

**TUGAS PENDAHULUAN MODUL 05
KONSTRUKSI PERANGKAT LUNAK**

GENERIC



**DISUSUN OLEH:
KHOLIFAH DINA
2211104004**

SE 06 01

**S1 REKAYASA PERANGKAT LUNAK
FAKULTAS INFORMATIKA
TELKOM UNIVERSITY
2025**

1. MEMBUAT BRANCH BARU PADA GIT PROJECT

Pada folder project yang sedang aktif, langkah-langkah yang perlu dikerjakan:

- A. Membuat git branch baru dengan nama “generic-method”.
- B. Berpindah ke branch yang sudah dibuat sebelumnya yaitu “generic-method”.

```
PS D:\Kuliah\SEMESTER 6\KPL_KHOLIFAH DINA_2211104004_SE0601\05_Generics> git add .
PS D:\Kuliah\SEMESTER 6\KPL_KHOLIFAH DINA_2211104004_SE0601\05_Generics> git checkout -b generic-method
Switched to a new branch 'generic-method'
```

2. MENAMBAHKAN METHOD DENGAN GENERIC

Tanpa membuat file baru (gunakan file yang dibuat saat membuat project):

- A. Buatlah sebuah class bernama “HaloGeneric”.
- B. Pada class tersebut, tambahkan sebuah method dengan nama “SapaUser” yang memiliki generic parameter yang akan melakukan print “Halo user X” dimana X adalah input/nilai argument yang diberikan pada method tersebut.
- C. Panggil method tersebut pada fungsi/method utama dengan input String dengan isi nilai nama panggilan praktikan.

Source Kode:

```
using System;

class HaloGeneric
{
    public void SapaUser<T>(T user)
    {
        Console.WriteLine($"Halo user {user}");
        Console.ReadLine();
    }
}

class Program
{
    static void Main()
    {
        HaloGeneric halo = new HaloGeneric();
        halo.SapaUser("Dina"); // Ganti dengan nama panggilan praktikan
    }
}
```

Output:

```
D:\Kuliah\SEMESTER 6\KPL_KHOLIFAH DINA_2211104004_SE0601\05_Generics\tpmodul5_2211104004\bin\Debug\net8.0\TP5.exe
Halo user Dina
```

Penjelasan:

Kelas HaloGeneric memiliki metode SapaUser<T>(T user), di mana <T> adalah **generic type parameter** yang memungkinkan metode ini menerima berbagai jenis data, seperti string, integer, atau tipe lainnya. Di dalam metode tersebut, program mencetak pesan sapaan menggunakan Console.WriteLine(\$"Halo user {user}"), yang menampilkan teks dengan nama pengguna yang diberikan sebagai argumen. Metode ini juga menggunakan

Console.ReadLine() agar program tidak langsung tertutup setelah menampilkan pesan. Kemudian, di dalam Main(), objek HaloGeneric dibuat, dan metode SapaUser dipanggil dengan nilai string "Dina" sebagai parameter, sehingga program akan menampilkan output **"Halo user Dina"**. Program ini menunjukkan bagaimana konsep **generics** digunakan untuk membuat metode yang fleksibel dan dapat menangani berbagai tipe data tanpa kehilangan tipe aslinya.

3. MENAMBAHKAN METHOD DENGAN GENERIC

Tanpa membuat file baru (gunakan file yang dibuat saat membuat project dan pastikan branch aktif adalah pada branch generic-class):

- A. Buatlah sebuah class bernama "DataGeneric" dengan mengikuti class model yang ditunjukkan pada gambar/tabel di bawah ini. Class tersebut memiliki property "Data" yang bertipe generic "T" dan memiliki konstruktor dengan parameter data.
DataGeneric
- data: T
+ DataGeneric(T)
+ PrintData(): void
- B. Class tersebut juga memiliki method bernama PrintData yang melakukan print di console dengan output "Data yang tersimpan adalah: Y", dengan "Y" adalah nilai dari property "data" dari kelas tersebut.
- C. Panggil method PrintData() setelah mengisi "data" dengan NIM pada fungsi/method utama.

