

LAPORAN PERTEMUAN 8

NAMA : AHMAD FIKRI ZAKARIA
NIM : H1D024062
MATERI : MULTIPLE INTERFACE

A. Alur Kerja (Workflow)

Pada pertemuan ini, saya menyelesaikan studi kasus final "Manajemen SDM Programmer Magang" yang mensimulasikan peran ganda seorang magang (sebagai karyawan kontrak dan pengguna sistem). Solusi ini menerapkan **Multiple Interfaces** untuk mengatasi keterbatasan *Single Inheritance* di Java. Langkah pengerjaan meliputi:

1. **Perancangan Interface:**
 - o Membuat interface KaryawanKontrak dengan method abstrak hitungGaji() dan perpanjangKontrak(), serta method default getStatusCuti().
 - o Membuat interface AksesSistem dengan method abstrak login() dan logout(), serta method default getRoleAkses().
2. **Implementasi Class (ProgrammerMagang):**
 - o Class ini mengimplementasikan **dua interface sekaligus** (implements KaryawanKontrak, AksesSistem).
 - o Mengisi logika untuk semua method abstrak yang diwajibkan oleh kedua interface.
 - o Melakukan **Override pada Default Method**: Mengubah logika getStatusCuti (dari 12 hari menjadi 5 hari) dan getRoleAkses (dari Staff Biasa menjadi Magang IT) sesuai aturan bisnis untuk anak magang.
3. **Pengujian (UjiSDM):**
 - o Membuat objek ProgrammerMagang dan mengujinya secara komprehensif mulai dari perhitungan gaji, validasi login, pengecekan hak akses, hingga perpanjangan kontrak.

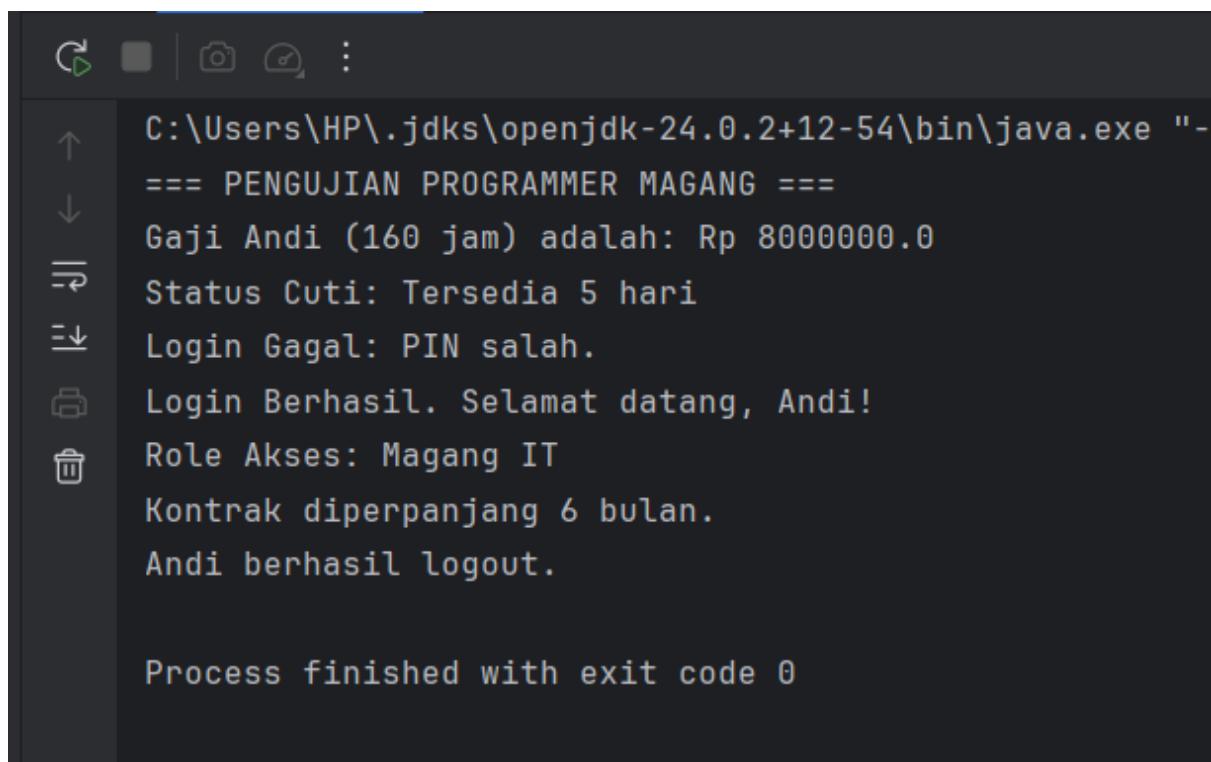
B. Penjelasan Kode & Fungsi

Berikut adalah analisis mendalam mengenai konsep teknis yang diterapkan dalam kode program:

1. **Multiple Interfaces Implementation** Java tidak mengizinkan sebuah class mewarisi lebih dari satu class induk (extends A, B = Error). Namun, realitasnya sebuah objek seringkali memiliki banyak peran. Solusinya adalah implements InterfaceA, InterfaceB. Class ProgrammerMagang membuktikan hal ini dengan sukses menjalankan peran ganda sebagai karyawan (mengurus kontrak/gaji) dan user sistem (login/logout) dalam satu entitas.
2. **Default Method (Java 8 Feature)** Fitur default pada interface memungkinkan kita menyediakan implementasi dasar.
 - o **Fleksibilitas:** Jika class implementer cocok dengan logika default (misal staff biasa), ia tidak perlu menulis kode apa-apa.
 - o **Overriding:** Jika class implementer butuh logika khusus (seperti ProgrammerMagang yang jatah cutinya beda), ia bisa melakukan override method default tersebut layaknya method biasa.
3. **Kontrak & Implementasi** Kode ini memisahkan definisi (Interface) dan eksekusi (Class). KaryawanKontrak hanya peduli bahwa karyawan *harus* digaji, tapi tidak peduli rumusnya. ProgrammerMagang-lah yang menentukan bahwa gajinya dihitung per jam (jam * tarif).

C. Hasil Output Program

Berikut adalah tangkapan layar/teks hasil eksekusi program UjiPengguna.java:



```
C:\Users\HP\.jdks\openjdk-24.0.2+12-54\bin\java.exe "-  
== PENGUJIAN PROGRAMMER MAGANG ==  
Gaji Andi (160 jam) adalah: Rp 8000000.0  
Status Cuti: Tersedia 5 hari  
Login Gagal: PIN salah.  
Login Berhasil. Selamat datang, Andi!  
Role Akses: Magang IT  
Kontrak diperpanjang 6 bulan.  
Andi berhasil logout.  
  
Process finished with exit code 0
```