, cNumeroDocumento, cDireccion, cNacionalidad,
cOcupacion, cCorreoEletronico, cTelefono, cFechaDeNacimiento)
        {
        }
    }
}

```

Clase CategoriaPlata:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace BLL
{
    internal class CategoriaPlatino : Cliente
    {
        //Declaro Atributo
        float aplicarDescuento = 0.05f;

        //Declaro Propiedad
        public float AplicarDescuento
        {
            get { return aplicarDescuento; }
            set { aplicarDescuento = value; }
        }

        public CategoriaPlatino(string cNombre, string cApellido, string cTipoDeDocumento,
long cNumeroDocumento, string cDireccion, string cNacionalidad, string cOcupacion, string
cCorreoEletronico, long cTelefono, string cFechaDeNacimiento) : base(cNombre, cApellido,

```



```

cTipoDeDocumento, cNumeroDocumento, cDireccion, cNacionalidad, cOcupacion,
cCorreoEletronico, cTelefono, cFechaDeNacimiento)
    {
    }
}
}

```

Clase CategoriaPlata:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace BLL
{
    internal class CategoriaPlata : Cliente
    {
        //Declaro Atributo
        float aplicarDescuento = 0.05f;

        //Declaro Propiedad
        public float AplicarDescuento
        {
            get { return aplicarDescuento; }
            set { aplicarDescuento = value; }
        }

        public CategoriaPlata(string cNombre, string cApellido, string cTipoDeDocumento,
long cNumeroDocumento, string cDireccion, string cNacionalidad, string cOcupacion, string
cCorreoEletronico, long cTelefono, string cFechaDeNacimiento) : base(cNombre, cApellido,
cTipoDeDocumento, cNumeroDocumento, cDireccion, cNacionalidad, cOcupacion,
cCorreoEletronico, cTelefono, cFechaDeNacimiento)
        {
        }
    }
}

```

Clase Caja de ahorro:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace BLL
{
    class CajaDeAhorro : Cuenta
    {
        public CajaDeAhorro(string cNombre, string cApellido, string cTipoDeDocumento, long
cNumeroDocumento, string cDireccion, string cNacionalidad, string cOcupacion, string
cCorreoEletronico, long cTelefono, string cFechaDeNacimiento) : base(cNombre, cApellido,
cTipoDeDocumento, cNumeroDocumento, cDireccion, cNacionalidad, cOcupacion,
cCorreoEletronico, cTelefono, cFechaDeNacimiento)
        {
        }

        public override double Depositar { get => throw new NotImplementedException(); set
=> throw new NotImplementedException(); }
        public override double Extraer { get => throw new NotImplementedException(); set =>
throw new NotImplementedException(); }
    }
}

```

```

        public override double Transferir { get => throw new NotImplementedException(); set
=> throw new NotImplementedException(); }
        public override double Pagar { get => throw new NotImplementedException(); set =>
throw new NotImplementedException(); }
    }
}

```

Clase CuentaCorriente:

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

```

```
namespace BLL
```

```

{
    class CuentaCorriente : Cuenta
    {
        public CuentaCorriente(string cNombre, string cApellido, string cTipoDeDocumento,
long cNumeroDocumento, string cDireccion, string cNacionalidad, string cOcupacion, string
cCorreoEletronico, long cTelefono, string cFechaDeNacimiento)
            : base(cNombre, cApellido, cTipoDeDocumento, cNumeroDocumento, cDireccion,
cNacionalidad, cOcupacion, cCorreoEletronico, cTelefono, cFechaDeNacimiento)
        {

        }

        public override double Depositar { get => throw new NotImplementedException(); set
=> throw new NotImplementedException(); }
        public override double Extraer { get => throw new NotImplementedException(); set =>
throw new NotImplementedException(); }
        public override double Transferir { get => throw new NotImplementedException(); set
=> throw new NotImplementedException(); }
        public override double Pagar { get => throw new NotImplementedException(); set =>
throw new NotImplementedException(); }
    }
}

```

Clase CuentaClaseBase

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

```

```
namespace BLL
```

```

{
    public class CuentaClaseBase : Cuenta
    {

        public CuentaClaseBase(string cNombre, string cApellido, string cTipoDeDocumento,
long cNumeroDocumento, string cDireccion, string cNacionalidad, string cOcupacion, string
cCorreoEletronico, long cTelefono, string cFechaDeNacimiento) : base(cNombre, cApellido,
cTipoDeDocumento, cNumeroDocumento, cDireccion, cNacionalidad, cOcupacion,
cCorreoEletronico, cTelefono, cFechaDeNacimiento)
        {

        }

        public override double Depositar { get => throw new NotImplementedException(); set
=> throw new NotImplementedException(); }
        public override double Extraer { get => throw new NotImplementedException(); set =>
throw new NotImplementedException(); }
        public override double Transferir { get => throw new NotImplementedException(); set
=> throw new NotImplementedException(); }
    }
}