Source Code:

```
using System;

namespace GenericExample
{
    // Kelas generic untuk menyimpan data
    class DataGeneric<T>
    {
        private T data;

        // Konstruktor dengan parameter data
        public DataGeneric(T data)
        {
            this.data = data;
        }

        // Method untuk mencetak data
        public void PrintData()
        {
            Console.WriteLine($"Data yang tersimpan adalah: {data}");
        }
    }
}
```

```

    }

    class Program
    {
        static void Main()
        {
            // Memasukkan NIM sebagai data generik
            DataGeneric<string> dataNIM = new DataGeneric<string>("2211104004"); //
            Ganti dengan NIM kamu

            // Memanggil method PrintData
            dataNIM.PrintData();

            // Mencegah console langsung tertutup
            Console.WriteLine("\nTekan ENTER untuk keluar...");
            Console.ReadLine();
        }
    }
}

```

Output:



Penjelasan Program:

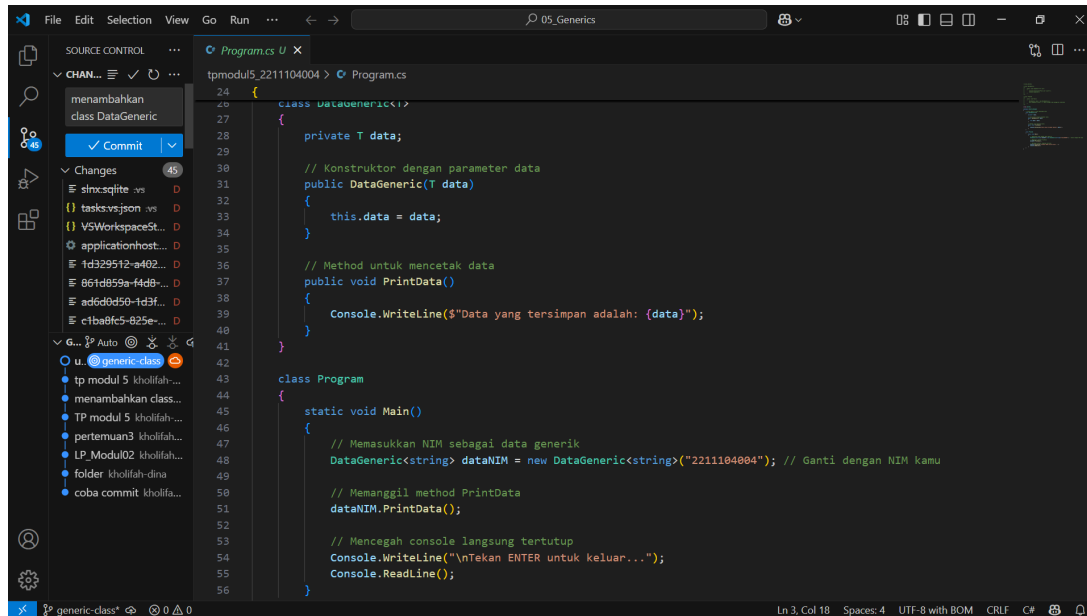
Program di atas merupakan implementasi **generic class** di C# yang dapat dijalankan di **Visual Studio**. Program ini berada dalam namespace **GenericExample**, yang membantu dalam mengorganisir kode agar tidak terjadi konflik dengan kelas lain dalam proyek. Kelas **DataGeneric<T>** menggunakan parameter **generic <T>**, sehingga dapat menyimpan berbagai tipe data. Kelas ini memiliki atribut **data** yang bertipe **T**, serta sebuah konstruktor yang menerima nilai awal untuk atribut tersebut. Selain itu, terdapat method **PrintData()** yang mencetak nilai yang tersimpan dengan format "**Data yang tersimpan adalah: Y**", di mana **Y** adalah nilai dari atribut **data**.

Di dalam **Main()**, objek **dataNIM** dibuat dari kelas **DataGeneric<string>** dengan nilai "**2211104004**" sebagai contoh NIM. Kemudian, method **PrintData()** dipanggil untuk menampilkan data yang tersimpan. Untuk mencegah **console langsung tertutup** setelah program dieksekusi, ditambahkan **Console.WriteLine("\nTekan ENTER untuk keluar...")** diikuti oleh **Console.ReadLine();**. Program ini bisa dijalankan di **Visual Studio** dengan menekan **Ctrl + F5**, sehingga pengguna dapat melihat output tanpa console langsung tertutup. Dengan menggunakan **generic class**, program ini menjadi fleksibel dan dapat digunakan untuk berbagai tipe data selain string.

4. MELAKUKAN COMMIT, PUSH, DAN PINDAH BRANCH BAGIAN 2

Pada branch yang sedang aktif saat ini (branch “generic-class”):

- Lakukan commit dengan pesan “menambahkan class DataGeneric”.
- Lakukan push ke github pada branch “generic-class” di remote/github repo.
- Setelah proses push berhasil, ganti branch yang aktif ke master/main branch.



