RELASI

- Object Relationship
- Pewarisan
- Final Specifier
- Warisan Berlian
- Function Overriding
- Virtual dan Polymorphism
- Enkapsulasi
- Abstraksi

OBJECT RELATIONSHIP

object dapat saling berelasi dan kadang relasi tersebut membentuk pola untuk menciptakan *object* yang besar.

Ada 3 macam relasi :

1.Association
2.Compositon
3.Aggregation

COMPOSITION

Composition merupakan relasi dengan *model* “bagian dari”, masing-masing *object* akan saling bergantung, dan hubungan dari mereka akan menentukan jangka hidup masing-masing dari mereka.

Contoh: relasi badan manusia dengan jantung, jika salah satu mati maka yang lainnya tidak akan bisa bertahan.

ASSOSIATION

Association merupakan *object* yang independen tapi berelasi, masing-masing *object* memiliki jangka hidup mereka sendiri dan tidak ada yang namanya kepemilikan.

Contoh : hubungan antara dokter dan pasien, yang masing-masin memiliki jangka hidupnya sendiri, dan mereka dapat memutuskan atau mengganti hubungan kapan saja.

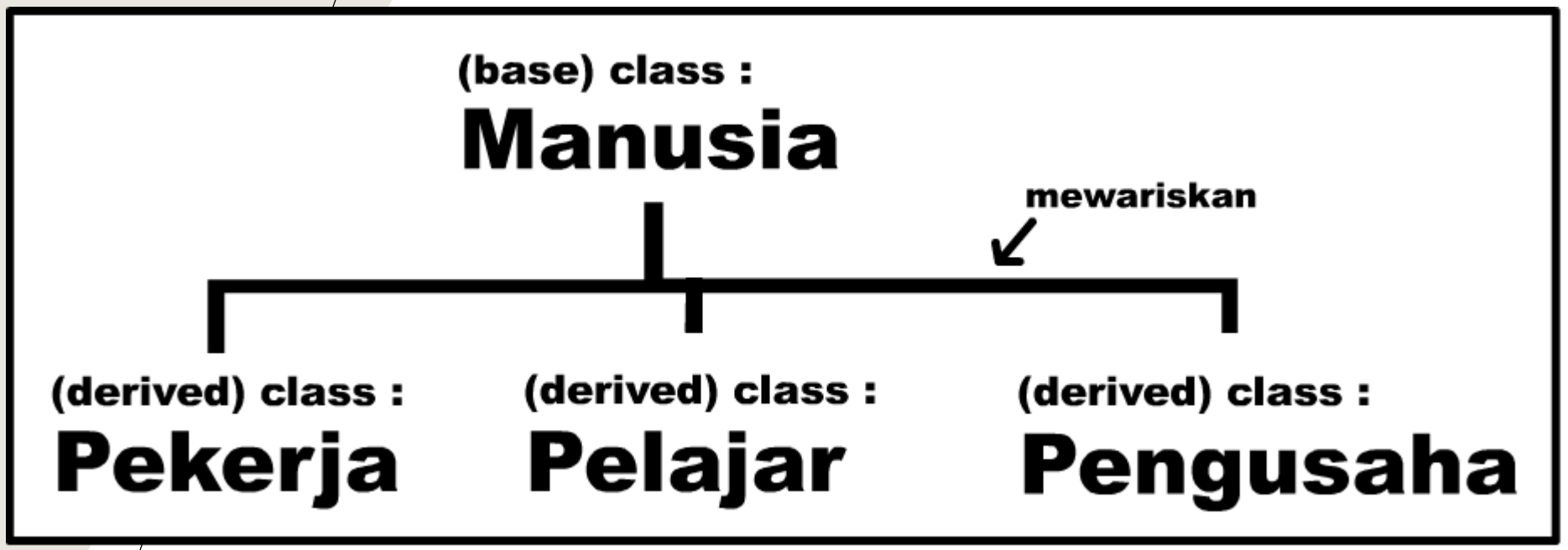
AGGREGATION

Aggregation merupakan relasi dengan model “memiliki”, dimana semua *object* memiliki jangka kehidupanya masing-masing dan adanya sebuah relasi kepemilikan lebih dari satu.

Contoh: hubungan antara anak dan orang tua, dimana masing-masing memiliki jangka kehidupan masing-masing, dan orang tua memiliki (banyak) anak.

PEWARISAN (INHERITANCE)

Inheritance atau Pewarisan adalah salah satu konsep pada object-oriented programming, yang mengadopsi konsep pewarisan yang di miliki oleh *object* pada dunia nyata. Konsep Pewarisan ini memiliki model relasi “adalah”, dimana sebuah *class* akan dimungkinkan untuk mengambil isi dari *class* lain sebagai isi dari class tersebut.



CARA MENDIRIKAN *INHERITACE* / PEWARISAN

Untuk mendirikan sebuah base class, sama seperti layaknya kita membuat class biasa.

