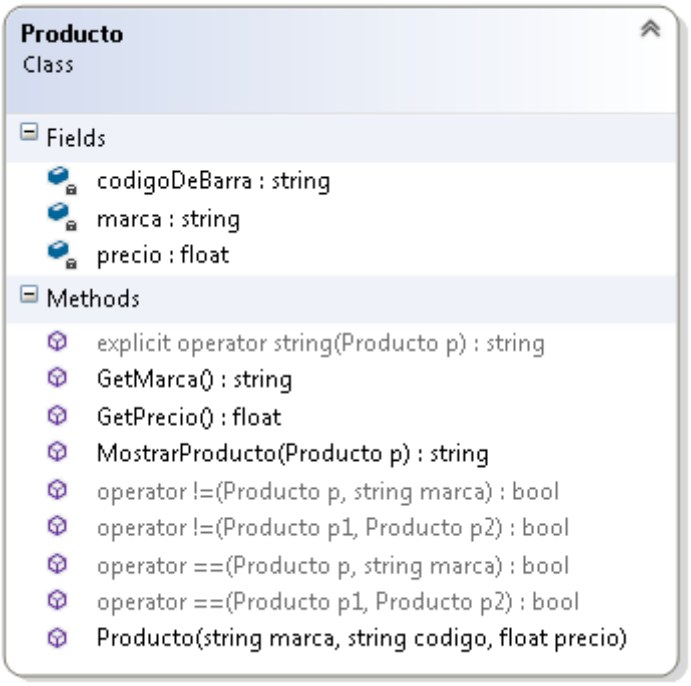
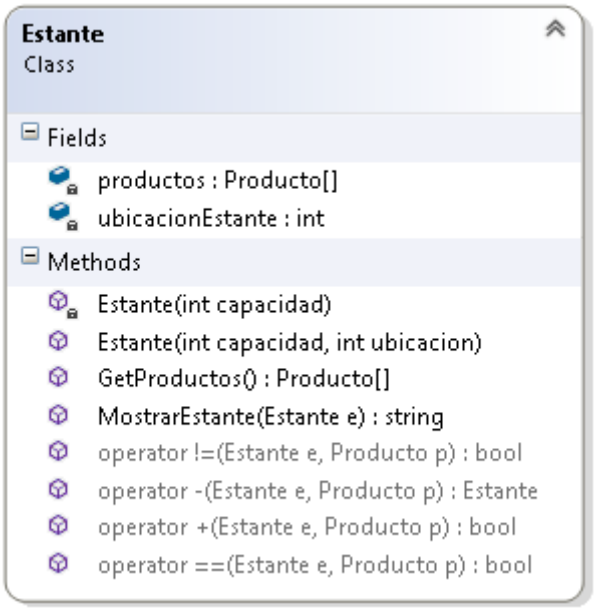
Generar una Solución llamada Repaso la cual contendrá un proyecto cuya clase base **Producto**.



1) Todos sus atributos son protegidos. Posee **sólo** un constructor de instancia. El método **GetMarca**, retornará el valor correspondiente del atributo *marca : string*. El método **GetPrecio**, retornará el valor asociado al atributo *precio : float*. También poseerá el atributo *codigoDeBarras : string*, el cual se publicará sólo a través de la comversión explísita nombrada más adelante.
2) El método de **clase MostrarProducto(Producto)**, es público y retornará una cadena detallando los atributos de la clase.
3) La clase Producto posee sobrecarga de operadores:
   Explícito. Retornará el código de barras del producto que recibe como parámetro.
   Igualdad (Producto, Producto). Retornará *true*, si las marcas y códigos de barras son iguales, *false*, caso contrario.
   Igualdad (Producto, string). Retornará *true*, si la marca del producto coincide con la cadena pasada por parámetro, *false*, caso contrario.

Generar la clase **Estante**. La misma posee dos atributos privados. Uno será un entero que indicará la ubicación del estante y el otro es un array de tipo Producto.



4) El constructor de instancia **privado** será el **único** que inicializará el array. La sobrecarga pública del constructor inicializará la capacidad del estante, recibiendo como parámetro capacidad y ubicación. Reutilizar código.
5) El método público GetProductos, retornará el valor asociado del atributo *productos*.
6) El método público de **clase MostrarEstante**, retornará una cadena con toda la información del estante, incluyendo el detalle de cada uno de sus productos. Reutilizar código.

7) Sobrecarga de operadores:
Igualdad, retornará *true*, si es que el producto ya se encuentra en el estante, *false*, caso contrario.
Adición, retornará *true* y agregará el producto si el estante posee capacidad de almacenar al menos un producto más y dicho producto no se encuentra en él; *false*, caso contrario. Reutilizar código.
Sustracción (Estante, Producto), retornará un estante sin el producto, siempre y cuando el producto se encuentre en el listado. Reutilizar código.

8) Agregar un Main en una clase llamada **TestEstante** con el siguiente código:

```
// Creo un estante
Estante estante = new Estante(3, 1);
// Creo 4 productos
Producto p1 = new Producto("Pepsi", "PESDS97413", (float)10.5);
Producto p2 = new Producto("Coca-Cola", "COSDS55752", (float)10.5);
Producto p3 = new Producto("Manaos", "MASDS51292", (float)10.5);
Producto p4 = new Producto("Crush", "CRSDS54861", (float)10.5);

// Agrego los productos al estante
if (estante + p1)
{
    Console.WriteLine("Agregó {0} {1} {2}", p1.GetMarca(), (string)p1, p1.GetPrecio());
}
else
{
    Console.WriteLine("¡NO agregó {0} {1} {2}!", p1.GetMarca(), (string)p1, p1.GetPrecio());
}
if (estante + p1)
{
    Console.WriteLine("Agregó {0} {1} {2}", p1.GetMarca(), (string)p1, p1.GetPrecio());
}
else
{
    Console.WriteLine("¡NO agregó {0} {1} {2}!", p1.GetMarca(), (string)p1, p1.GetPrecio());
}
if (estante + p2)
{
    Console.WriteLine("Agregó {0} {1} {2}", p2.GetMarca(), (string)p2, p2.GetPrecio());
}
else
{
    Console.WriteLine("¡NO agregó {0} {1} {2}!", p2.GetMarca(), (string)p2, p2.GetPrecio());
}
if (estante + p3)
{
    Console.WriteLine("Agregó {0} {1} {2}", p3.GetMarca(), (string)p3, p3.GetPrecio());
}
else
{
    Console.WriteLine("¡NO agregó {0} {1} {2}!", p3.GetMarca(), (string)p3, p3.GetPrecio());
}
if (estante + p4)
{
    Console.WriteLine("Agregó {0} {1} {2}", p4.GetMarca(), (string)p4, p4.GetPrecio());
}
else
{
    Console.WriteLine("¡NO agregó {0} {1} {2}!", p4.GetMarca(), (string)p4, p4.GetPrecio());
}

// Muestro todo el estante
Console.WriteLine();
Console.WriteLine("<---------------------------------------------->");
Console.WriteLine(Estante.MostrarEstante(estante));

Console.ReadKey();
```