

LAPORAN PRAKTIKUM MODUL 4



Disusun Oleh :

Zaenarif Putra 'Ainurdin – 2311104049

Kelas :

SE-07-02

Dosen :

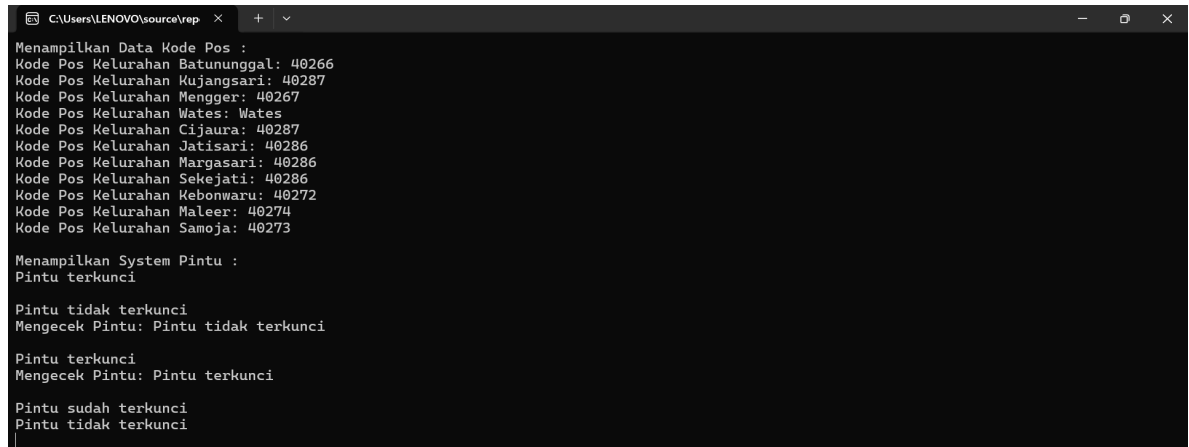
Yudha Islami Sulistya

**PROGRAM STUDI SOFTWARE ENGINEERING
DIREKTORAT KAMPUS PURWOKERTO
TELKOM UNIVERSITY
PURWOKERTO
2025**

I. Link Github

https://github.com/zaenarifputra/KPL_Zaenarif-Putra-Ainurdin_2311104049_S1SE-07-02/tree/5f9002e73ba8f850f6f32bf81935a3e32d5999c8/04_Automata_dan_Table-Driven_Construction/TP_Modul4

