

## class Presupuesto

```
{ public double Precio {get;set;}  
  private Cliente Solicitante;  
  private List<Productos> lista = new List<Productos>();  
  
  public Presupuesto (string nombre, string Direccion)  
  {  
    solicitante = new Cliente (nombre, direccion);  
  }  
  
  public void AgregarProducto (Producto unProducto)  
  {  
    lista.Add(unProducto);  
  }  
  
  private Producto BuscarProducto (int codigo)  
  {  
    Producto buscado = new Producto (0, 0);  
    buscado.Codigo = codigo;  
    lista.Sort();  
    if (lista.BinarySearch(buscado) < 0)  
    {  
      return null;  
    }  
    else {  
      int pos = lista.BinarySearch(buscado);  
      return lista[pos];  
    }  
  }  
  
  public bool QuitarProducto (int codigo)  
  {  
    Producto borrar = new Producto (0, 0);  
    if (borrar.BuscarProducto (codigo) != null)  
    {  
      lista.Remove (borrar);  
      return true;  
    }  
    return false;  
  }  
  
  public String[] Resumen()  
  {  
    String[] resultado = new String [lista.Count + 2];  
    int n = 0; resultado[n++] = solicitante.ToString();
```

Resumen no entendi

```
foreach (Producto p in lista)
{ resultado [n++] = $"Codigo {p.codigo} - Precio {p.precio} - Precio {p.precio}";
  resultado [n++] = $"Total: {p.precio}";
  return resultado;
}
```

class Cliente

```
{ private string nombre;
  private string direccion;
```

public Cliente (string nombre, string dir)

```
{ this.nombre = nombre;
  this.direccion = dir; }
```

public override ToString()

```
{ return $"Nombre - {this.nombre} - Direccion {this.direccion}"; }
```

class abstract Producto : IComparable

```
{ protected double PrecioBase;
  protected double Largo;
```

private int codigo;

```
public int Codigo {get; set; }
```

public Producto (double precio, double largo)

```
{ this.PrecioBase = precio;
  this.Largo = largo; }
```

public abstract double Precio();

public abstract double Largo();

public int CompareTo (Object obj)

```
{ Producto p = obj as Producto;
```

```
if (p != null)
```

```
{ return p.Codigo.CompareTo (p.Codigo); }
```

```
return 0; }
```



Abstract Method Override

Class Banco : Producto

```
{ public Banco (double precio, double largo) : base (precio, largo);  
  public override double Precio ()  
  { double precio = peso * this.precioBase * 1,15;  
    return precio; } }
```

Public <sup>override</sup> double Peso ()

```
{ double return (this.largo * 0,25) * 0,42; } }
```

class Mesa : Producto

```
{ private double ancho;  
  private double grosor;
```

```
public Mesa (double p, double L, double A, double g)  
{  
  : base (precio, largo)  
  { this.ancho = A;  
    this.grosor = g; } }
```

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

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

Form 1 Presupuesto presupuesto;

### Boton Agregar

```
{ double precio = (TbPrecio.Text).Convert.ToDouble;  
  double Ancho = Convert.ToDouble(TbAncho.Text);  
  double Largo = Convert.ToDouble(TbLargo.Text);  
  double grosor = Convert.ToDouble(TbGrosor.Text);  
  int int codigo = Convert.ToInt32(TbCodigo.Text);  
  
  if (string.IsNullOrEmpty(TbAnchoText && TbGrosor.Text))  
  { Producto nuevo = new BANCO(precio, largo);  
    nuevo.Codigo = codigo;  
    lista.Add(nuevo); }  
  else { Producto nuevo = new Mesa(precio, largo, ancho, grosor);  
    nuevo.Codigo = codigo;  
    lista.Add(nuevo); }  
  else { MessageBox.Show("Faltan completar campos"); }  
  if (nuevo != null)  
    comboBox1.Items.Add(codigonuevo);
```

### Boton Iniciar Presupuesto

```
{ string nombre = TbNombre.Text;  
  string dire = TbDir.Text; Presupuesto  
Presupuesto Presupuesto nuevo = new Codigo (nombre, dire); }
```

### Boton Agregar

```
{ if (presupuesto != null) // inicializar a vera para usarlo en este Boton  
  { ListProductosProductos pedido = new List<Productos>(1);  
    pedido pedido.Items.Add(comboBox1.SelectedItem); }
```

### Boton Borrar

```
{ pedido.Remove(comboBox1.SelectedItem); if (comboBox1.SelectedItem != null)  
  if (comboBox1.SelectedItem != null)  
  { Quitar Producto (Convert.ToInt32(TbCodigo.Text)); }  
  else { }  
}
```



### Boton BORRAR

```
{ if (ComboBox1.Selected Item != Null) // si es un objeto  
  { pedido.Remove(ComboBox1.Selected Item); }
```

```
if (String.IsNullOrEmpty(IDCodigo) & Null)  
{ Crear = new Producto
```

```
{ int cod = Conver.to Int 32 (ComboBox1.Selected Item);  
  Presupuesto  
  pedido.QuitarProducto(cod); }
```

### Boton Cerrar Presupuesto

```
{ MessageBox.Show("Presupuesto finalizado"); double precio total Precio  
  foreach (Producto p in pedido)
```

```
double precio total
```

```
{ double precio total Precio + = p.Precio(); }
```

```
String[] pedido = new String[] presupuesto.Resumen(); }
```