LAPORAN PEMROGRAMAN
APLIKASI PEMESANAN MAKANAN

DISUSUN OLEH:

KELOMPOK 17

Nama Anggota :

1. Andrian Revaldo (G1A025029)
2. Aldi Cahya Dinitra (G1A025085)
3. Jovancha Adi Dharma (G1A025113)

Dosen Pengampu :

1. Funny Farady Coastera, S Kom, M.T
2. Arie Vatresia, S.T., M.TI., P.h.D

PROGRAM STUDI INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS BENGKULU

2025

DAFTAR ISI

LANDASAN TEORI

Java merupakan 1ystem pemrograman yang berorientasi objek secara murni (pure object-oriented programming), artinya seluruh elemen di dalamnya didasarkan pada konsep objek dan kelas. Prinsip utama dari pemrograman berorientasi objek seperti enkapsulasi, pewarisan, dan polimorfisme diterapkan secara konsisten dalam Java. Hal ini menjadikan Java sebagai 1ystem yang terstruktur, modular, serta mudah dikembangkan dan dipelihara.

Dalam pengembangannya, Java menyediakan berbagai versi dan alat bantu, salah satunya 1ystem JDK (Java Development Kit) yang berfungsi sebagai paket utama bagi para pengembang untuk membuat, mengompilasi, dan menjalankan program Java. Di dalam JDK telah tersedia berbagai pustaka (library) dan kelas bawaan yang memudahkan pemrogram dalam mengelola struktur data, seperti List, Queue, dan Stack. Struktur data tersebut berperan penting dalam pengelolaan dan penyimpanan data secara dinamis, sehingga proses manipulasi data dapat dilakukan dengan lebih efisien dan fleksibel.

Kelas List digunakan untuk menyimpan sekumpulan elemen yang terurut dan dapat diakses berdasarkan indeks. Sementara itu, Queue berfungsi sebagai antrian dengan prinsip *First In First* Out (FIFO), di mana elemen yang pertama dimasukkan akan menjadi yang pertama keluar. Adapun Stack menggunakan prinsip Last In First Out (LIFO), yaitu elemen terakhir yang dimasukkan akan menjadi yang pertama diambil. Java menyediakan implementasi dari struktur-struktur ini dalam paket java.util, seperti ArrayList, LinkedList, PriorityQueue, dan Stack, yang semuanya siap digunakan tanpa perlu membuat struktur data dari awal.

Selain itu, Java juga menyediakan mekanisme untuk mengakses data dari struktur tersebut melalui array dan iterator. Data yang tersimpan di dalam list, queue, atau stack dapat dengan mudah dikonversi menjadi array untuk memudahkan pengolahan data secara indeks, atau diakses menggunakan iterator yang memungkinkan penelusuran data secara sekuensial (satu arah). Dengan fitur-fitur ini, Java memberikan fleksibilitas tinggi dalam pengolahan data, sekaligus mempertahankan prinsip berorientasi objek yang menjadi fondasi utama 1ystem tersebut.

Secara keseluruhan, keberadaan JDK dan berbagai kelas bawaan Java menunjukkan bahwa 1ystem ini tidak hanya kuat dari sisi konsep, tetapi juga praktis dalam penerapannya, terutama dalam pemrograman yang memerlukan pengelolaan struktur data yang kompleks dan efisien.

Array merupakan salah satu struktur data dasar dalam pemrograman yang berfungsi sebagai wadah untuk menyimpan sekumpulan data dengan tipe yang sama. Dalam bahasa pemrograman Java, array digunakan untuk mengelola sejumlah elemen yang memiliki tipe data seragam, seperti integer, string, atau tipe data lainnya. Array memiliki peranan penting karena memudahkan programmer dalam mengorganisasi, mengakses, dan memanipulasi data secara efisien. Setiap elemen dalam array disimpan pada lokasi memori yang berurutan dan dapat diakses melalui indeks, di mana indeks pertama dimulai dari angka nol. Dengan demikian, proses pengambilan atau pengubahan data menjadi lebih cepat karena posisi setiap elemen sudah diketahui secara pasti.

