


```
}
```

```
//En clase ITEM
```

```
public decimal CalcularPrecioItem()
{
    return _cant * _producto.CalcularPrecio();
}
```

```
//En clase Cliente
```

```
//Podria recibir solamente la FormaPago
```

```
public abstract decimal ObtenerDescuento(Factura f);
```

```
public abstract string GetTipo();
```

```
//En Clase Ocasional
```

```
//Podria recibir solamente la FormaPago
```

```
public override decimal ObtenerDescuento(Factura f)
{
    decimal descuento = 0;
    if(f.FormaPago == FormaPago.Credito) descuento = Ocasional.DescCredito;
    return descuento;

    //Con operador condicional ternario
    // return f.FormaPago == FormaPago.Credito ? Ocasional.DescCredito : 0;
}
```

```
public override string GetTipo()
{
    return "Ocasional";
}
```

```
//En Clase Empresarial
```

```
//Podria recibir solamente la FormaPago
```

```
public override decimal ObtenerDescuento(Factura f)
{
    decimal descuento = 0;
    if(f.FormaPago == FormaPago.Contado) descuento = this._descContado;
    return descuento;

    //Con operador condicional ternario
    //return f.FormaPago == FormaPago.Contado ? this._descContado : 0;
}
```

```
public override string GetTipo()
{
    return "Empresarial";
}
```

```
//En clase Factura
```

```

public decimal CalcularTotalAPagar()
{
    decimal total = 0;
    foreach(Item i in _items)
    {
        total += i.CalcularPrecioItem();
    }

    decimal descuentoCliente = _cliente.ObtenerDescuento(this);
    total -= (total * descuentoCliente / 100);

    int diasVencimiento = (_vencimiento - _fecha).Days;
    if(_formaPago == FormaPago.Credito && diasVencimiento > 30) total += (total *
Factura.RecargoPorVencimiento / 100);

    return total;
}

public override ToString()
{
    return $"Nro Factura: {_numero} - Cliente: {_cliente.Nombre} - Tipo: {_cliente.GetTipo()} - Total facturado:
{CalcularTotalAPagar()}";
}

//En clase Sistema

public List<Factura> VentasPorFecha(DateTime d)
{
    List<Factura> listado = new List<Factura>();
    foreach(Factura f in _facturas)
    {
        if(f.Fecha.Date == d.Date) listado.Add(f);
    }

    return listado;
}

//PARTE B

//En Clase Cliente
public override bool Equals(object obj)
{
    Cliente c = obj as Cliente;
    return c != null && this._id.Equals(c.Id);
}

//En Clase Factura : IComparable<Factura>

public int CompareTo(Factura f)
{
    return this._numero.CompareTo(f._numero);
}

public string MostrarDatosRequerimientoB()
{

```

```
return $"Nro Factura: {_numero} - Fecha: {_fecha.ToShortDateString()} - Forma Pago: {_formaPago}";  
}
```

//En Sistema

```
public List<Factura> VentasPorClienteOrdenadas(Cliente c)  
{  
    List<Factura> listado = new List<Factura>();  
    foreach(Factura f in _facturas)  
    {  
        if(f.Cliente.Equals(c)) listado.Add(f);  
    }  
  
    listado.Sort();  
    return listado;  
}
```

//MVC

//Indique como guardar el valor "admin" en dicha variable de session

```
HttpContext.Session.SetString("rol", "admin");
```

//Indique como recupera el valor de la variable de session "rol"

```
HttpContext.Session.GetString("rol");
```