Untuk mendirikan derived class dan mewarisi informasi dari base class, mendirikan class sama seperti kita mendirikan class seperti biasanya dan akan dibutuhkan tambahan berupa daftar class penurunan (base class) yang ditulis setelah tanda : .

Sintaks :

```
class nama_derived_class : access_modifier base_class{
//...
};
```

TIPE-TIPE *INHERITANCE* / PEWARISAN

- Public Inheritance : Jika mendaftarkan base class pada derived class menggunakan access specifier public maka akan membuat member dari base class yang memiliki sifat private, protected, dan public akan menjadi diri mereka sendiri pada derived class tersebut.

- Protected Inheritance : jika mendaftarkan base class pada derived class menggunakan access specifier protected maka akan membuat member dari base class yang bersifat protected dan public menjadi bersifat protected pada derived class.

- Private Inheritance : jika mendaftarkan base class pada derived class menggunakan access specifier private maka akan membuat member dari base class yang bersifat protected dan public menjadi bersifat private pada derived class.

MULTIPLE *INHERITANCE* / PEWARISAN

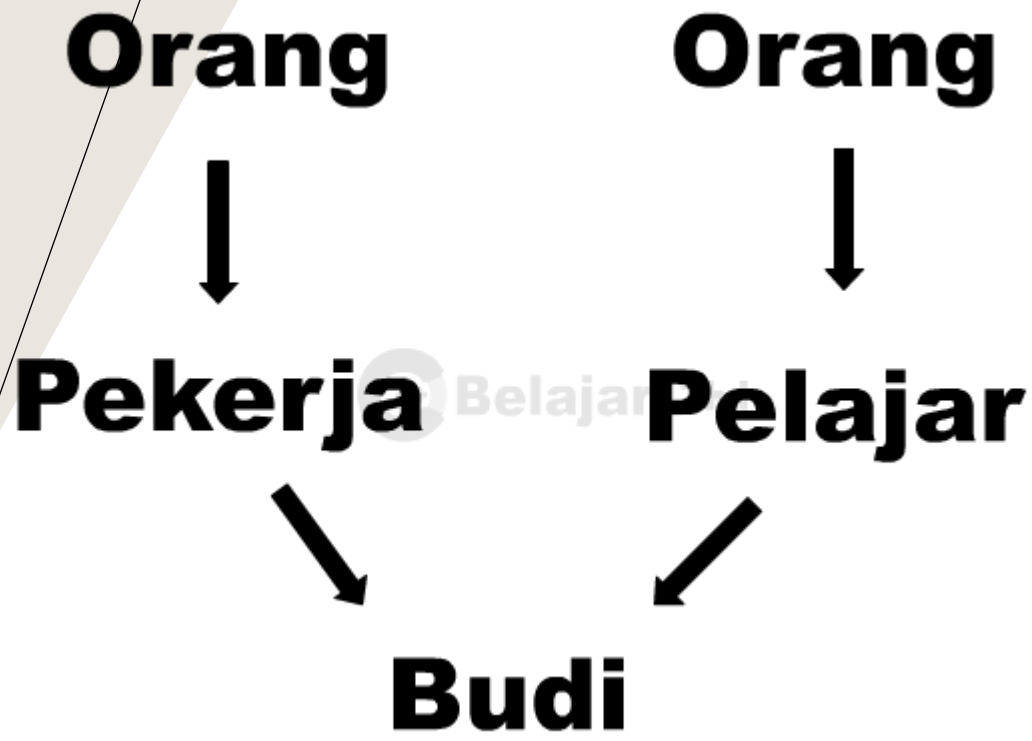
Dalam bahasa pemrograman C++ dimungkinkan untuk mendaftarkan banyak base class pada satu derived class. Cara penulisanya sama seperti kita mendaftarkan satu base class pada satu derived class, hanya daftar dari base class akan ditulis secara sejejar dan dipisahkan dengan tanda koma , .

Contoh :

