LAPORAN PRAKTIKUM
KONSTRUKSI PERANGKAT BERGERAK

MODUL V

GENERICS

Disusun Oleh :

Ganesha Rahman Gibran

S1SE-06-02

Asisten Praktikum :

Muhamad Taufiq Hidayat

Dosen Pengampu :

Riyan Dwi Yulian Prakoso, S.Kom., M.Kom.

PROGRAM STUDI S1 REKAYASA PERANGKAT LUNAK

DIREKTORAT TELKOM KAMPUS PURWOKERTO

2025

BAB I
PENDAHULUAN

A. DASAR TEORI

Pengantar Generics

Generics merupakan pendekatan pemrograman yang memungkinkan pengembangan kode yang lebih fleksibel, efisien, dan dapat digunakan kembali dengan mudah untuk berbagai jenis data. Di sejumlah bahasa pemrograman seperti TypeScript dan C#, dukungan terhadap generics bersifat eksplisit. Sementara itu, dalam JavaScript, konsep serupa dapat diterapkan melalui pola penggunaan class atau function. Penerapan generics sangat bermanfaat dalam dunia rekayasa perangkat lunak karena mendorong pembuatan fungsi atau class yang bersifat umum, sehingga tidak perlu dibuat ulang untuk setiap tipe data yang berbeda. Hasilnya, kode menjadi lebih terstruktur, mudah dikembangkan, serta mengurangi risiko duplikasi.

Generic Class

Generic Class adalah konsep dalam pemrograman yang memungkinkan kita membangun struktur data yang mampu menangani berbagai jenis elemen tanpa perlu membuat class terpisah untuk tiap tipe data. Di JavaScript, prinsip ini bisa diwujudkan melalui penggunaan array atau class yang dirancang agar cukup dinamis dalam menerima berbagai tipe nilai.

Generic Function

Generic Function adalah pendekatan pemrograman yang memungkinkan penulisan fungsi yang bersifat universal, sehingga dapat digunakan untuk berbagai tipe data tanpa perlu menyalin atau menulis ulang kode. Teknik ini sangat berguna, terutama saat menangani proses manipulasi data dengan karakteristik atau jenis yang beragam.

Generic Delegate

Generic Delegate merupakan bagian dari konsep generics yang memungkinkan suatu fungsi diteruskan sebagai argumen, sehingga dapat meningkatkan fleksibilitas dan modularitas dalam penulisan kode. Pendekatan ini kerap dimanfaatkan dalam JavaScript, khususnya dalam penerapan callback function.

B. MAKSUD DAN TUJUAN

1. Memahami konsep generics untuk meningkatkan fleksibilitas dan efisiensi dalam pemrograman.
2. Menerapkan generic class yang dapat menangani berbagai tipe data dalam satu struktur.
3. Membuat generic function agar fungsi dapat digunakan untuk berbagai jenis data tanpa duplikasi kode.
4. Menggunakan generic delegate untuk memaksimalkan modularitas melalui fungsi yang diteruskan sebagai parameter

BAB II
IMPLEMENTASI (GUIDED)

A. Generic Class

Source code

```
05_Generics > JS Class.js > ...
1   class GenericList {
2       constructor() {
3         this.items = [];
4       }
5
6       add(item) {
7         this.items.push(item);
8       }
9
10      getAll() {
11        return this.items;
12      }
13    }
14
15    const list = new GenericList();
16    list.add(1);
17    list.add("Hello");
18    console.log(list.getAll());
```

Output

```
[Running] node "e:\Konstruksi Perangkat Lunak\KPL_Ganesha_Rahman_Gibran
[ 1, 'Hello' ]

[Done] exited with code=0 in 0.618 seconds
```

Deskripsi Code

Generic Class memungkinkan pengelolaan berbagai tipe data dalam satu struktur yang sama. Contohnya, class GenericList menggunakan array items untuk menyimpan elemen yang ditambahkan. Fungsi add(item) berfungsi untuk menambahkan data ke array, sedangkan getAll() mengambil seluruh isi array tersebut. Setelah membuat instance dari class ini, kita bisa menambahkan nilai seperti angka 1 dan string "Hello", lalu menampilkannya menggunakan console.log(list.getAll()), yang akan menghasilkan output [1, "Hello"]. Pendekatan ini memudahkan penyimpanan beragam data tanpa perlu membuat class khusus untuk setiap tipe.

B. Generic Function

Code

```
05_Generics > JS Function.js > ...
1   function swap(a, b) {
2       return [b, a];
3   }
4
5   let x = 3, y = 7;
6   [x, y] = swap(x, y);
7   console.log(x, y);
```

Output

```
[Running] node "e:\Konstruksi Perangkat Lunak\KPL_Ganesha_Rahman
7 3
```

Deskripsi Code

Generic Function di JavaScript dapat digunakan untuk menukar nilai dua variabel dengan lebih sederhana. Fungsi swap(a, b) menerima dua argumen dan mengembalikannya dalam urutan terbalik sebagai array [b, a]. Misalnya, jika variabel x dan y bernilai 5 dan 10, maka setelah pemanggilan swap(x, y) dan penerapan destructuring [x, y] = swap(x, y);, nilai x dan y akan bertukar. Jika fungsi ini dipanggil dengan nilai 3 dan 7, maka hasil akhirnya adalah x = 7 dan y = 3, yang akan ditampilkan dengan console.log(x, y);. Teknik ini memungkinkan pertukaran nilai tanpa variabel tambahan.

