Muhammad Aqil Karomy
H1D024096
Shift KRS C
Shift Baru G

Pertemuan 8

DESKRIPSI TUGAS:
Membuat sistem manajemen SDM untuk programmer magang yang mengimplementasikan dua interface sekaligus: KaryawanKontrak (untuk perhitungan gaji dan kontrak) dan AksesSistem (untuk login dan autorisasi). Program ini mendemonstrasikan konsep multiple interface implementation, default method, dan method overriding dalam konteks interface. Class ProgrammerMagang harus memenuhi kontrak dari kedua interface dengan implementasi spesifiknya.

KODE PROGRAM:

1. Interface AksesSistem.java

```
public interface AksesSistem {
   void login(String pin);
   void logout();

   default String getRoleAkses() {
      return "Staff Biasa";
   }
}
```

2. Interface KaryawanKontrak.java

```
public interface KaryawanKontrak {
   double hitungGaji(int jamKerja);
   void perpanjangKontrak(int durasiBulan);

   default String getStatusCuti() {
      return "Tersedia 12 hari";
   }
}
```

3. Class ProgrammerMagang.java

```
public class ProgrammerMagang implements KaryawanKontrak, AksesSistem {
```

```
    private String nama;
    private double gajiPerJam;
    private String pinRahasia;
    private boolean sedangLogin;

    public ProgrammerMagang(String nama, double gajiPerJam, String pinRahasia) {
        this.nama = nama;
        this.gajiPerJam = gajiPerJam;
        this.pinRahasia = pinRahasia;
        this.sedangLogin = false;
    }

    @Override
    public double hitungGaji(int jamKerja) {
        double totalGaji = jamKerja * gajiPerJam;
        System.out.println("Gaji " + nama + " untuk " + jamKerja + " jam kerja: Rp " + totalGaji);
        return totalGaji;
    }

    @Override
    public void perpanjangKontrak(int durasiBulan) {
        System.out.println("Kontrak " + nama + " diperpanjang selama " + durasiBulan + "
bulan.");
    }

    @Override
    public String getStatusCuti() {
        return "Tersedia 5 hari";
    }

    @Override
    public void login(String pin) {
        if (pin.equals(pinRahasia)) {
            sedangLogin = true;
            System.out.println("Login berhasil! Selamat datang, " + nama + ".");
        } else {
            System.out.println("Login gagal! PIN salah.");
        }
    }

    @Override
```

```
    public void logout() {
        sedangLogin = false;
        System.out.println(nama + " telah logout dari sistem.");
    }

    @Override
    public String getRoleAkses() {
        return "Magang IT";
    }
}
```

4. Main Class UjiSDM.java

```
public class UjiSDM {
    public static void main(String[] args) {
        System.out.println("=== SISTEM MANAJEMEN SDM ===");
        System.out.println();

        ProgrammerMagang andi = new ProgrammerMagang("Andi", 50000.0, "1234");

        System.out.println("--- PERHITUNGAN GAJI ---");
        andi.hitungGaji(160);
        System.out.println();

        System.out.println("--- STATUS CUTI ---");
        System.out.println("Status Cuti: " + andi.getStatusCuti());
        System.out.println();

        System.out.println("--- UJI LOGIN (PIN SALAH) ---");
        andi.login("9999");
        System.out.println();

        System.out.println("--- UJI LOGIN (PIN BENAR) ---");
        andi.login("1234");
        System.out.println();

        System.out.println("--- ROLE AKSES ---");
        System.out.println("Role Akses: " + andi.getRoleAkses());
        System.out.println();

        System.out.println("--- PERPANJANG KONTRAK ---");
```

```
        andi.perpanjangKontrak(6);
        System.out.println();

        System.out.println("--- LOGOUT ---");
        andi.logout();
    }
}
```

OUTPUT PROGRAM:

```
=== SISTEM MANAJEMEN SDM ===

--- PERHITUNGAN GAJI ---
Gaji Andi untuk 160 jam kerja: Rp 8000000.0

--- STATUS CUTI ---
Status Cuti: Tersedia 5 hari

--- UJI LOGIN (PIN SALAH) ---
Login gagal! PIN salah.

--- UJI LOGIN (PIN BENAR) ---
Login berhasil! Selamat datang, Andi.

--- ROLE AKSES ---
Role Akses: Magang IT

--- PERPANJANG KONTRAK ---
Kontrak Andi diperpanjang selama 6 bulan.

--- LOGOUT ---
Andi telah logout dari sistem.
```

PENJELASAN ALUR PROGRAM:

1. Deklarasi Interface AksesSistem
   Interface ini mendefinisikan kontrak untuk sistem login dan autorisasi. Berisi 2 abstract method (login, logout) dan 1 default method (getRoleAkses). Default method memberikan implementasi default yang mengembalikan "Staff Biasa", tetapi bisa di-override oleh class yang implements interface ini.

