

**TUGAS JURNAL  
KONSTRUKSI PERANGKAT LUNAK**

**Pertemuan 13**

**CLEAN CODE**



**Disusun Oleh :**

**Andera Singgih Pratama / 2211104007**

**SE-06-01**

**Asisten Praktikum:**

**Naufal El Kamil Aditya Pratama Rahman**

**Imelda**

**Dosen Pengampu :**

**Yudha Islami Sulistya, S.Kom., M.Cs.**

**PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK**

**FAKULTAS INFORMATIKA**

**TELKOM UNIVERSITY PURWOKERTO**

**2025**

**TUGAS JURNAL**

## 1. MEMBUAT PROJECT MODUL

- Copy salah satu folder tugas jurnal yang dimiliki sebelumnya (dari modul 2 sampai modul 13), kemudian rename folder hasil copy-paste tersebut dengan modul14\_NIM (coba pilih tugas pendahuluan yang paling sederhana)
- Misalnya menggunakan Visual Studio, bukalah project/folder yang di-copy sebelumnya.

SAYA MEMILIH TUGAS PENDAHULUAN YANG TP 14

## 2. Screenshoot Hasil Run

Sebelum Refactoring

```
Application Output - modul14_2211104007
Hasil Tambah: 5
Hasil Kali: 6
```

Setelah Refactoring (Standar C# .NET)

```
Application Output - modul14_2211104007
Hasil Penjumlahan: 5
Hasil Perkalian: 6
```

## 3. Source Code: Sebelum dan Sesudah Refactoring

Sebelum Refactoring

Aljabar.cs

```
1 using System;
2
3 namespace AljabarApp
4 {
5     public class Aljabar
6     {
7         public int tambah(int x, int y)
8         {
9             return x + y;
10        }
11
12        public int kali(int x, int y)
13        {
14            return x * y;
15        }
16    }
17 }
```

## Program.cs:

```
using System;

namespace AljabarApp
{
    /// <summary>
    /// Kelas utama untuk menampilkan operasi aljabar.
    /// </summary>
    public class Program
    {
        public static void Main(string[] args)
        {
            Aljabar a = new Aljabar();

            Console.WriteLine("Hasil Penjumlahan: " + a.tambah(2, 3));
            Console.WriteLine("Hasil Perkalian: " + a.kali(2, 3));
        }
    }
}
```

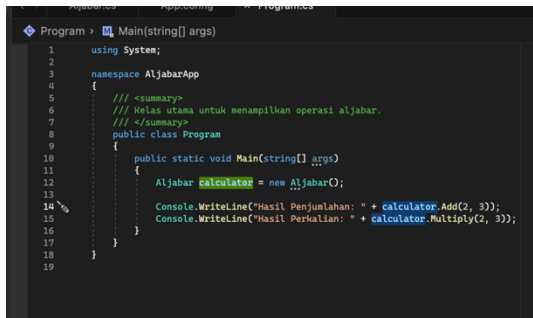
## Setelah Refactoring (Standar C# .NET) Aljabar.cs

```
using System;

namespace AljabarApp
{
    /// <summary>
    /// Berisi metode matematika dasar.
    /// </summary>
    public class Aljabar
    {
        /// <summary>
        /// Menambahkan dua bilangan bulat.
        /// </summary>
        public int Add(int number1, int number2)
        {
            return number1 + number2;
        }

        /// <summary>
        /// Mengalikan dua bilangan bulat.
        /// </summary>
        public int Multiply(int number1, int number2)
        {
            return number1 * number2;
        }
    }
}
```

## Program.cs

A screenshot of a code editor showing the Program.cs file. The code is written in C# and includes a namespace AljabarApp, a class Program, and a static Main method. The Main method creates an AljabarCalculator object and calls its Add and Multiply methods. The code is formatted with 4-space indentation and includes XML-style comments. Line numbers 1 through 19 are visible on the left side of the editor.

```
1  Program > Main(string[] args)
2  using System;
3
4  namespace AljabarApp
5  {
6      /// <summary>
7      /// Kelas utama untuk menampilkan operasi aljabar.
8      /// </summary>
9      public class Program
10     {
11         public static void Main(string[] args)
12         {
13             Aljabar calculator = new AljabarO();
14
15             Console.WriteLine("Hasil Penjumlahan: " + calculator.Add(2, 3));
16             Console.WriteLine("Hasil Perkalian: " + calculator.Multiply(2, 3));
17         }
18     }
19 }
```

## penjelasan:

Pada Jurnal Modul 14 ini, saya melakukan proses refactoring terhadap kode program C# sederhana yang sebelumnya telah dibuat pada modul sebelumnya. Refactoring dilakukan dengan mengacu pada standar penulisan kode .NET agar kode lebih rapi, mudah dibaca, dan sesuai praktik terbaik. Langkah pertama yang saya lakukan adalah memperbaiki naming convention pada variabel, method, dan objek. Misalnya, method tambah diubah menjadi Add, dan kali menjadi Multiply, agar sesuai dengan penamaan PascalCase yang umum digunakan dalam bahasa C#. Selain itu, variabel lokal seperti x dan y diganti menjadi number1 dan number2 agar lebih deskriptif dan mudah dipahami. Saya juga menata kembali white space dan indentation pada setiap baris kode agar struktur kode terlihat lebih teratur, menggunakan indentasi 4 spasi sesuai standar. Komentar dalam format XML (///) ditambahkan pada class dan setiap method untuk memberikan penjelasan mengenai fungsionalitasnya, sehingga dokumentasi kode menjadi lebih lengkap. Terakhir, saya melakukan uji coba run program sebelum dan sesudah refactoring, dan memastikan bahwa program tetap berjalan sesuai fungsinya, yaitu menampilkan hasil penjumlahan dan perkalian dua bilangan. Melalui refactoring ini, kualitas kode menjadi lebih baik dan lebih mudah untuk dikembangkan di masa mendatang.