

**JURNAL
KONSTRUKSI PERANGKAT LUNAK**

**PERTEMUAN 12
DESIGN PATTERN IMPLEMENTATION**



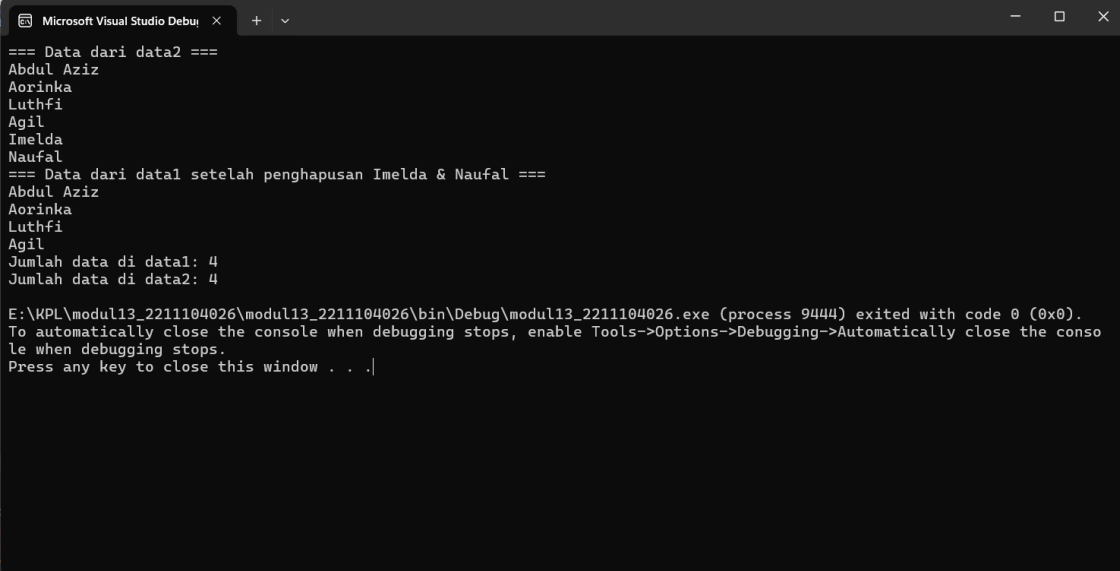
**Disusun Oleh :
Muhammad Abdul Aziz
2211104026
SE0601**

**Asisten Praktikum :
Naufal El Kamil Aditya Pratama Rahman
Imelda**

**Dosen Pengampu :
Yudha Islami Sulistya, S.Kom., M.Cs.**

**PROGRAM STUDI S1 SOFTWARE ENGINEERING
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY PURWOKERTO
2025**

1. Screenshot hasil run



```
Microsoft Visual Studio Debu x + v
=== Data dari data2 ===
Abdul Aziz
Aorinka
Luthfi
Agil
Imelda
Naufal
=== Data dari data1 setelah penghapusan Imelda & Naufal ===
Abdul Aziz
Aorinka
Luthfi
Agil
Jumlah data di data1: 4
Jumlah data di data2: 4

E:\KPL\modul13_2211104026\modul13_2211104026\bin\Debug\modul13_2211104026.exe (process 9444) exited with code 0 (0x0).
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.
Press any key to close this window . . .|
```

2. Penjelasan singkat dari kode implementasi yang dibuat (beserta screenshot dari potongan source code yang dijelaskan).

MENJELASKAN DESIGN PATTERN SINGLETON

A. Dua Contoh Penggunaan Singleton Pattern:

1. Koneksi ke Database: Dalam aplikasi, koneksi database sebaiknya hanya dibuat satu kali dan digunakan secara bersama-sama oleh seluruh sistem.
2. Logger: Sistem pencatatan log sering kali menggunakan Singleton agar seluruh bagian aplikasi menulis ke objek log yang sama.

B. Langkah-langkah Mengimplementasikan Singleton Pattern:

1. Buat class dengan konstruktor `private`, agar tidak bisa dibuat instance dari luar class.
2. Tambahkan atribut static bertipe class itu sendiri.
3. Buat method static (biasanya `GetInstance()` atau sejenis) yang mengembalikan instance tersebut, dan membuatnya jika belum ada.

C. Tiga Kelebihan dan Kekurangan Singleton:

Kelebihan	Kekurangan
1. Mengontrol akses ke satu-satunya instance.	1. Sulit untuk unit testing karena global state.
2. Mengurangi penggunaan memori karena hanya satu objek.	2. Menyebabkan tight coupling.
3. Cocok untuk resource sharing seperti konfigurasi global.	3. Tidak cocok untuk multithreading jika tidak hati-hati.

Source Code :

Class PusatDataSingelton

```
1  using System;
2  using System.Collections.Generic;
3
4  8 references
5  public class PusatDataSingleton
6  {
7      private static PusatDataSingleton _instance;
8      7 references
9      public List<string> DataTersimpan { get; private set; }
10
11      // Konstruktor private
12      1 reference
13      private PusatDataSingleton()
14      {
15          DataTersimpan = new List<string>();
16      }
17
18      // Method Singleton
19      2 references
20      public static PusatDataSingleton GetDataSingleton()
21      {
22          if (_instance == null)
23          {
24              _instance = new PusatDataSingleton();
25          }
26          return _instance;
27      }
28
29      // Mengembalikan semua data
30      4 references
31      public List<string> GetSemuaData()
32      {
33          return DataTersimpan;
34      }
```

Program Utama

```
30
31      // Menampilkan semua data
32      2 references
33      public void PrintSemuaData()
34      {
35          if (DataTersimpan.Count == 0)
36          {
37              Console.WriteLine("Data kosong.");
38          }
39          else
40          {
41              foreach (var data in DataTersimpan)
42              {
43                  Console.WriteLine(data);
44              }
45          }
46
47      // Menambahkan data baru
48      6 references
49      public void AddSebuahData(string input)
50      {
51          DataTersimpan.Add(input);
52      }
53
54      // Menghapus data berdasarkan index
55      2 references
56      public void HapusSebuahData(int index)
57      {
58          if (index >= 0 && index < DataTersimpan.Count)
59          {
60              DataTersimpan.RemoveAt(index);
61          }
62          else
63          {
64              Console.WriteLine("Index tidak valid!");
65          }
66      }
```

Penjelasan :

Program ini merupakan implementasi dari design pattern Singleton dalam bahasa C#. Tujuan utamanya adalah memastikan hanya ada satu objek dari class PusatDataSingleton yang digunakan bersama di seluruh program. Kelas PusatDataSingleton memiliki atribut DataTersimpan berupa list string yang menyimpan data, seperti nama anggota kelompok dan asisten praktikum. Pada method Main, dibuat dua variabel data1 dan data2 yang keduanya mengambil instance dari method GetDataSingleton(). Karena menggunakan Singleton, kedua variabel tersebut merujuk pada objek yang sama. Data anggota kelompok dan asisten praktikum ditambahkan melalui data1, lalu ditampilkan lewat data2. Selanjutnya, data asisten praktikum (“Imelda” dan “Naufal”) dihapus melalui data2. Ketika data ditampilkan kembali lewat data1, data yang terhapus tidak muncul lagi, membuktikan bahwa kedua variabel mengakses instance yang sama. Terakhir, program mencetak jumlah total data yang tersisa dari kedua variabel. Program ini menunjukkan penggunaan pola Singleton untuk berbagi data yang konsisten di seluruh bagian aplikasi.