2. Deklarasi Interface KaryawanKontrak

Interface ini mendefinisikan kontrak untuk manajemen karyawan kontrak. Berisi 2 abstract method (hitungGaji, perpanjangKontrak) dan 1 default method (getStatusCuti). Default method memberikan implementasi default "Tersedia 12 hari" yang bisa di-override untuk kasus khusus seperti magang.

3. Multiple Interface Implementation

Class ProgrammerMagang mengimplementasikan kedua interface dengan keyword "implements KaryawanKontrak, AksesSistem". Ini mendemonstrasikan multiple interface implementation di Java. Class ini wajib menyediakan implementasi untuk semua abstract method dari kedua interface (total 4 method: hitungGaji, perpanjangKontrak, login, logout).

4. Atribut dan Constructor

Class memiliki 4 atribut private: nama, gajiPerJam, pinRahasia, dan sedangLogin. Constructor menerima 3 parameter dan menginisialisasi sedangLogin dengan false sebagai default state (belum login).

5. Implementasi Method dari KaryawanKontrak

- hitungGaji(): Menghitung gaji berdasarkan jam kerja × gaji per jam, menampilkan hasil, dan mengembalikan nilai double.
- perpanjangKontrak(): Menampilkan pesan konfirmasi perpanjangan kontrak dengan durasi yang diberikan.
- getStatusCuti(): Meng-override default method untuk mengembalikan "Tersedia 5 hari" (khusus magang, lebih sedikit dari staff biasa).

6. Implementasi Method dari AksesSistem

- login(): Memvalidasi PIN yang diterima dengan pinRahasia. Jika cocok, set sedangLogin = true dan tampilkan pesan sukses. Jika tidak, tampilkan pesan gagal.
- logout(): Set sedangLogin = false dan tampilkan pesan logout.
- getRoleAkses(): Meng-override default method untuk mengembalikan "Magang IT" (role spesifik untuk magang programmer).

7. Testing di Main Class

Program membuat objek ProgrammerMagang dengan data Andi (gaji per jam 50000, PIN 1234). Kemudian menguji semua fungsi secara berurutan: perhitungan gaji untuk 160 jam, cek status cuti, login dengan PIN salah (gagal), login dengan PIN benar (berhasil), cek role akses, perpanjang kontrak 6 bulan, dan logout.

PENJELASAN FUNGSI METHOD:

1. login(String pin)

Fungsi: Memvalidasi autentikasi user berdasarkan PIN. Mengubah state internal (sedangLogin) berdasarkan hasil validasi. Memberikan feedback kepada user tentang sukses/gagal login.

2. logout()
Fungsi: Mengakhiri sesi user dengan mengubah state sedangLogin menjadi false. Penting untuk keamanan sistem.

3. hitungGaji(int jamKerja)
Fungsi: Menghitung total gaji berdasarkan jam kerja. Mengembalikan nilai untuk kemungkinan perhitungan lebih lanjut (seperti pajak atau bonus).

4. perpanjangKontrak(int durasiBulan)
Fungsi: Menangani proses perpanjangan kontrak karyawan. Dalam implementasi lengkap bisa update database atau generate dokumen kontrak.

5. getStatusCuti()
Fungsi: Memberikan informasi status cuti yang tersedia. Di-override untuk memberikan nilai spesifik magang (5 hari) yang berbeda dari default interface (12 hari).

6. getRoleAkses()
Fungsi: Mengembalikan role atau level akses user dalam sistem. Di-override untuk memberikan role spesifik "Magang IT" yang berbeda dari default "Staff Biasa".

KONSEP OOP YANG DITERAPKAN:

1. Multiple Interface Implementation
Java mendukung class untuk mengimplementasikan lebih dari satu interface sekaligus. Ini adalah solusi Java untuk mendapatkan benefit multiple inheritance tanpa masalah diamond problem. ProgrammerMagang implements KaryawanKontrak dan AksesSistem, artinya objek tersebut memiliki dua role atau capability sekaligus: sebagai karyawan kontrak DAN sebagai user yang bisa akses sistem.

2. Default Method (Java 8+)
Default method adalah method di interface yang memiliki implementasi default. Fitur ini diperkenalkan untuk backward compatibility - memungkinkan menambah method baru ke interface tanpa breaking existing implementations. Default method bisa di-override jika class membutuhkan behavior khusus, atau digunakan apa adanya jika implementasi default sudah sesuai. Dalam program ini, getStatusCuti() dan getRoleAkses() adalah default method yang di-override untuk customization.