Proses pembuatan array di Java dimulai dengan mendeklarasikan tipe data dan jumlah elemen yang akan digunakan. Misalnya, ketika seorang programmer ingin membuat array bertipe integer dengan lima elemen, maka deklarasinya akan menentukan kapasitas array sekaligus tipe data yang dapat disimpan di dalamnya. Setelah array dibuat, setiap elemen dapat diisi dengan nilai tertentu dan dapat diakses menggunakan indeksnya. Java juga menyediakan berbagai metode atau properti bawaan yang memudahkan manipulasi array, seperti menentukan panjang array dengan atribut .length, mengubah nilai elemen berdasarkan indeks tertentu, maupun melakukan iterasi terhadap seluruh elemen menggunakan perulangan.

Selain digunakan untuk menyimpan data sederhana, array juga memiliki peran penting dalam pembentukan struktur data yang lebih kompleks. Misalnya, array dua dimensi dapat digunakan untuk merepresentasikan matriks yang terdiri dari baris dan kolom. Dalam konteks ini, array menjadi dasar bagi berbagai algoritma yang digunakan dalam komputasi numerik, grafik komputer, hingga pengolahan data ilmiah. Tidak hanya itu, array juga dapat dikembangkan menjadi bentuk struktur data yang lebih dinamis seperti stack, queue (antrian), dan linked list, yang masing-masing memiliki fungsi spesifik dalam pemrograman. Stack misalnya digunakan untuk sistem penyimpanan data dengan prinsip LIFO (Last In, First Out), sedangkan queue bekerja dengan prinsip FIFO (First In, First Out).

Secara keseluruhan, array merupakan fondasi penting dalam pemrograman karena menyediakan cara yang sistematis dan efisien untuk menyimpan serta mengelola data. Pemahaman terhadap konsep dasar array akan membantu programmer dalam membangun algoritma yang lebih kompleks dan mengoptimalkan kinerja program. Dalam konteks praktikum, penggunaan array memungkinkan mahasiswa untuk memahami konsep penyimpanan data berurutan, cara mengakses

dan memodifikasinya, serta bagaimana array dapat diterapkan dalam berbagai kasus nyata seperti pengolahan data, simulasi, maupun pengembangan aplikasi berbasis logika yang membutuhkan pengelolaan data secara terstruktur.

Enkapsulasi merupakan salah satu prinsip dasar dalam paradigma pemrograman berorientasi objek (Object Oriented Programming/OOP) yang memiliki peranan penting dalam menjaga keamanan serta keteraturan struktur program. Konsep ini bertujuan untuk menyembunyikan detail implementasi suatu objek dari dunia luar dan hanya memperlihatkan hal-hal yang benar-benar diperlukan melalui antarmuka (interface) 3ystem. Dengan 3ystem33, enkapsulasi membantu melindungi data agar tidak dapat diakses atau diubah secara langsung oleh bagian lain dari program yang tidak berhak, melainkan melalui metode yang telah disediakan oleh kelas tersebut.

Dalam praktiknya, enkapsulasi sering diterapkan pada berbagai 3ystem pemrograman modern seperti Java, Python, C++, maupun C#. Dalam 3ystem pemrograman Java sendiri, prinsip ini diwujudkan melalui penggunaan modifier akses seperti private, protected, dan public yang mengatur 3ystem3 visibilitas atribut dan metode di dalam kelas. Misalnya, atribut yang bersifat private hanya dapat diakses melalui metode getter dan setter, sehingga programmer memiliki kendali penuh terhadap bagaimana data tersebut dimanipulasi. Hal ini tidak hanya meningkatkan keamanan data, tetapi juga menjaga agar perubahan internal pada suatu kelas tidak memengaruhi bagian lain dari 3ystem.

