**Nama**: Raihana Aisha Az-Zahra

**NIM**: 241511056

**Prodi/Kelas**: D3/2B

# **Tugas – 2: Object & Class**

**Deskripsi Aplikasi**

Aplikasi ini adalah program sederhana berbasis Java Console untuk mengelola data Karyawan dan Departemen di sebuah perusahaan.

**Fitur Utama:**

1. **Tambah Departemen**

* User dapat memasukkan jumlah departemen yang ingin ditambahkan
* Setiap departemen memiliki ID otomatis, nama departemen, dan deskripsi
* Data departemen disimpan dalam **ArrayList departemenList**

1. **Tambah Karyawan**

* User dapat memasukkan jumlah karyawan yang ingin ditambahkan
* Setiap karyawan memiliki ID otomatis, nama karyawan, gaji, dan harus dipetakan ke salah satu departemen yang sudah ada.
* Data karyawan disimpan dalam **ArrayList karyawanList**

1. **Lihat Data**

* Menampilkan semua data Departemen dan Karyawan yang telah dimasukkan.
* Data karyawan ditampilkan lengkap dengan informasi departemen yang menaunginya.

**Konsep yang Digunakan:**

1. **Encapsulation**

Semua atribut di class Karyawan dan Departemen bersifat private, hanya bisa diakses melalui getter dan setter.

1. **Auto Increment ID**

Atribut id pada Karyawan dan Departemen dibuat otomatis bertambah dengan bantuan variabel static counter.

1. **ArrayList**

Digunakan untuk menyimpan data dinamis (jumlah karyawan/departemen tidak dibatasi).

1. **Menu Looping**

Menu utama berjalan berulang hingga user memilih keluar.

**Penjelasan Program**

|  |  |  |
| --- | --- | --- |
| **Gambar** | **Konsep/Fitur** | **Deskripsi** |
| Karyawan.java    Departemen.java | Enkapsulasi | Semua atribut (id, name, departemen, gaji, desc) dibuat private.  Akses ke atribut hanya bisa lewat getter dan setter. |
| Karyawan.java    Departemen.java | Constructor | Constructor digunakan untuk menginisialisasi id karyawan dan id departemensecara otomatis dengan autoincrement. |
| Karyawan.java    Departemen.java | Method/Behavior | Method display() dipakai untuk menampilkan data karyawan dan departemen dengan format rapi.  Hal Ini merupakan contoh **encapsulation + behavior**: objek menyimpan datanya sendiri, lalu bisa menampilkan datanya sendiri. |
| Main.java | Inisialisasi object & variable | scanner => untuk input dari user  karyawanList => menyimpan semua data karyawan  departemenList => menyimpan semua data departemen |
| Main.java | Looping menu utama | Program berjalan terus-menerus sampai user memilih Keluar (4) |
| Main.java | Case 1: Tambah Departemen | User bisa menambahkan beberapa departemen sekaligus  Data yang diminta untuk input adalah nama departemen dan deskripsinya  Objek departemen baru dibuat lalu ditambahkan ke departemenList |
| Main.java | Case 2: Tambah Karyawan | User bisa menambahkan beberapa karyawan  Data yang diminta adalah nama dan gaji  Lalu user diminta memilih departemen dari daftar departemenList  Setelah itu, karyawan ditambahkan ke karyawanList |
| Main.java | Case 3: Melihat data karyawan dan data departemen | Pemanggilan display() menampilkan detail objek masing-masing. |

**Hasil**

|  |  |
| --- | --- |
| **Gambar** | **Fitur** |
|  | Menu utama |
|  | Tambah Departemen |
|  | Tambah Karyawan |
|  | Menampilkan data departemen dan data karyawan |

**Lesson Learned**

1. **Praktik Langsung Lebih Efektif**

Dengan mencoba langsung membuat class dan object, pemahaman jadi lebih mendalam dibanding hanya membaca teori. Praktik nyata membantu mengerti alur bagaimana object dibuat, bagaimana atribut dan method saling berhubungan, serta bagaimana konsep OOP diterapkan dalam kode.

1. **Pemahaman Getter dan Setter**

Saya jadi paham bahwa getter digunakan untuk mengambil nilai atribut secara aman, sementara setter dipakai untuk mengubah nilai atribut dengan kendali tertentu. Konsep ini membantu menjaga prinsip enkapsulasi, supaya atribut tidak bisa diakses/diubah sembarangan dari luar class.

1. **Tidak Semua Atribut Perlu Setter**

Dari implementasi class Karyawan, saya belajar bahwa tidak semua atribut harus bisa diubah. Contohnya id, saya membuatnya auto increment dengan counter static. Artinya setiap object Karyawan otomatis punya id unik, dan nilainya tidak boleh diubah manual. Hal ini menegaskan pentingnya access modifier dan perancangan atribut sesuai kebutuhan.

1. **Eksplorasi Object dalam Array dan ArrayList**

Saya menemukan bahwa object tidak hanya dibuat satu per satu, tetapi juga dapat disimpan dalam bentuk Array maupun ArrayList. Array cocok untuk jumlah data yang tetap, sedangkan ArrayList lebih fleksibel karena ukurannya bisa berubah. Dari sini saya belajar bagaimana mengelola Kumpulan object dengan lebih dinamis

**Referensi**

[**https://www.geeksforgeeks.org/java/how-to-create-array-of-objects-in-java/**](https://www.geeksforgeeks.org/java/how-to-create-array-of-objects-in-java/)

[**https://www.w3schools.com/java/java\_arraylist.asp**](https://www.w3schools.com/java/java_arraylist.asp)

[**https://stackoverflow.com/questions/3982550/creating-an-arraylist-of-objects**](https://stackoverflow.com/questions/3982550/creating-an-arraylist-of-objects)