PEMROGRAMAN LANJUT

03 | Inheritance in C++

Team Teaching Pemrograman Lanjut

Fakultas Ilmu Komputer UB

Pengertian dasar inheritance

- Inheritance (Pewarisan) merupakan salah satu dari tiga konsep dasar OOP.
- Konsep inheritance ini mengadopsi dunia riil dimana suatu entitas/obyek dapat mempunyai entitas/obyek turunan.
- Dengan konsep inheritance, sebuah class dapat mempunyai class turunan.

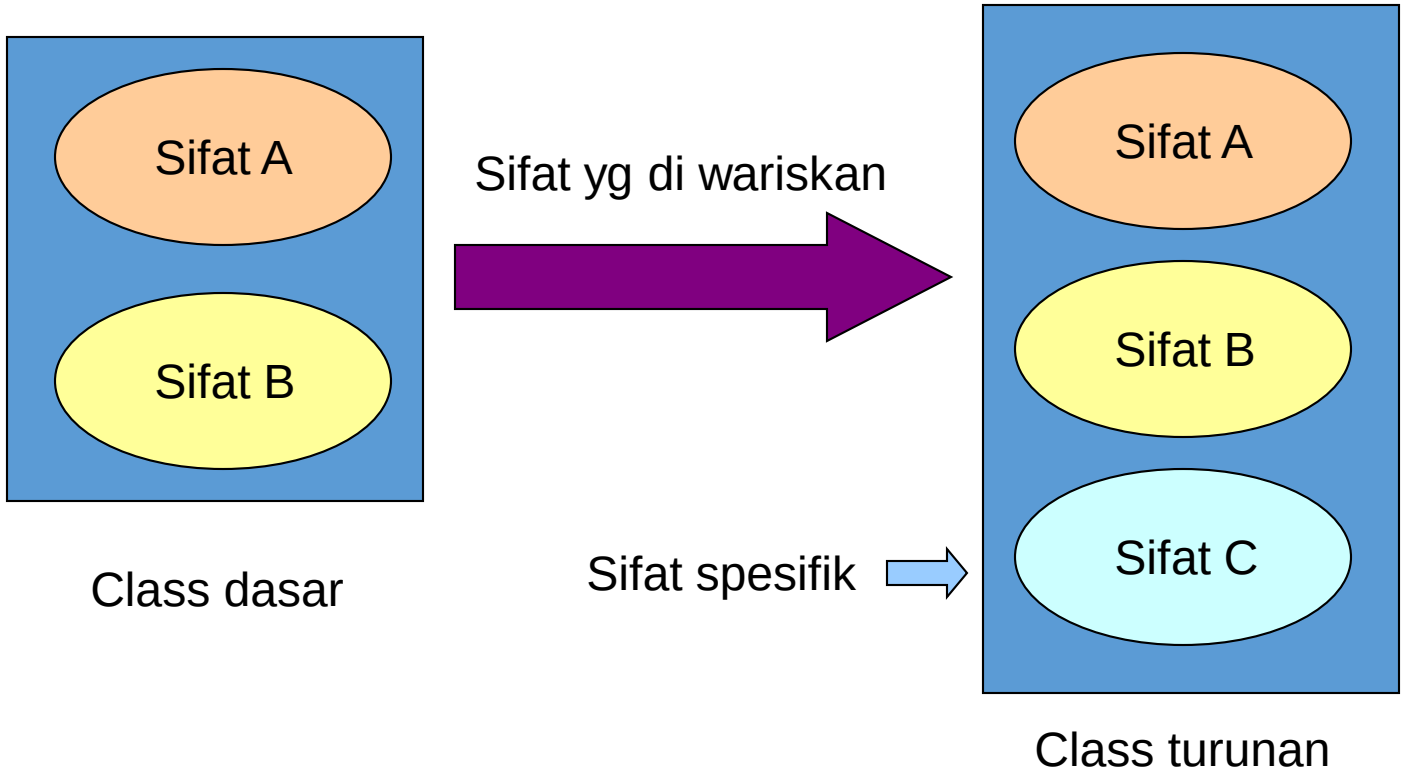
Pengertian dasar inheritance

- Suatu class yang mempunyai class turunan dinamakan parent class atau base class.
- Sedangkan class turunan itu sendiri seringkali disebut subclass atau child class.
- Suatu subclass dapat mewarisi sifat yang dipunyai oleh parent class.

Pengertian dasar inheritance

- Karena suatu subclass dapat mewarisi sifat yang dipunyai oleh parent class-nya, maka member dari suatu subclass adalah terdiri dari sifat yang ia punyai dan juga sifat yang ia warisi dari class parent-nya.
- Kesimpulannya, boleh dikatakan bahwa suatu subclass adalah tidak lain hanya memperluas parent class-nya.

