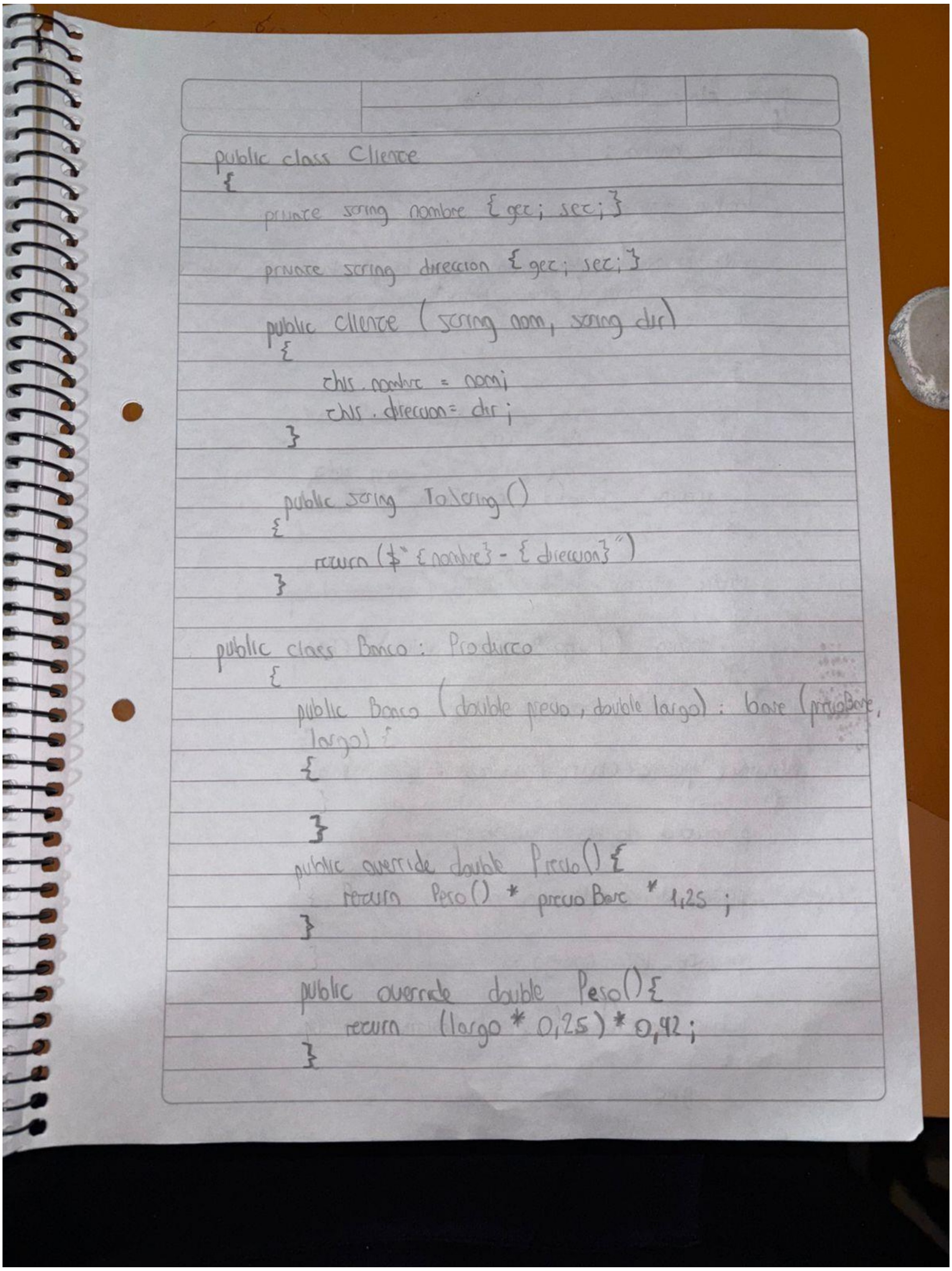
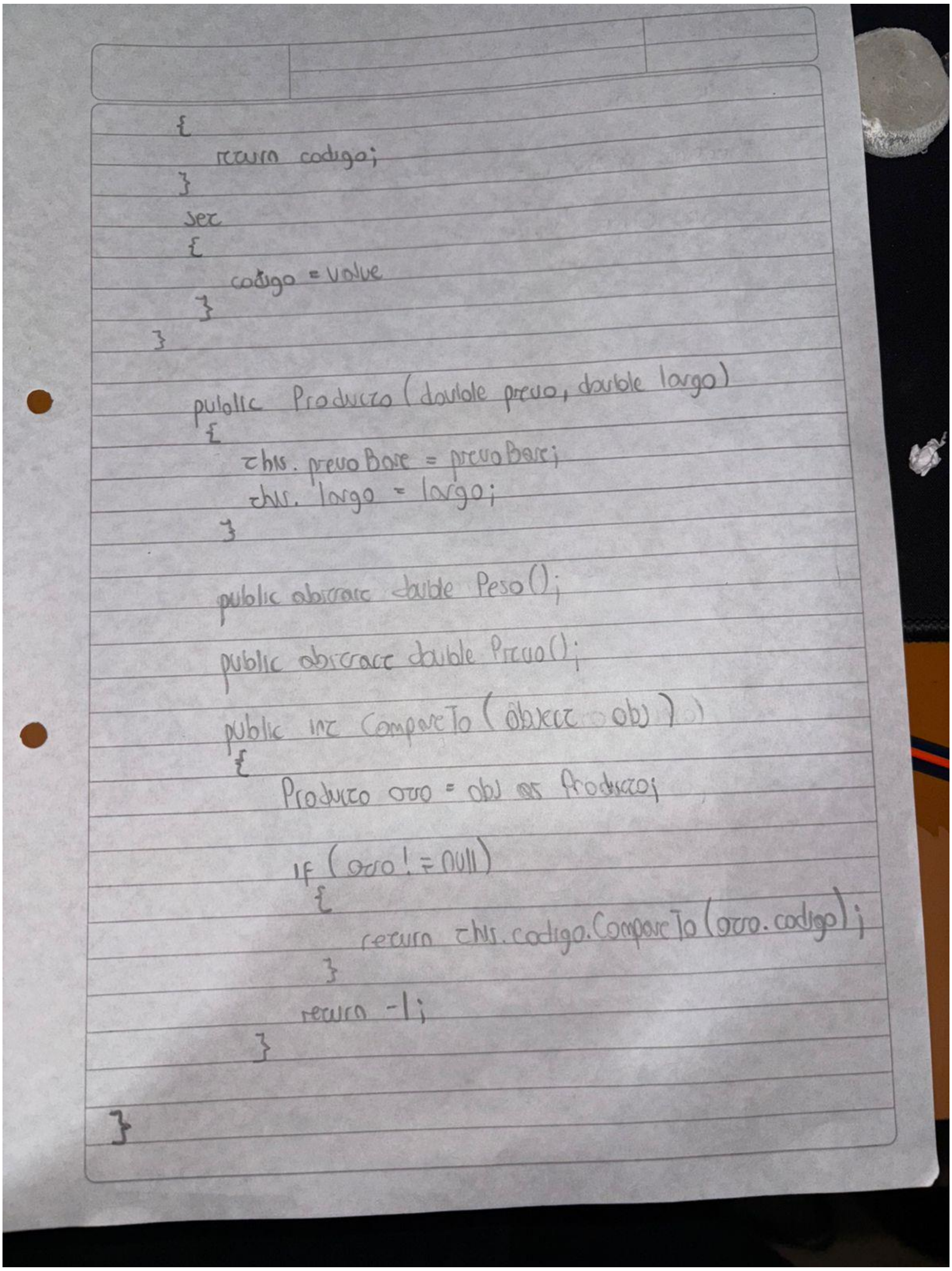
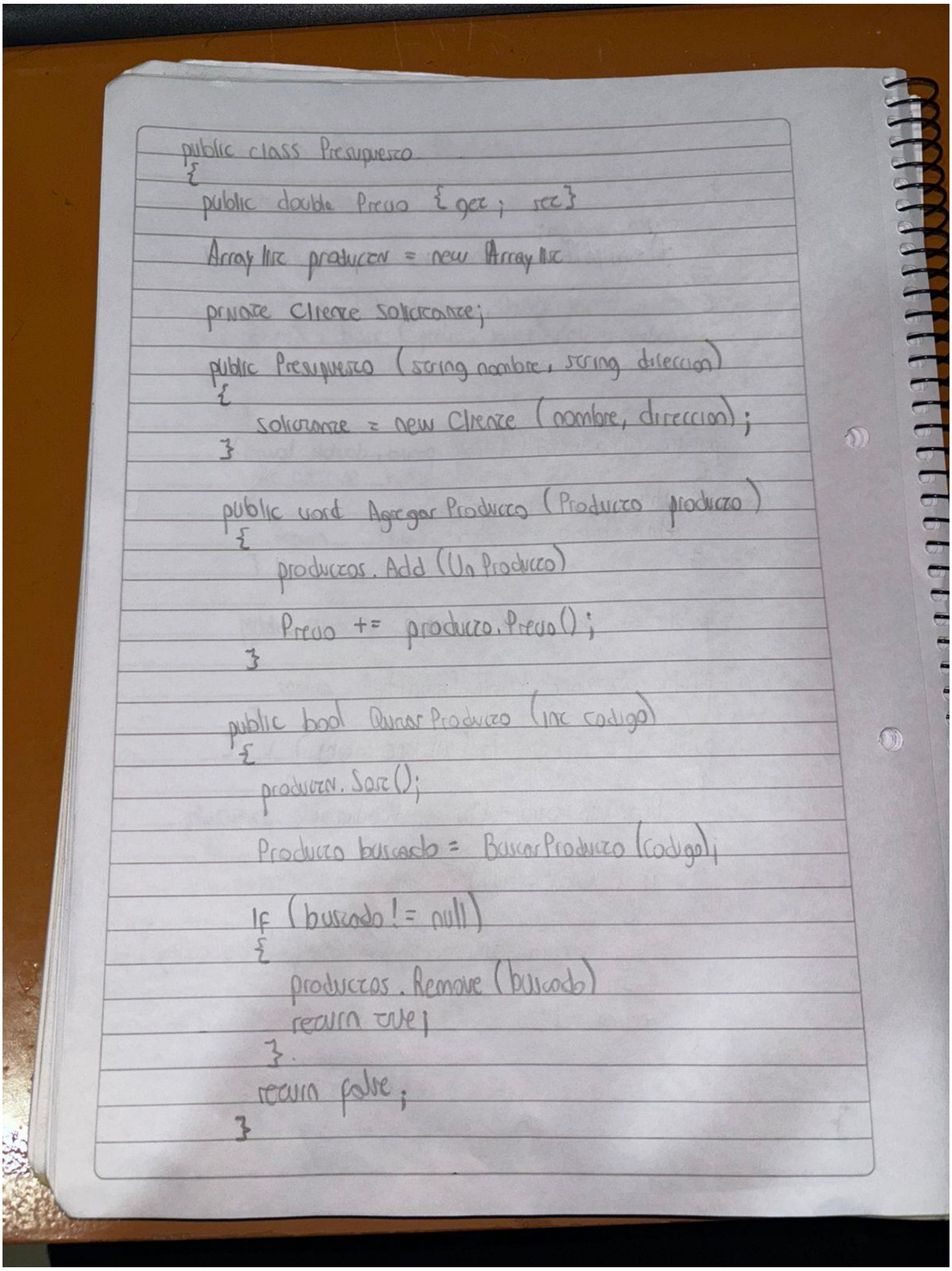


```
public class Cliente
{
    private string nombre {get; set;}

    private string direccion {get; set;}

    public Cliente (string nom, string dir)
     {
        this.nombre = nom;
        this.direccion= dir;
     }

     public string ToString()
    {
        return ($"{nombre} - {direccion}")
    }

public class Banco : Producto
    {
      public Banco (double precio, double largo) : base (precioBase,
      largo)
      {

      }
      public override double Precio() {
          return Peso() * precioBase * 1,25 ;
      }

      public override double Peso(){