Penerapan enkapsulasi juga banyak ditemukan dalam pengembangan 3ystem yang lebih luas, seperti dalam pembuatan REST API di mana data dari server tidak diberikan secara langsung kepada pengguna, melainkan melalui endpoint tertentu yang telah dikendalikan. Demikian pula, dalam proses hosting website, konsep enkapsulasi diterapkan untuk membatasi akses terhadap sumber daya server agar hanya bagian tertentu yang dapat diakses 3ystem. Dengan demikian, enkapsulasi tidak hanya berfungsi dalam skala kecil di level kode program, tetapi juga memiliki dampak signifikan dalam menjaga keamanan, efisiensi, dan modularitas 3ystem perangkat lunak secara keseluruhan.

Secara keseluruhan, enkapsulasi menjadikan program lebih mudah untuk dikembangkan, dikelola, dan diperluas. Dengan memisahkan antara bagian internal dan eksternal dari sebuah objek, prinsip ini membantu menciptakan struktur kode yang lebih rapi dan fleksibel.

SOAL DAN PEMBAHASAN

STUDI KASUS 7 : APLIKASI PEMESANAN MAKANAN

Deskripsi:

Seorang pengusaha ingin membuat aplikasi sederhana untuk menerima pesanan makanan dari pelanggan. Program akan meminta input nama pelanggan, nomor telepon, jenis makanan, jumlah pesanan, dan harga per porsi. Sistem akan menghitung total harga dan menampilkan data pesanan ke layar menggunakan konsep encapsulation.

Struktur Program:

1. Class Pemesanan

a. Atribut: pelanggan (String[][]), pesanan (Object[][]), totalHarga (int)

b. Method:

a. setPelanggan(String nama, String telepon)

Menyimpan data pelanggan.

b. setPesanan(String jenisMakanan, int jumlah)

Menyimpan data pesanan.

c. hitungTotalHarga(int harga)

Mengalikan harga * jumlah.

d. getPelanggan(), getPesanan(), getTotalHarga()

Mengambil data hasil input.

2. Class Main

a. Input data pelanggan dan pesanan.

b. Menampilkan data pemesanan dan total harga pesanan.

***Printscreen Source Code* :**

```
public class Pemesanan {
    private String[][] pelanggan = new String[1][2];
    private Object[][] pesanan = new Object[1][2];
    private int totalHarga;

    public void setPelanggan(String nama, String telepon) {
        pelanggan[0][0] = nama;
        pelanggan[0][1] = telepon;
    }

    public void setPesanan(String jenisMakanan, int jumlah) {
        pesanan[0][0] = jenisMakanan;
        pesanan[0][1] = jumlah;
    }

    public void hitungTotalHarga(int hargaPerPorsi) {
        int jumlah = (int) pesanan[0][1];
        totalHarga = hargaPerPorsi * jumlah;
    }

    public String[][] getPelanggan() {
        return pelanggan;
    }

    public Object[][] getPesanan() {
        return pesanan;
    }

    public int getTotalHarga() {
        return totalHarga;
    }
}
```

Gambar 1.1 Printscreen output code class pemesanan.