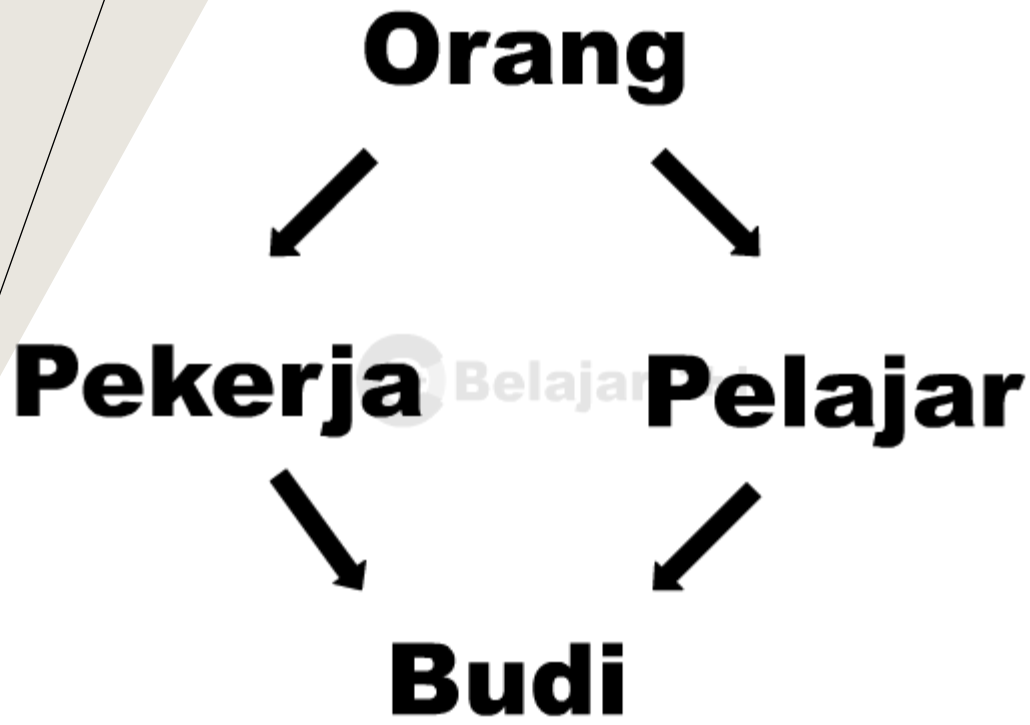
```
class pegawai: public orang, private manusia{
//...
}
```

DIAMOND INHERITANCE (WARISAN BERLIAN)

Dalam hal ini, pewarisan juga bisa memiliki kasus seperti, derived class mendapatkan warisan dari dua base class, dan kenyataannya base class tersebut sebelumnya juga mendapakan warisan dari satu base class.



DIAMOND INHERITANCE DENGAN VIRTUAL INHERITANCE



POLYMORPHISM

Polymorphism adalah sebuah fitur yang ada pada bahasa pemrograman C++. digunakan untuk memberikan kemampuan pada function pada anggota class, untuk memberikan seolah-olah function tersebut memiliki berbagai bentuk yang sesuai berdasarkan class masing-masing.

Ketika menggunakan Polymorphism, jika kita mencoba untuk memanggil sebuah member function, maka program akan merespon dengan member function yang sesuai berdasarkan yang ada pada object dan class yang digunakan.

Untuk membuat function pada class bisa memiliki kemampuan Polymorphism, dibutuhkan kata kunci ***virtual*** sebelum tipe data function pada base class.

Virtual function adalah sebuah member function yang didirikan di dalam base class dan overridden oleh derived class. Ketika anda menunjuk ke suatu object yang dibuat dari derived class dengan menggunakan sebuah object pointer yang dibuat menggunakan base class, anda dapat memanggil sebuah function yang dibuat sebagai virtual function. setiap pemanggilan function, akan mengeksekusi masing-masing function berdasarkan yang dimiliki oleh class yang ditunjuk.

Jika saat anda membuat kemampuan Polymorphism pada class-class anda dan berpikiran bahwa virtual function yang berada di base class sebenarnya tidak membutuhkan definisi, maka anda dapat membuat sebuah “pure vitual function”

KEGUNAAN FINAL SPECIFIER

final adalah sebuah specifier dari C++ yang digunakan untuk mencegah terjadinya ***overriding*** terhadap suatu function pada sebuah class dan dapat pula untuk mencegah terjadinya ***inheritance*** pada suatu class.

ENCAPSULATION (ENKAPSULASI)

Encapsulation adalah sebuah konsep Object Oriented Programming digunakan untuk membungkus data dan fungsi, untuk menjaga tetap terjaga agar tidak adanya penyalahgunaan.

Dengan enkapsulasi pengguna tidak diperbolehkan untuk mengakses data yang disembunyikan secara langsung. tapi bisa menggunakan dan memahami dengan mudah berdasarkan antar muka yang telah disediakan.

Syarat *Encapsulation :*

- Data dan fungsi yang disembunyikan harus berlabel private atau protected(jika dibutuhkan untuk hubungan antar class), agar tidak bisa diakses secara sembarang dan disalahgunakan.
- Data dan fungsi yang digunakan untuk antarmuka harus berlabel public.

ABSTRAKSI (ABSTRACTION)

Abstraksi didefinisikan sebagai penyampaian informasi yang penting saja ke dunia luar sambil menutupi detail latar belakang, yaitu merepresentasikan informasi yang diperlukan dalam program tanpa menunjukkan hal-hal yang spesifik. Abstraksi adalah metode desain dan pemrograman yang memisahkan antarmuka dari implementasi.

Tipe-tipe abstraksi dalam C++

1. ***Abstraksi kontrol*** - Dalam jenis implementasi abstraksi ini, proses disembunyikan dari pengguna.
2. ***Abstraksi data*** adalah proses penyampaian data yang diperlukan ke dunia luar sambil menyembunyikan detail yang diperlukan di dalamnya, yaitu hanya mewakili detail yang diperlukan dalam program. Abstraksi data adalah metode pemrograman yang memisahkan antarmuka program dan detail implementasi.
    a. Abstraksi dengan menggunakan file header
    b. Abstraksi dengan menggunakan Class (akses modifier)