NAMA: SITI NURVIATIKA

KELAS: TI22A

NIM: 20220040281

TUGAS 3

PEMROGRAMAN BERBASIS OBJECT

Tugas:

1. Kode Program diupload ke github dengan nama repository praktikum-inheritance
   Jawab: https://github.com/nurviatika/praktikum-inheritance
2. Berikan analisa setiap percobaan dalam bentuk File teks pdf dan upload juga ke github praktikum-inheritance

   Percobaan 1:
   - Pada kelas `Parent` terdapat satu variabel anggota `x` yang diinisialisasi dengan nilai 5.
   - Kelas `Child` mewarisi `x` dari kelas `Parent`, lalu mendeklarasikan variabel `x` baru dengan nilai 10.
   - terdapat metode `Info(int x)` dalam kelas `Child`, yang mencetak nilai `x` sebagai parameter, nilai `x` yang ada di kelas `Child`, dan nilai `x` yang diwarisi dari kelas `Parent` menggunakan kata kunci `super`.
   - Pada fungsi `main`, sebuah objek dari kelas `Child` dibuat dan metode `Info(20)` dipanggil dengan argumen 20.
   - Output yang dihasilkan adalah:
     Nilai x sebagai parameter =20
     Data member x di class child =10
     Data member x di class parent =5
   - **Penjelasan output:**
     - "Nilai x sebagai parameter =20": Mencetak nilai `x` yang diberikan saat memanggil metode `Info`.
     - "Data member x di class child =10": Mencetak nilai `x` dari kelas `Child`.
     - "Data member x di class parent =5": Mencetak nilai `x` dari kelas `Parent` menggunakan kata kunci `super`.

   Percobaan 2:
   Terjadi error sebab class Manajer ingin mengakses varibael nama yang dideklarasikan menjadi private pada kelas Pegawai
   - `nama`: bertipe data `String` bersifat private, Maksudnya hanya bisa diakses di dalam kelas pegawai saja
   - `gaji`: bertipe data `double` yang bersifat public. Dimana nilai gaji dapat diakses dan diubah dari luar kelas Pegawai.
   - `getNama()`: Ini adalah metode getter untuk variabel `nama`. Metode ini mengembalikan nilai dari variabel `nama` saat ini. Karena `nama` bersifat private, kita perlu metode ini untuk mengakses nilainya dari luar kelas.
   - `setNama': ialah metode setter untuk variabel `nama`. dimana untuk mengubah nilai dari variabel `nama` dengan nilai yang diberikan sebagai parameternya.

- Memiliki atribut publik `departemen` yang bertipe data String.
- Tidak ada konstruktor yang didefinisikan secara eksplisit, sehingga menggunakan konstruktor bawaan (default) yang disediakan oleh Java.
- Memiliki metode `isiData` yang menerima dua parameter, yaitu `n` (nama) dan `d` (departemen).
- Dalam metode `isiData`, nilai dari parameter `n` digunakan untuk mengatur nilai nama menggunakan setter `setNama(n)` yang mungkin merupakan metode yang diwarisi dari kelas `Pegawai`.
- Nilai dari parameter `d` kemudian disimpan dalam atribut `departemen` dari objek kelas `Manajer`.

Percobaan 3:

- Error terjadi pada saat membuat kelas Child (turunan) dan kelas Parent (Induk). Java akan memanggil konstruktornya dari sebuah kelas parent02, tetapi kelas parent02 tidak memiliki konstruktor.
- Kelas Parent02 tidak memiliki atribut atau metode apapun didalamnya
- Kelas Child02 ialah turunan dari kelas `Parent02`, yang diasumsikan ada tetapi tidak ditampilkan dalam potongan kode yang diberikan.
- Ada konstruktor tanpa parameter yang tidak langsung menginisialisasi variabel `x` dengan nilai 5. Ini dilakukan dalam konstruktor menggunakan pernyataan `x = 5;`.
- Saat objek dari kelas `Child2` dibuat, konstruktor ini akan dieksekusi, dan nilai `x` akan diatur menjadi 5.

Percobaan 4:

Kelas Employee:

- Memiliki beberapa atribut pribadi, yaitu `Name`, `Salary`, dan `birthDate`.
- Atribut `BASE_SALARY` ialah konstanta dengan nilai 15000.00 yang bersifat statis.
- Konstruktor pertama tanpa parameter.
- Konstruktor kedua dengan parameter nama, gaji, dan tanggal lahir.
- Konstruktor ketiga dengan parameter nama dan gaji, dengan tanggal lahir default ke `null`.
- Konstruktor keempat dengan parameter nama dan tanggal lahir, dengan gaji default dari `BASE_SALARY`.
- Konstruktor kelima dengan parameter nama saja, dengan gaji default dari `BASE_SALARY`
- Kelas Employee memiliki metode-metode getter untuk mendapatkan nilai dari atribut `Name` dan `Salary`.