3. Interface Segregation

Kedua interface (KaryawanKontrak dan AksesSistem) memiliki concern yang berbeda dan terpisah. KaryawanKontrak fokus pada aspek employment, AksesSistem fokus pada aspek security. Ini mengikuti Interface Segregation Principle (ISP) dari SOLID: client shouldn't be forced to depend on interfaces they don't use. Dengan pemisahan ini, class lain bisa implements hanya salah satu interface jika hanya membutuhkan satu capability.

4. Encapsulation

Semua atribut (nama, gajiPerJam, pinRahasia, sedangLogin) bersifat private. Akses ke data ini hanya melalui method yang terkontrol. Atribut pinRahasia dan sedangLogin sangat krusial untuk security dan tidak boleh diakses langsung dari luar class.

5. Polymorphism via Interface

Interface memungkinkan polymorphism. Kita bisa mendeklarasikan variabel dengan tipe interface dan assign objek dari class yang implements interface tersebut:

KaryawanKontrak karyawan = new ProgrammerMagang(...);

AksesSistem user = new ProgrammerMagang(...);

Ini memberikan fleksibilitas untuk menukar implementasi tanpa mengubah code yang menggunakan interface tersebut.

6. Contract-Based Programming

Interface adalah kontrak. Dengan mengimplementasikan interface, class berkomitmen untuk menyediakan semua method yang dideklarasikan di interface. Compiler memastikan kontrak ini dipenuhi. Jika ada method yang tidak diimplementasikan, akan terjadi compile error (kecuali class dideklarasikan abstract).

7. @Override Annotation

Annotation @Override digunakan pada semua method yang meng-implement atau meng-override method dari interface atau superclass. Ini adalah safety check yang membantu compiler mendeteksi error jika signature method tidak sesuai dengan yang ada di interface.

8. State Management

Atribut sedangLogin adalah contoh state management. State ini berubah berdasarkan action (login/logout) dan mempengaruhi behavior sistem. Dalam implementasi yang lebih lengkap, state ini bisa digunakan untuk authorization check sebelum mengizinkan operasi tertentu.

KEUNTUNGAN MULTIPLE INTERFACE:

1. Role-Based Design

Object bisa memiliki multiple role atau capability. ProgrammerMagang berperan sebagai karyawan kontrak dan user sistem sekaligus. Ini lebih natural dan fleksibel dibanding single inheritance hierarchy.

2. Loose Coupling

Code yang menggunakan interface KaryawanKontrak tidak perlu tahu tentang AksesSistem, dan sebaliknya. Ini menciptakan loose coupling antar komponen sistem.

3. Reusability

Interface bisa digunakan ulang oleh banyak class yang berbeda. Interface AksesSistem bisa diimplementasikan oleh ProgrammerMagang, Manager, Admin, dll dengan implementasi yang berbeda-beda.

4. Testability

Interface memudahkan testing karena kita bisa membuat mock implementation untuk testing tanpa membutuhkan class konkret yang lengkap.

5. Flexibility

Mudah menambah capability baru dengan implements interface tambahan tanpa mengubah inheritance hierarchy. Ini mendukung Open/Closed Principle: open for extension, closed for modification.

DEFAULT METHOD VS ABSTRACT METHOD:

1. Abstract Method

Tidak memiliki implementasi sama sekali. Semua class yang implements interface WAJIB menyediakan implementasi. Digunakan ketika behavior harus didefinisikan oleh masing-masing class karena implementasinya sangat spesifik dan berbeda-beda.

2. Default Method

Memiliki implementasi default di interface. Class yang implements interface bisa menggunakan implementasi default atau meng-override jika butuh behavior khusus. Digunakan ketika ada behavior umum yang cocok untuk kebanyakan implementasi, tetapi tetap memberi fleksibilitas untuk customization. Sangat berguna untuk backward compatibility ketika menambah method baru ke interface yang sudah ada.

KESIMPULAN:

Program ini mendemonstrasikan kekuatan multiple interface implementation dalam Java. Dengan mengimplementasikan KaryawanKontrak dan AksesSistem secara bersamaan, class ProgrammerMagang memiliki dua set capabilities yang berbeda namun saling melengkapi.

Default method memberikan fleksibilitas tambahan dengan menyediakan implementasi default yang bisa di-override sesuai kebutuhan. Pendekatan ini menghasilkan design yang modular, fleksibel, dan mudah di-maintain. Interface segregation memungkinkan pemisahan concern yang jelas, sementara multiple interface implementation memberikan cara untuk mengombinasikan multiple capabilities tanpa masalah complexity yang muncul pada multiple inheritance. Ini adalah pattern yang sangat powerful dalam software design, terutama untuk sistem yang kompleks dengan berbagai role dan responsibility.