***Source Code* :**

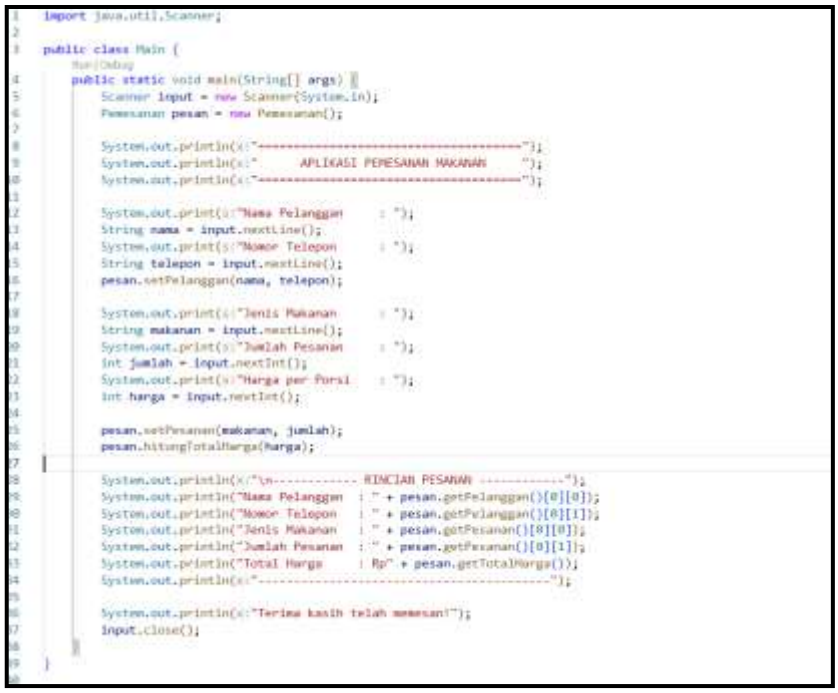
```
public class Pemesanan {
    private String[][] pelanggan = new String[1][2];
    private Object[][] pesanan = new Object[1][2];
    private int totalHarga;
    public void setPelanggan(String nama, String telepon) {
        pelanggan[0][0] = nama;
        pelanggan[0][1] = telepon;
    }
    public void setPesanan(String jenisMakanan, int jumlah) {
        pesanan[0][0] = jenisMakanan;
        pesanan[0][1] = jumlah;
    }
    public void hitungTotalHarga(int hargaPerPorsi) {
        int jumlah = (int) pesanan[0][1];
        totalHarga = hargaPerPorsi * jumlah;
    }
    public String[][] getPelanggan() {
        return pelanggan;
    }
    public Object[][] getPesanan() {
        return pesanan;
    }
    public int getTotalHarga() {
        return totalHarga;
    }
}
```

Pembahasan *Source Code* :

Class Pemesanan berfungsi sebagai tempat penyimpanan dan pengelolaan data pelanggan serta pesanan makanan dengan menerapkan konsep encapsulation agar data tidak dapat diakses secara langsung dari luar class, sehingga keamanan data tetap terjaga. Di dalamnya terdapat atribut pelanggan untuk menyimpan nama dan nomor telepon pelanggan, pesanan untuk menyimpan jenis

makanan dan jumlah pesanan, serta totalHarga untuk menyimpan hasil perhitungan harga keseluruhan. Melalui method setPelanggan dan setPesanan data pelanggan dan pesanan akan diproses dan dimasukkan oleh program ke dalam atribut masing-masing, sedangkan method hitungTotalHarga digunakan untuk menghitung total harga berdasarkan jumlah pesanan dan harga per porsi. Hasil penyimpanan dan perhitungan data tersebut dapat diakses kembali menggunakan method getPelanggan, getPesanan, dan getTotalHarga. Secara logika, class ini berperan sebagai komponen yang menangani seluruh proses pengelolaan data pemesanan makanan secara terstruktur tanpa berinteraksi langsung dengan pengguna, melainkan menyediakan data yang sudah diolah untuk ditampilkan oleh class Main nanti, sehingga program menjadi lebih terorganisir, mudah dipelihara, dan sesuai dengan prinsip pemrograman berorientasi objek.

***Printscreen Source Code* :**



Gambar 1.2 Printscreen source code class main.

***Source Code* :**

```
import java.util.Scanner;
public class Main {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);
        Pemesanan pesan = new Pemesanan();
        System.out.println("=====================================");
```

