LAPORAN AKHIR PRAKTIKUM

Mata Praktikum	: Pengolahan Citra
Kelas	: 4IA01
Praktikum ke-	: 2
Tanggal	: 22 Oktober 2025
Materi	: OOP
NPM	: 50422657
Nama	: Harry Mardika
Ketua Asisten	: Satya Bara Justitia
Paraf Asisten	: -
Nama Asisten	: 1.
	2.
	3.
Jumlah Lembar	: ... Lembar

LABORATORIUM INFORMATIKA

UNIVERSITAS GUNADARMA

2025

Soal

1. Jelaskan konsep - konsep utama dalam object oriented programming (OOP) seperti encapsulation, inheritance, dan polymorphism. Berikan contoh penerapan setiap konsep ini dalam program Java yang dikerjakan pada ACT.

2. Jelaskan logika program pada ACT

Penjelasan

---

1. Dalam program Java pada ACT, penerapan konsep–konsep utama OOP terlihat jelas melalui penggunaan enkapsulasi (encapsulation), pewarisan (inheritance), dan polimorfisme (polymorphism). Berikut penjelasan masing-masing konsep beserta penerapannya dalam program tersebut:

a. Encapsulation (Enkapsulasi)

Enkapsulasi adalah prinsip penyembunyian data (data hiding) dalam OOP, di mana atribut suatu kelas dibuat private agar tidak bisa diakses langsung dari luar kelas. Akses terhadap data tersebut hanya dimungkinkan melalui method publik seperti getter dan setter. Tujuannya adalah untuk menjaga keamanan data serta mengontrol bagaimana data diubah atau diambil dari luar kelas.

Pada program ini, kelas User menerapkan konsep enkapsulasi dengan menjadikan atribut–atribut seperti username, password, dan role bersifat private. Kelas ini juga menyediakan method publik seperti getter dan setter agar atribut tersebut dapat diakses dan dimodifikasi secara aman oleh kelas lain.

Melalui penerapan ini, data pribadi setiap pengguna (baik admin, seller, maupun buyer) tidak dapat dimanipulasi secara langsung, melainkan harus melalui mekanisme resmi yang telah ditentukan di kelas User. Dengan demikian, program menjadi lebih aman, terstruktur, dan mudah dalam pemeliharaan.

b. Inheritance (Pewarisan)

Inheritance memungkinkan suatu kelas (kelas anak) untuk mewarisi atribut dan method dari kelas lain (kelas induk). Dengan pewarisan, kode dapat digunakan kembali tanpa perlu menulis ulang, serta memungkinkan pembentukan hierarki antar kelas yang lebih terorganisir.

Dalam program ini, kelas User berperan sebagai kelas induk (superclass) yang menyimpan atribut dan method dasar yang dimiliki oleh semua jenis pengguna. Kemudian, kelas Admin, Seller, dan Buyer masing–masing menjadi kelas turunan (subclass) yang mewarisi seluruh atribut dan method dari User.

Melalui inheritance ini:

- Admin memiliki kemampuan untuk melakukan fungsi manajemen tertentu.
- Seller memiliki kemampuan untuk menampilkan dan mengelola data penjualan.
- Buyer memiliki kemampuan untuk melihat dan melakukan pembelian.

Meskipun ketiganya mewarisi sifat dasar dari User, masing-masing kelas dapat menambahkan fungsionalitas khusus yang sesuai dengan perannya. Hal ini menunjukkan bahwa inheritance digunakan untuk membangun struktur hierarki pengguna yang efisien dan modular.

c. Polymorphism (Polimorfisme)

Polimorfisme berarti “banyak bentuk”. Dalam konteks OOP, ini berarti method yang sama dapat memiliki perilaku berbeda tergantung dari objek yang memanggilnya. Dengan

demikian, Java memungkinkan satu referensi kelas induk untuk merujuk ke berbagai objek kelas turunannya dan tetap mengeksekusi perilaku yang sesuai dengan jenis objek sebenarnya.

Program ini memanfaatkan polimorfisme pada method yang diturunkan dari User, khususnya pada method seperti menu atau tampilan interaksi pengguna. Masing-masing kelas turunan (Admin, Seller, dan Buyer) memiliki implementasi versi sendiri dari method yang sama, sesuai dengan peran dan kebutuhan mereka.

Ketika objek dibuat dan dipanggil melalui referensi bertipe User, Java secara otomatis akan menjalankan versi method yang sesuai dengan tipe objek aktualnya, bukan tipe referensinya. Dengan demikian, saat pengguna login sebagai admin, seller, atau buyer, sistem akan menampilkan menu yang berbeda meskipun method yang dipanggil sama. Ini merupakan bentuk runtime polymorphism, yang membuat program fleksibel dan mudah dikembangkan tanpa harus menulis logika kondisi berulang di satu tempat.

2. Pada program ACT adalah simulasi sederhana sistem pengguna dengan beberapa peran: User (dasar), Buyer (pembeli), Seller (penjual), dan Admin (administrator). Ada satu titik masuk program (kelas utama) yang membuat instance tiap peran dan memanggil method-method yang merepresentasikan aksi (login, logout, menambah produk, dsb.). Program menekankan demonstrasi konsep OOP (pewarisan, overriding, enkapsulasi) melalui perilaku-perilaku ini.

1. Kelas dasar (User)

Berfungsi sebagai representasi umum pengguna, menyediakan perilaku dasar yang umum dimiliki semua pengguna, misalnya kemampuan untuk “login” dan “logout”, tidak menyimpan atau memanipulasi data peran-spesifik pada level ini (hanya menyediakan perilaku generik).

2. Kelas Buyer (turunan dari User)

Mengambil sifat dasar dari kelas User tetapi menyesuaikan (meng-override) perilaku login dan logout agar output atau tindakan menjadi spesifik untuk “Buyer”. Tidak menambahkan atribut domain-spesifik selain menyesuaikan pesan/aksi login-logout.

3. Kelas Seller (turunan dari User)

Selain mewarisi perilaku login/logout, menambahkan fungsionalitas spesifik penjual, yaitu kemampuan untuk menambahkan produk. Memiliki atribut internal untuk menyimpan produk yang ditambahkan; atribut ini disembunyikan (private) sehingga hanya dapat diubah melalui method yang tersedia (contoh enkapsulasi). Method menambah produk juga menampilkan pemberitahuan (log) bahwa produk berhasil ditambahkan.

4. Kelas Admin

Merepresentasikan admin dengan kemampuan login/logout serta kemampuan administratif tambahan (mis. mengelola pengguna). Dalam implementasimu, Admin berdiri sendiri (tidak pernah di-set sebagai turunan User), tetapi memiliki method yang mirip (login/logout) dan method tambahan untuk fungsi admin. Perilaku Admin juga menampilkan pesan yang sesuai ketika method dipanggil.

5. Kelas Main (entry point)

Bertanggung jawab membuat instance dari setiap kelas peran (User, Buyer, Seller, Admin) sesuai urutan yang ditentukan. Memanggil method-method masing-masing objek untuk mensimulasikan skenario: login, logout, menambah produk, dsb. Tidak ada logika kondisional kompleks atau loop program berjalan sekuensial, memanggil aksi satu per satu untuk tiap objek.