          return (largo * 0,25) * 0,42 ;
      }
```

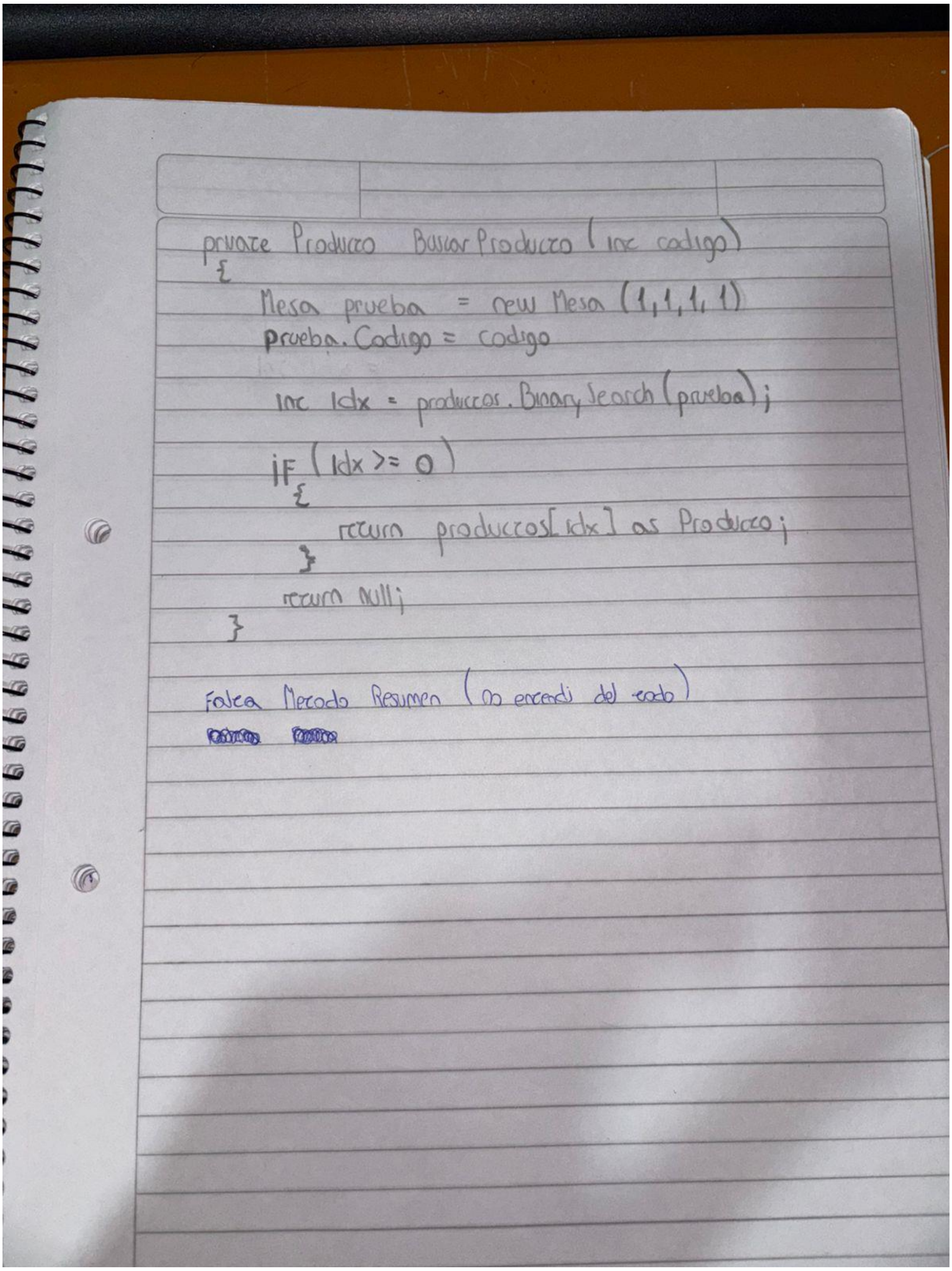
```
public class Mesa : Producto
{
  double ancho ;

  double grosor;

   public Mesa (double precio, double largo, double ancho, double
    grosor) : base ( precio, base)
    {

    }

    public override double Precio ()
    {
       return   Peso() * precioBase * 1,25
    }

    public override double Peso()
    {
       return  (largo * ancho * grosor) * 0,3 ;
    }
  }

abstract public class Producto : IComparable
 {
     protected double precioBase;

     protected double largo;

     private int codigo ;

     public int Codigo
      {
         get
```



```
    {
        return codigo;
    }