II. Hasil Running



```
C:\Users\LENOVO\source\rep x + v
Menampilkan Data Kode Pos :
Kode Pos Kelurahan Batununggal: 40266
Kode Pos Kelurahan Kujangsari: 40287
Kode Pos Kelurahan Mengger: 40267
Kode Pos Kelurahan Wates: Wates
Kode Pos Kelurahan Cijaura: 40287
Kode Pos Kelurahan Jatisari: 40286
Kode Pos Kelurahan Margasari: 40286
Kode Pos Kelurahan Sekejati: 40286
Kode Pos Kelurahan Kebonwaru: 40272
Kode Pos Kelurahan Maleer: 40274
Kode Pos Kelurahan Samoja: 40273

Menampilkan System Pintu :
Pintu terkunci

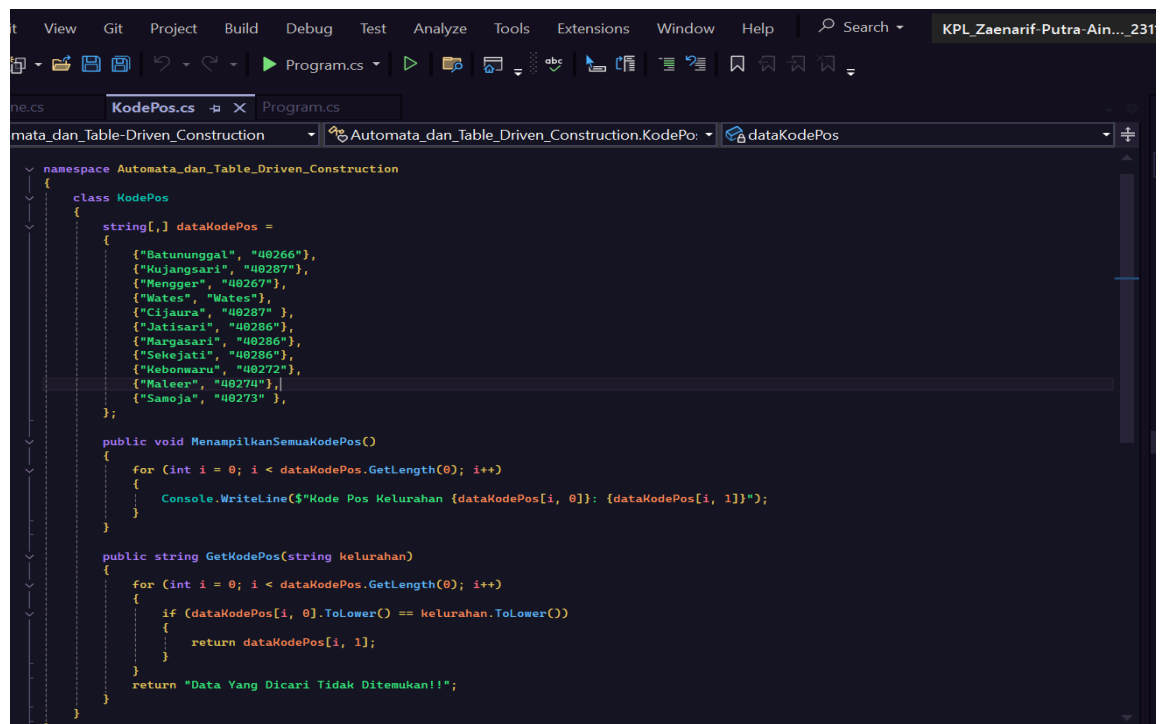
Pintu tidak terkunci
Mengecek Pintu: Pintu tidak terkunci

Pintu terkunci
Mengecek Pintu: Pintu terkunci

Pintu sudah terkunci
Pintu tidak terkunci
```

III. Penjelasan Syntax Secara Singkat

1. Implementasi Class KodePos



```
namespace Automata_dan_Table_Driven_Construction
{
    class KodePos
    {
        string[,] dataKodePos =
        {
            {"Batununggal", "40266"},
            {"Kujangsari", "40287"},
            {"Mengger", "40267"},
            {"Wates", "Wates"},
            {"Cijaura", "40287"},
            {"Jatisari", "40286"},
            {"Margasari", "40286"},
            {"Sekejati", "40286"},
            {"Kebonwaru", "40272"},
            {"Maleer", "40274"},
            {"Samoja", "40273"},
        };

        public void MenampilkanSemuaKodePos()
        {
            for (int i = 0; i < dataKodePos.GetLength(0); i++)
            {
                Console.WriteLine($"Kode Pos Kelurahan {dataKodePos[i, 0]}: {dataKodePos[i, 1]}");
            }
        }

        public string GetKodePos(string kelurahan)
        {
            for (int i = 0; i < dataKodePos.GetLength(0); i++)
            {
                if (dataKodePos[i, 0].ToLower() == kelurahan.ToLower())
                {
                    return dataKodePos[i, 1];
                }
            }
            return "Data Yang Dicari Tidak Ditemukan!!";
        }
    }
}
```

Kode C# ini dirancang untuk menyimpan dan menampilkan kode pos dari berbagai kelurahan dalam format array dua dimensi. Kelas KodePos memiliki metode MenampilkanSemuaKodePos(), yang bertugas mencetak seluruh kelurahan beserta kode pos yang terkait melalui penggunaan perulangan. Selain itu, terdapat metode GetKodePos(string kelurahan), yang berfungsi untuk mencari kode pos berdasarkan nama kelurahan yang diberikan. Apabila kelurahan tersebut ditemukan, kode pos akan dikembalikan; sebaliknya, jika tidak ditemukan, akan ditampilkan pesan yang menyatakan bahwa data tidak tersedia. Program ini menawarkan cara pencarian kode pos yang efisien dan fleksibel dengan perbandingan huruf yang tidak peka terhadap kapital.