Kelas Manager:

- Ialah sebuah turunan dari kelas Employee.
- Memiliki atribut tambahan yaitu `department`.
- Konstruktor pertama dengan parameter nama, gaji, dan departemen, memanggil konstruktor superclass Employee yang sesuai.
- Konstruktor kedua dengan parameter nama dan departemen, memanggil konstruktor superclass Employee yang sesuai.
- Konstruktor ketiga dengan parameter departemen saja, memanggil konstruktor superclass Employee yang tanpa parameter.
- Memiliki metode `GetDept()` untuk mendapatkan nilai dari atribut `department`.

Kelas TestManager:

- Di dalam metode `main()`, objek Manager `Utama` pertama kali dibuat dengan memanggil konstruktor yang menerima parameter nama, gaji, dan departemen.
- Kemudian, informasi tentang objek `Utama` dicetak, yaitu nama, gaji, dan departemen.
- Selanjutnya, objek `Utama` diinisialisasi kembali dengan memanggil konstruktor yang menerima parameter nama dan departemen saja, dan informasi baru tentang objek `Utama` dicetak lagi.

Percobaan 5:

- Kode ini mengimplementasikan tentang konsep pewarisan, di mana kelas turunan (SadObject dan HappyObject) mewarisi sifat dan perilaku dari kelas induk (MoodyObject).
- Polimorfisme juga ditunjukkan di sini, di mana objek m dari tipe MoodyObject bisa ke objek dari kelas turunan yang berbeda (SadObject dan HappyObject
- Method overriding terjadi pada kelas turunan (SadObject dan HappyObject) yang dimana metode getMood() dioverride menghasilkan mood yang sesuai.
- Implementasi dari metode laugh() dan cry() berbeda di setiap kelas turunan, menunjukkan fleksibilitas dalam implementasi yang sesuai dengan kebutuhan kelas tersebut.

Percobaan 6:

- Kelas `A`: memiliki beberapa variabel anggota yang dapat diakses oleh kelas turunannya (`var_a`, `var_b`, `var_c`, `var_d`).
- Kelas `B`ialah turunan dari kelas `A`.
- Konstruktor kelas `B` menginisialisasi nilai dari variabel anggota `var_a` dan `var_b` yang diwarisi dari kelas `A`.
- Di dalam metode `main`, objek `aa` dari kelas `A` dibuat terlebih dahulu.
- Setelah itu, objek `bb` dari kelas `B` dibuat.
- Kemudian, nilai dari variabel anggota objek `aa` dan `bb` dicetak.

Percobaan 7:

Kelas Bapak:

- Memiliki dua variabel anggota (atribut) yaitu `a` dan `b`.
- Memiliki metode `show_variabel()` yang mencetak nilai dari variabel `a` dan `b`.

Kelas Anak:

- Merupakan turunan dari kelas Bapak.
- Memiliki tambahan satu variabel anggota yaitu `c`.
- Override metode `show_variabel()` yang sudah ada di kelas Bapak. Dalam metode ini, selain mencetak nilai `a` dan `b` dari kelas Bapak, juga mencetak nilai `c` dari kelas Anak.

Kelas InheritExample:

- Memiliki metode `main`, yang merupakan metode utama dari program.
- Di dalam metode `main`, dibuat dua objek: `objectBapak` dari kelas Bapak dan `objectAnak` dari kelas Anak.
- Nilai `a` dan `b` dari `objectBapak` diatur masing-masing menjadi 1.
- Metode `show_variabel()` dari `objectBapak` dipanggil, yang akan mencetak nilai `a` dan `b` dari kelas Bapak.
- Nilai `c` dari `objectAnak` diatur menjadi 5.
- Metode `show_variabel()` dari `objectAnak` dipanggil, yang akan mencetak nilai `a`, `b`, dan `c` dari kelas Anak.

Percobaan 8:

Kelas Parent03:

- Kelas ini memiliki satu atribut `ParentName` yang merupakan tipe data String.
- Konstruktor pertama tidak memiliki parameter dan tidak melakukan operasi tertentu.
- Konstruktor kedua menerima satu parameter `ParentName` dan menginisialisasi atribut `ParentName` dengan nilai yang diterima. Selain itu, konstruktor ini mencetak pesan "Konstruktor Parent".

Kelas Baby:

- Kelas ini merupakan turunan dari kelas Parent03.
- Memiliki atribut `babyName` yang juga merupakan tipe data String.
- Ada satu konstruktor dalam kelas ini yang menerima satu parameter `babyName`.
- Konstruktor ini memanggil konstruktor dari kelas induk (`Parent03`) menggunakan kata kunci `super()`. Ini adalah pemanggilan konstruktor kosong dari kelas Parent03.
- Menginisialisasi atribut `babyName` dengan nilai yang diterima.
- Mencetak pesan "Konstruktor Baby" dan nilai `babyName`.
- Memiliki satu metode `cry()` yang mencetak pesan "huhu huhu".