    set
    {
        codigo = value
    }
}

    public Producto (double precio, double largo)
    {
        this.precioBase = precioBase;
        this.largo = largo;
    }

    public abstract double Peso();

    public abstract double Precio();

    public int CompareTo ( object obj )
    {
        Producto otro = obj as Producto;

        if (otro != null)
        {
            return this.codigo.CompareTo(otro.codigo);
        }
        return -1;
    }

}
```



```
public class Presupuesto
{
 public double Precio { get ; set}

 Array list productos = new Array list

 private Cliente solicitante;

 public Presupuesto (string nombre, string direccion)
 {
    solicitante = new Cliente (nombre, direccion);
 }

  public void Agregar Producto (Producto producto)
  {
     productos. Add (Un Producto)

     Precio += producto. Precio();
  }

  public bool Quitar Producto (int codigo)
  {
    productos. Sort();

    Producto buscado = Buscar Producto (codigo);

    If (buscado != null)
    {
      productos . Remove (buscado)
      return true;
    }.
    return false;
  }
```



```
private Producto BuscarProducto (int codigo)
 {
     Mesa prueba = new Mesa (1,1,1,1)
     prueba.Codigo = codigo

     int idx = productos.BinarySearch(prueba);

     if (idx >= 0)
      {
          return productos[idx] as Producto;
      }
     return null;
 }
```

Falta Metodo Resumen (no entendi del todo)