```

```

        public override double Pagar { get => throw new NotImplementedException(); set =>
throw new NotImplementedException(); }
    }
}

```

Clase Movimiento ClaseBase

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace BLL
{
    public class MovimientoClaseBase : Movimiento
    {

        public MovimientoClaseBase(string cNombre, string cApellido, string
cTipoDeDocumento, long cNumeroDocumento, string cDireccion, string cNacionalidad, string
cOcupacion, string cCorreoEletronico, long cTelefono, string cFechaDeNacimiento) :
base(cNombre, cApellido, cTipoDeDocumento, cNumeroDocumento, cDireccion, cNacionalidad,
cOcupacion, cCorreoEletronico, cTelefono, cFechaDeNacimiento)
        {
        }

        public override double realizarMovimiento { get => throw new
NotImplementedException(); set => throw new NotImplementedException(); }
        public override double calcularMovimiento { get => throw new
NotImplementedException(); set => throw new NotImplementedException(); }
        public override double consultarSaldo { get => throw new NotImplementedException();
set => throw new NotImplementedException(); }
        public override double AplicarDescuento { get => throw new
NotImplementedException(); set => throw new NotImplementedException(); }
        public override double Depositar { get => throw new NotImplementedException(); set
=> throw new NotImplementedException(); }
        public override double Extraer { get => throw new NotImplementedException(); set =>
throw new NotImplementedException(); }
        public override double Transferir { get => throw new NotImplementedException(); set
=> throw new NotImplementedException(); }
        public override double Pagar { get => throw new NotImplementedException(); set =>
throw new NotImplementedException(); }
    }
}

```

Clase TarjetaDeDebito

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace BLL
{
    class TarjetaDebito : Movimiento
    {
        public TarjetaDebito(string cNombre, string cApellido, string cTipoDeDocumento, long
cNumeroDocumento, string cDireccion, string cNacionalidad, string cOcupacion, string
cCorreoEletronico, long cTelefono, string cFechaDeNacimiento) : base(cNombre, cApellido,
cTipoDeDocumento, cNumeroDocumento, cDireccion, cNacionalidad, cOcupacion,
cCorreoEletronico, cTelefono, cFechaDeNacimiento)
        {
        }
    }
}

```

```

        public override double realizarMovimiento { get => throw new
NotImplementedException(); set => throw new NotImplementedException(); }
        public override double calcularMovimiento { get => throw new
NotImplementedException(); set => throw new NotImplementedException(); }
        public override double consultarSaldo { get => throw new NotImplementedException();
set => throw new NotImplementedException(); }
        public override double AplicarDescuento { get => throw new
NotImplementedException(); set => throw new NotImplementedException(); }
        public override double Depositar { get => throw new NotImplementedException(); set
=> throw new NotImplementedException(); }
        public override double Extraer { get => throw new NotImplementedException(); set =>
throw new NotImplementedException(); }
        public override double Transferir { get => throw new NotImplementedException(); set
=> throw new NotImplementedException(); }
        public override double Pagar { get => throw new NotImplementedException(); set =>
throw new NotImplementedException(); }
    }
}

```

Clase TarjetaDeCredito

```

using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace BLL
{
    class TarjetaCredito : Movimiento
    {
        public TarjetaCredito(string cNombre, string cApellido, string cTipoDeDocumento,
long cNumeroDocumento, string cDireccion, string cNacionalidad, string cOcupacion, string
cCorreoEletronico, long cTelefono, string cFechaDeNacimiento) : base(cNombre, cApellido,
cTipoDeDocumento, cNumeroDocumento, cDireccion, cNacionalidad, cOcupacion,
cCorreoEletronico, cTelefono, cFechaDeNacimiento)
        {
        }

        public override double realizarMovimiento { get => throw new
NotImplementedException(); set => throw new NotImplementedException(); }
        public override double calcularMovimiento { get => throw new
NotImplementedException(); set => throw new NotImplementedException(); }
        public override double consultarSaldo { get => throw new NotImplementedException();
set => throw new NotImplementedException(); }
        public override double AplicarDescuento { get => throw new
NotImplementedException(); set => throw new NotImplementedException(); }
        public override double Depositar { get => throw new NotImplementedException(); set
=> throw new NotImplementedException(); }
        public override double Extraer { get => throw new NotImplementedException(); set =>
throw new NotImplementedException(); }
        public override double Transferir { get => throw new NotImplementedException(); set
=> throw new NotImplementedException(); }
        public override double Pagar { get => throw new NotImplementedException(); set =>
throw new NotImplementedException(); }
    }
}

```