Problem

```
public class LegacyPrinter
{
    public void PrintDocument(string document)
    {
        Console.WriteLine($"Legacy Printer is printing: {document}");
    }
}
```

Rozwiązanie

Drukowanie wielu kopii dokumentu

- Tworzymy **Adapter** starej drukarki

```
// Adapter
public class LegacyPrinterAdapter
{
    // Adaptee
    private LegacyPrinter legacyPrinter = new LegacyPrinter();

    public void PrintDocument(string document, byte copies)
    {
        for (int i = 0; i < copies; i++)
        {
            legacyPrinter.PrintDocument(document);
        }
    }
}
```

Obliczanie kosztu wydruku

- Tworzymy interfejs

```
// Abstract
public interface IPrinter
{
```

```
    void PrintDocument(string document, byte copies);
}
```

- Tworzymy **dekorator** drukarki

```
// Decorator
public class PrinterWithCostDecorator(IPrinter printer) : IPrinter
{
    private decimal pricePerChar = 0.11m;

    public void PrintDocument(string document, byte copies)
    {
        printer.PrintDocument(document, copies);

        var cost = document.Length * copies * pricePerChar;

        Console.WriteLine($"Total cost: {cost}");
    }
}
```

- Teraz możemy łatwo dodawać nowe dekoratory

```
public class PrinterWithCounterDecorator(IPrinter printer) : IPrinter
{
    public void PrintDocument(string document, byte copies)
    {
        printer.PrintDocument(document, copies);

        Console.WriteLine($"Counter: {copies}");
    }
}
```