

Nama : Julhan Abdul Malik
NIM : 20220040126
Kelas : TI22A
Program Studi : Teknik Informatika



- Lampiran -

• **Rincian Tugas:**

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

• **Jawaban:**

1. Link GitHub Julhan Abdul Malik -> <https://github.com/JulhanAbdulMalik-TI22A/Praktikum-Inheritance.git>

2. Analisis sebagai berikut:

• Percobaan 1:

Analisis:

- Saat memanggil **this.x**, itu merujuk pada nilai x dari objek saat ini (yaitu objek Child yang baru dibuat).
- Saat memanggil **super.x**, itu merujuk pada nilai x dari kelas induk, yaitu kelas Parent.
- Variabel **x** dalam metode Info akan mengambil nilai yang diteruskan saat metode dipanggil (dengan nilai 20).
- Output:

Nilai x sebagai parameter = 20

Data member x di class child = 10

Data member x di class parent = 5

• Percobaan 2:

Kode yang diberikan itu sudah terjadi erorr karena kelas Manajer mencoba mengakses variabel nama yang dideklarasikan sebagai private di kelas Pegawai.

Analisis:

- Kode yang diberikan menunjukkan penerapan konsep pewarisan (inheritance) yang di mana kelas Manajer mewarisi variabel dan metode dari kelas Pegawai.
- Konsep Enkapsulasi diterapkan dengan mendeklarasikan variabel nama sebagai private, sehingga hanya dapat diakses melalui metode getter dan setter yang telah disediakan.
- Kelas Manajer menambahkan fitur tambahan dalam bentuk variabel anggota departemen, yang tidak dimiliki oleh kelas Pegawai.

- Metode `isiData` (`String n`, `String d`) digunakan untuk mengisi data sekaligus mengatur nilai nama dan departemen dari objek kelas `Manajer`.

- Percobaan 3:

Dalam Kode yang diberikan sudah terjadi error karena ketika membuat sebuah kelas turunan (Child) dari kelas induk (Parent), Java akan mencoba untuk memanggil konstruktor dari kelas induk (`Parent2`). Namun, dalam kelas `Parent2` tidak ada konstruktor yang didefinisikan, dan itu menyebabkan error.

Analisis:

- Kelas `Parent2` adalah kelas dasar atau induk.
- Kelas `Child2` adalah turunan dari kelas `Parent2`. Ini ditunjukkan dengan penggunaan kata kunci `extends` di deklarasi kelas.
- Variabel `x` adalah variabel anggota (`instance variable`) kelas `Child2`, yang memiliki nilai awal 5.
- Konstruktor kelas `Child2` adalah metode khusus yang memiliki nama yang sama dengan nama kelasnya.

- Percobaan 4:

Analisis:

- Kode tersebut menunjukkan penggunaan konsep pewarisan (`inheritance`) di mana kelas `Manager` mewarisi sifat dan perilaku dari kelas `Employee`.
- Konstruktor dalam kelas `Manager` memanfaatkan konstruktor kelas `Employee` menggunakan kata kunci `super()`.
- Setiap konstruktor memiliki fungsi yang berbeda sesuai dengan jumlah parameter yang diberikan dan nilai-nilai yang ingin diinisialisasi.
- Kode tersebut juga menunjukkan penggunaan `overloading` konstruktor untuk memberikan fleksibilitas dalam pembuatan objek.

- Percobaan 5:

Analisis:

- Kode tersebut mengimplementasikan konsep pewarisan (`inheritance`), 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` dapat merujuk ke objek dari kelas turunan yang berbeda (`SadObject` dan `HappyObject`).

- Method overriding terjadi di kelas turunan (SadObject dan HappyObject) di mana metode `getMood()` dioverride untuk 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:

Analisis:

- Kode tersebut menggambarkan konsep pewarisan (inheritance) di mana kelas B mewarisi sifat dan perilaku dari kelas A.
- Konstruktor kelas A akan dieksekusi terlebih dahulu saat membuat objek kelas A maupun kelas B.
- Penggunaan `super()` dalam konstruktor kelas B memungkinkan untuk memanggil konstruktor kelas induk sebelum melakukan inisialisasi kelas turunan.
- Setelah konstruktor kelas B dieksekusi, nilai dari variabel anggota `var_a` dan `var_b` diubah, menunjukkan bahwa kelas turunan dapat mengakses dan memanipulasi variabel anggota dari kelas induk.
- Kode tersebut memberikan contoh sederhana tentang bagaimana pewarisan bekerja dalam Java dan bagaimana konstruktor kelas turunan dapat memanggil konstruktor kelas induk.

- Percobaan 7:

Analisis:

- Kode tersebut mengilustrasikan konsep pewarisan, di mana kelas Anak mewarisi sifat dan perilaku dari kelas Bapak.
- Metode `show_variabel()` di kelas Anak melakukan overriding terhadap metode yang sama di kelas Bapak. Ini artinya, metode yang digunakan adalah milik kelas Anak, bukan kelas Bapak.
- Override method memungkinkan kelas turunan untuk memberikan implementasi yang berbeda terhadap metode yang sudah didefinisikan di kelas induknya.
- Ketika metode `show_variabel()` dipanggil pada objek `objectAnak`, yang dieksekusi adalah metode yang ada di kelas Anak, sehingga mencetak nilai dari variabel `a`, `b`, dan `c`.
- Hal ini menunjukkan bagaimana inheritance memungkinkan penggunaan ulang kode dan memfasilitasi pengembangan software yang lebih terstruktur dan terorganisir.

- Percobaan 8:

Analisis:

- Kode tersebut menunjukkan konsep pewarisan (inheritance) dalam pemrograman berorientasi objek (OOP) di Java, di mana kelas Baby mewarisi sifat dan perilaku dari kelas Parent2.
- Konstruktor Baby memanggil konstruktor superclass Parent2 menggunakan kata kunci `super()`, sehingga konstruktor dari kelas Parent2 akan dieksekusi terlebih dahulu sebelum konstruktor dari kelas Baby.
- Dengan demikian, penggunaan `super()` memungkinkan untuk memanggil konstruktor superclass dan melakukan inisialisasi dari kelas induk sebelum melakukan inisialisasi dari kelas turunannya.
- Override method tidak terjadi dalam contoh tersebut karena tidak ada method dalam kelas Parent2 yang di-overriding oleh kelas Baby.
- Namun, jika ingin melakukan overriding method, Anda dapat menambahkan method di kelas Parent2 yang kemudian di-override oleh kelas Baby.