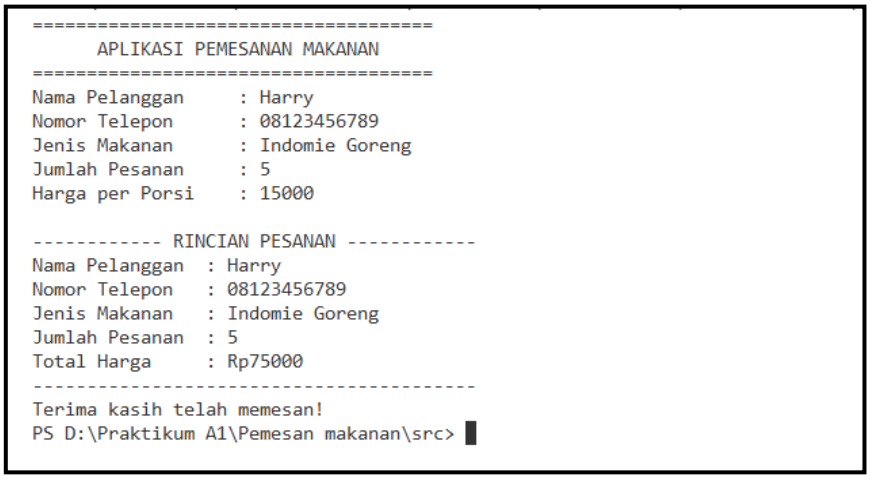
```
        System.out.println("      APLIKASI PEMESANAN MAKANAN     ");
        System.out.println("=====================================");
        System.out.print("Nama Pelanggan     : ");
        String nama = input.nextLine();
        System.out.print("Nomor Telepon      : ");
        String telepon = input.nextLine();
        pesan.setPelanggan(nama, telepon);
        System.out.print("Jenis Makanan      : ");
        String makanan = input.nextLine();
        System.out.print("Jumlah Pesanan     : ");
        int jumlah = input.nextInt();
        System.out.print("Harga per Porsi    : ");
        int harga = input.nextInt();
        pesan.setPesanan(makanan, jumlah);
        pesan.hitungTotalHarga(harga);
        System.out.println("\n------------ RINCIAN PESANAN ------------");
        System.out.println("Nama Pelanggan  : " + pesan.getPelanggan()[0][0]);
        System.out.println("Nomor Telepon   : " + pesan.getPelanggan()[0][1]);
        System.out.println("Jenis Makanan   : " + pesan.getPesanan()[0][0]);
        System.out.println("Jumlah Pesanan  : " + pesan.getPesanan()[0][1]);
        System.out.println("Total Harga     : Rp" + pesan.getTotalHarga());
        System.out.println("---------------------------------------");
        System.out.println("Terima kasih telah memesan!");
        input.close();
    }
}
```

Pembahasan *Source Code* :

Class Main berfungsi sebagai pusat eksekusi program yang mengatur jalannya proses pemesanan makanan dari awal hingga hasil akhirnya ditampilkan ke layar. Di dalam class ini terdapat method main, yaitu sebagai eksekutor ketika program dimulai. Class ini mengambil objek dari class Pemesanan untuk mengakses dan menginput data pelanggan serta pesanan makanan. Secara logika, class Main berperan sebagai penghubung antara pengguna dan sistem. Program

dimulai dengan menampilkan judul aplikasi, kemudian meminta pengguna untuk memasukkan data seperti nama pelanggan, nomor telepon, jenis makanan, jumlah pesanan, dan harga per porsi melalui input dari keyboard menggunakan Scanner. Setelah semua data diinputkan, data tersebut dikirim ke class Pemesanan melalui pemanggilan method setPelanggan dan setPesanan untuk disimpan dengan aman di dalam atribut yang bersifat privasi. Kemudian method hitungTotalHarga dijalankan untuk menghitung total biaya berdasarkan jumlah pesanan dan harga per porsi yang dimasukkan pengguna. Setelah perhitungan selesai, program mengambil kembali hasil data yang sudah diolah dengan memanggil method getPelanggan, getPesanan, dan getTotalHarga untuk kemudian ditampilkan ke layar dalam bentuk ringkasan atau detail pemesanan.

***Printscreen Output Code* :**



Gambar 1.3 Printscreen output code.