2. Implementasi Class DoorMachine

```
using System;

namespace Automata_dan_Table_Driven_Construction
{
    public class DoorMachine
    {
        public enum State { Terkunci, Terbuka }

        private State currentState;

        public DoorMachine()
        {
            currentState = State.Terkunci;
            Console.WriteLine("Pintu terkunci");
        }

        public void BukaPintu()
        {
            if (currentState == State.Terkunci)
            {
                currentState = State.Terbuka;
                Console.WriteLine("Pintu tidak terkunci");
            }
            else
            {
                Console.WriteLine("Pintu sudah terbuka");
            }
        }

        public void KunciPintu()
        {
            if (currentState == State.Terbuka)
            {
                currentState = State.Terkunci;
                Console.WriteLine("Pintu terkunci");
            }
            else
            {
                Console.WriteLine("Pintu sudah terkunci");
            }
        }

        // ☒ Mengubah StatusPintu agar mengembalikan string
        public string StatusPintu()
        {
            return currentState == State.Terkunci ? "Pintu terkunci" : "Pintu tidak terkunci";
        }
    }
}
```

Kode C# ini adalah sebuah implementasi dari finite state machine (FSM) yang digunakan untuk mengatur status pintu melalui kelas DoorMachine. Pintu memiliki dua status utama, yaitu Terkunci dan Terbuka, yang dikelola

menggunakan enumerasi State. Ketika objek DoorMachine diinisialisasi, pintu secara otomatis berada dalam keadaan terkunci. Metode BukaPintu() berfungsi untuk mengubah status pintu menjadi terbuka jika sebelumnya dalam keadaan terkunci, sedangkan metode KunciPintu() akan mengembalikan status pintu menjadi terkunci jika sebelumnya dalam keadaan terbuka. Selain itu, metode StatusPintu() memberikan informasi mengenai apakah pintu dalam keadaan terkunci atau tidak, sehingga sistem dapat mengontrol akses dengan logika yang sederhana dan efisien.

3. Implementasi Class Program

```
namespace Automata_dan_Table_Driven_Construction
{
    class Program
    {
        static void Main(string[] args)
        {
            KodePos kodePos = new KodePos(); // Membuat objek kode pos
            Console.WriteLine("Menampilkan Data Kode Pos : "); // Menampilkan data kode pos
            kodePos.MenampilkanSemuaKodePos(); // Menampilkan semua kode pos
            Console.WriteLine("");

            Console.WriteLine("Menampilkan System Pintu :");
            DoorMachine pintu = new DoorMachine();
            Console.WriteLine();

            pintu.BukaPintu();
            Console.WriteLine($"Mengecek Pintu: {pintu.StatusPintu()}");
            Console.WriteLine();

            pintu.KunciPintu();
            Console.WriteLine($"Mengecek Pintu: {pintu.StatusPintu()}");
            Console.WriteLine();

            pintu.KunciPintu(); // Coba kunci lagi
            pintu.BukaPintu(); // Coba buka lagi

            Console.ReadLine();
        }
    }
}
```

Kode C# ini mengimplementasikan program utama yang bertugas mengelola informasi kode pos serta sistem kontrol pintu. Sebuah objek KodePos diciptakan untuk menampilkan daftar kode pos dari berbagai kelurahan. Di sisi lain, objek DoorMachine digunakan untuk mensimulasikan status pintu, yang diawali dengan kondisi terkunci. Program ini kemudian membuka pintu, memeriksa statusnya, menguncinya kembali, dan mencetak status pintu setelah setiap perubahan yang terjadi. Selain itu, terdapat percobaan untuk mengunci dan membuka pintu beberapa kali guna menguji logika transisi status. Program ini menunjukkan penerapan automata sederhana serta interaksi dengan data yang berbasis tabel.