C. Generic Delegate

Code

```
05_Generics > JS Delegate.js > genericDelegate
1   function genericDelegate(callback, value) {
2       callback(value);
3   }
4
5   genericDelegate(console.log, "Event Triggered");
```

Output

```
[Running] node "e:\Konstruksi Perangkat Lunak\KPL_Ganesha_Rah
Event Triggered
```

Deskripsi Code

Generic Delegate di JavaScript memungkinkan kita meneruskan fungsi sebagai argumen ke fungsi lainnya. Misalnya, genericDelegate(callback, value) menerima sebuah fungsi dan nilai, lalu mengeksekusi fungsi tersebut dengan nilai yang diberikan melalui callback(value). Ketika dipanggil seperti genericDelegate(console.log, "Event Triggered"), maka console.log("Event Triggered") akan dijalankan dan mencetak pesan tersebut ke konsol. Pendekatan ini umum digunakan dalam callback function untuk membuat kode lebih modular dan fleksibel.

BAB III
PENUGASAN (UNGUIDED)

Soal 1 – Halo Generic

Code

```
05_Generics > JS HaloGeneric.js > ...
1   class HaloGeneric {
2       SapaUser(x) {
3         console.log(`Halo user ${x}`);
4       }
5     }
6
7     const halo = new HaloGeneric();
8
9     halo.SapaUser("Ganesha Gibran");
10
11    console.log("=== Code Execution Successful ===");
```

Output

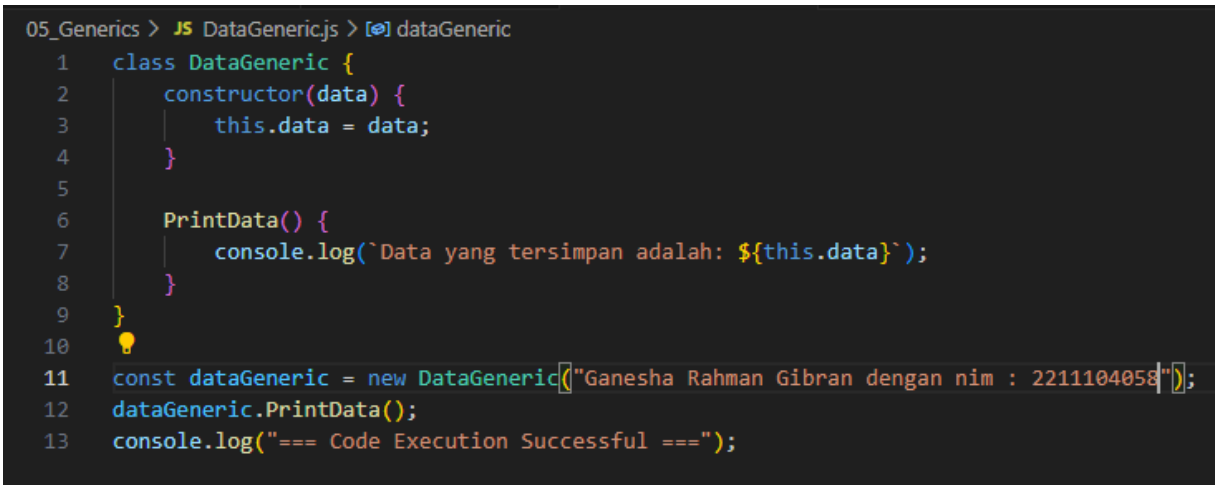
```
[Running] node "e:\Konstruksi Perangkat Lunak\KPL_Ga
Halo user Ganesha Gibran
=== Code Execution Successful ===
```

Deskripsi Code

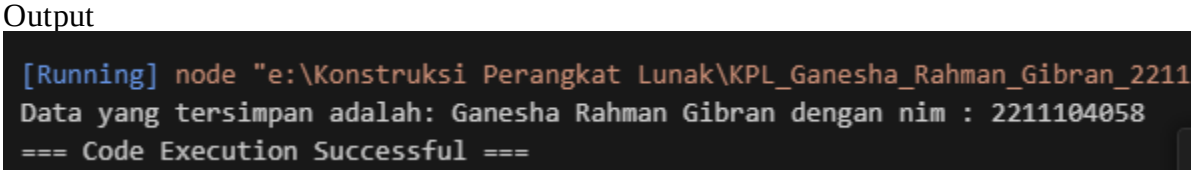
Class HaloGeneric memiliki constructor yang menerima argumen user dan menyimpannya sebagai properti. Method SapaUser() akan mencetak pesan sapaan seperti "Halo user X", dengan X merupakan nilai dari properti tersebut. Contohnya, jika objek dibuat dengan nama "Ganesha Gibran" lalu sapa.SapaUser(); dipanggil, maka output-nya adalah "Halo user Ganesha Gibran". Contoh ini memperlihatkan bagaimana sebuah class bisa digunakan secara fleksibel dengan berbagai input tanpa perlu mengubah struktur dasarnya

Soal 2 – Data Generic

Code



Output



Deskripsi Code

Class DataGeneric memiliki constructor yang menyimpan nilai data sebagai properti, dan menyediakan method PrintData() untuk menampilkan pesan "Data yang tersimpan adalah: X", di mana X adalah isi dari data tersebut. Ketika objek dibuat dengan nilai "Ganesha Rahman Gibran dengan nim : 2211104058" dan method dipanggil, maka output yang muncul di konsol adalah pesan tersebut. Contoh ini menggambarkan bagaimana class dapat mengelola berbagai tipe data secara fleksibel tanpa perlu mengubah strukturnya.