Pembahasan *Output Code* :

Output di atas menunjukkan hasil dari program Aplikasi Pemesanan Makanan yang dijalankan dengan data pelanggan bernama *Harry*. Pada bagian awal, program menampilkan judul aplikasi sebagai tampilan pembuka. Setelah pengguna mengisi data seperti nama, nomor telepon, jenis makanan, jumlah pesanan, dan harga per porsi, sistem secara otomatis menghitung total harga dengan mengalikan jumlah pesanan (5) dan harga per porsi (15.000), sehingga menghasilkan total sebesar Rp75.000. Kemudian program menampilkan kembali semua informasi tersebut dalam bentuk rincian pemesanan yang rapi dan mudah dibaca. Di akhir tampilan, terdapat garis pemisah dan ucapan terima kasih sebagai penutup, yang menandakan bahwa proses pemesanan telah selesai dengan sukses. Secara logika, output ini menunjukkan bahwa program berjalan dengan benar, memproses input dari pengguna, menghitung total biaya secara otomatis, dan menampilkan hasil akhir yang jelas.

KESIMPULAN DAN SARAN

A. Kesimpulan

Berdasarkan pembahasan di atas, dapat disimpulkan bahwa bahasa pemrograman Java merupakan sistem yang sangat kuat dan fleksibel dalam mendukung paradigma pemrograman berorientasi objek. Dengan menerapkan prinsip utama seperti enkapsulasi, pewarisan, dan polimorfisme secara konsisten, Java tidak hanya berfungsi sebagai bahasa pemrograman, tetapi juga sebagai platform yang menyediakan berbagai fasilitas untuk membangun sistem yang terstruktur, modular, dan mudah dikembangkan. Kehadiran JDK (Java Development Kit) menjadi bukti nyata bahwa Java dirancang tidak hanya untuk mempermudah penulisan kode, tetapi juga untuk memberikan lingkungan pengembangan yang lengkap dengan berbagai pustaka dan kelas bawaan yang memudahkan pengelolaan struktur data dan pengolahan informasi secara efisien.

Pemahaman terhadap konsep array dalam Java juga menjadi bagian fundamental dalam proses belajar dan praktik pemrograman. Array berperan penting sebagai wadah penyimpanan data yang bersifat homogen dan terstruktur, memungkinkan programmer untuk mengakses, memanipulasi, dan mengelola data secara cepat melalui indeks. Selain sebagai struktur data dasar, array juga menjadi fondasi bagi pembentukan struktur data yang lebih kompleks seperti stack, queue, dan linked list. Melalui array, mahasiswa dapat memahami konsep dasar penyimpanan data berurutan serta proses pengaksesan dan pengolahan elemen yang terorganisir dengan baik. Hal ini menjadi dasar penting dalam mengembangkan algoritma yang efisien dan dalam membangun sistem pemrograman yang lebih kompleks di masa depan.

B. Saran

Berdasarkan hasil praktikum dan pemahaman terhadap konsep Java, disarankan agar mahasiswa terus melatih kemampuan pemrogramannya dengan menerapkan konsep-konsep yang telah dipelajari ke dalam berbagai kasus nyata. Praktikum selanjutnya sebaiknya diarahkan untuk memperdalam penerapan struktur data dan prinsip OOP lainnya seperti pewarisan dan polimorfisme agar pemahaman menjadi lebih menyeluruh dan aplikatif.

Selain itu, mahasiswa juga disarankan untuk lebih aktif mengeksplorasi pustaka dan fitur yang tersedia dalam JDK agar mampu memanfaatkan seluruh potensi yang dimiliki oleh bahasa pemrograman Java. Pemahaman yang mendalam terhadap penggunaan kelas-kelas bawaan seperti List, Queue, Stack, dan ArrayList akan sangat membantu dalam membangun sistem yang efisien, aman, dan terstruktur dengan baik.

DAFTAR PUSTAKA

Niko, B. (2016). *Menggunakan Array di Java*. Www.Codepolitan.Com. https://www.codepolitan.com/blog/menggunakan-array-di-java/

Rosyida, M. (2024). *Enkapsulasi adalah Konsep Penting dalam Pemrograman OOP*. Www.Domainesia.Com. https://www.domainesia.com/berita/enkapsulasi-adalah/

Selamet, R. (2016). IMPLEMENTASI STRUKTUR DATA LIST, QUEUE DAN STACK DALAM JAVA. In *Media Informatika* (Vol. 15, Issue 3).

LAMPIRAN