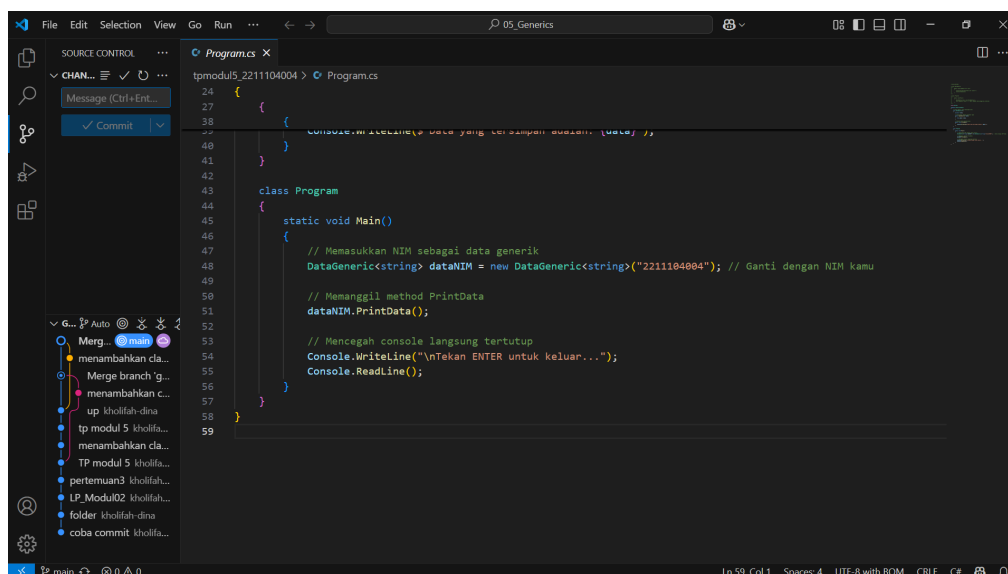
The screenshot shows the Visual Studio Code interface with the 'generic-class' branch selected in the Source Control panel. The 'Commit' button is highlighted, and the commit message 'menambahkan class DataGeneric' is entered. The main editor displays the code for 'Program.cs', which includes a 'DataGeneric' class and a 'Program' class with a 'Main' method.

```
24 {
25     class DataGeneric<T>
26     {
27         private T data;
28
29         // Konstruktor dengan parameter data
30         public DataGeneric(T data)
31         {
32             this.data = data;
33         }
34
35         // Method untuk mencetak data
36         public void PrintData()
37         {
38             Console.WriteLine($"Data yang tersimpan adalah: {data}");
39         }
40     }
41
42     class Program
43     {
44         static void Main()
45         {
46             // Memasukkan NIM sebagai data generik
47             DataGeneric<string> dataNIM = new DataGeneric<string>("2211104004"); // Ganti dengan NIM kamu
48
49             // Memanggil method PrintData
50             dataNIM.PrintData();
51
52             // Mencegah console langsung tertutup
53             Console.WriteLine("\nTekan ENTER untuk keluar...");
54             Console.ReadLine();
55         }
56     }
57 }
```

5. MELAKUKAN GIT MERGE DARI KEDUA BRANCH BARU

Pastikan branch aktif adalah branch master/main :

- Lakukan git merge branch “generic-method” ke branch master/main.
- Lakukan git merge branch “generic-class” ke branch master/main, dan jika terjadi merge conflict, pastikan semua baris yang conflict sudah diperbaiki.
- Lakukan git push untuk branch master/main ke github repository.



The screenshot shows the Visual Studio Code interface with the 'main' branch selected in the Source Control panel. The 'Merge' button is highlighted, and the merge message 'Merge branch "generic-class"' is entered. The main editor displays the code for 'Program.cs', which includes a 'DataGeneric' class and a 'Program' class with a 'Main' method.

```
24 {
25     class DataGeneric<T>
26     {
27         private T data;
28
29         // Konstruktor dengan parameter data
30         public DataGeneric(T data)
31         {
32             this.data = data;
33         }
34
35         // Method untuk mencetak data
36         public void PrintData()
37         {
38             Console.WriteLine($"Data yang tersimpan adalah: {data}");
39         }
40     }
41
42     class Program
43     {
44         static void Main()
45         {
46             // Memasukkan NIM sebagai data generik
47             DataGeneric<string> dataNIM = new DataGeneric<string>("2211104004"); // Ganti dengan NIM kamu
48
49             // Memanggil method PrintData
50             dataNIM.PrintData();
51
52             // Mencegah console langsung tertutup
53             Console.WriteLine("\nTekan ENTER untuk keluar...");
54             Console.ReadLine();
55         }
56     }
57 }
```