Penggambaran inheritance



Contoh inheritance (public)


```
#include <iostream>
using namespace std;
class Parent{
          public:
                    void infoParent(){cout<<"this parent class"<<endl; }
};
class Child:public Parent{
          public:
                    void infoChild(){ cout<<"this child class"<<endl; }
};
int main(){
          Child anak;
          anak.infoParent();
          anak.infoChild();
return 0;
}
```

Konsep inheritance

- Semua anggota yang bersifat **public** (dan juga **protected**) pada kelas dasar (**Basis**) diwariskan ke kelas turunan (**Turunan**) sebagai anggota yang bersifat **private**

- Konstruktor dan destruktor tidak diwariskan

- Bagian **private** dari suatu kelas **tidak diwariskan** → cara paling mudah mengganti private dengan **public** (shg **anggota data bisa diakses dari fungsi main**())
- **Solusi:** penentu akses **Protected**

Contoh inheritance (protected)

```
#include <iostream>
using namespace std;
class Parent{
          protected:
                    void infoParent(){ cout<<"this parent class"<<endl;}
};
class Child:public Parent{
          public:
                    void infoChild(){
                              infoParent();
                              cout<<"this child class"<<endl;
                    }
};
int main(){
          Child anak;
          anak.infoChild();
          return 0;
}
```

Konsep inheritance

- Anggota data bisa diakses pada kelas Child, tetapi tidak dapat diakses pada fungsi main()

Penentu Pewarisan	**Penentu Akses di Kelas Dasar**	**Akses Baru pada Kelas Turunan**
private	Private Protected Public	Tidak diwariskan Private Private
protected	Private Protected Public	Tidak diwariskan Protected (tetap) Protected
public	Private Protected Public	Tidak diwariskan Protected (tetap) Public (tetap)

Kontruktor dan destructor pada inheritance

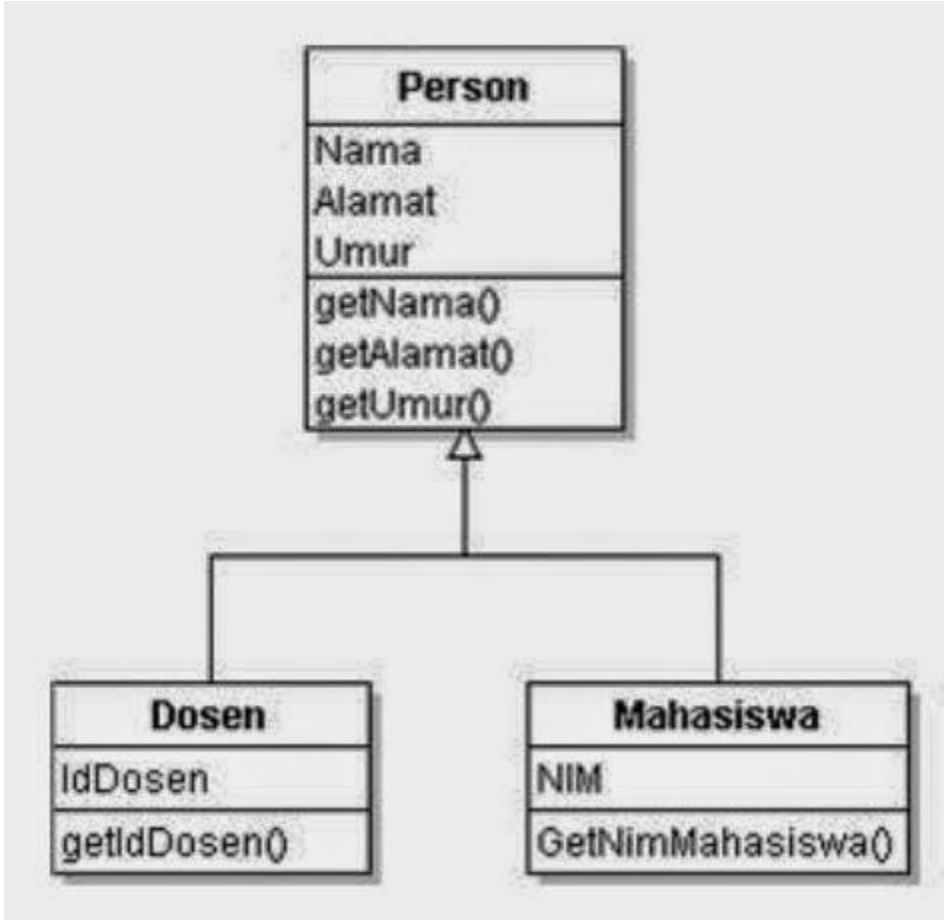
- Pada saat **obyek berkelas Turunan** diciptakan:
  - Konstruktor **kelas dasar** dengan sendirinya dijalankan
  - Kemudian konstruktor **kelas turunan** dijalankan

- Pada saat obyek berakhir:
  - **Destruktor kelas turunan** dijalankan terlebih dulu
  - Kemudian **destruktor kelas dasar** dijalankan

Contoh inheritance (Konstruktor & Destruktor)


```
class Parent{
          public:
                    Parent(){ cout<<"Constructor in parent class"<<endl;}
                    ~Parent(){ cout<<"Destructor in parent class"<<endl;}
                    void infoParent(){ cout<<"this parent class"<<endl;}
};
class Child:public Parent{
          public:
                    Child(){ cout<<"Constructor in child class"<<endl;}
                    ~Child(){ cout<<"Destructor in child class"<<endl;}
                    void infoChild(){cout<<"this child class"<<endl;}
};
int main(){
          Child anak;
          anak.infoParent();
          anak.infoChild();
          return 0;
}
```

Latihan 1


Person
Nama
Alamat
Umur
getNama()
getAlamat()
getUmur()
Dosen
IdDosen
getIdDosen()
Mahasiswa
NIM
GetNimMahasiswa()

Latihan 2

- Buatlah class diagram mengenai studi kasus yang dapat diselesaikan dengan menggunakan konsep inheritance
- Buatlah program berdasarkan class diagram tersebut