IPB University
Bogor Indonesia

KOM120C -- BAHASA PEMROGRAMAN

Inheritance (continue)

- Type of Inheritance in C++
- Static Keyword
- Virtual Function
- Abstract Class

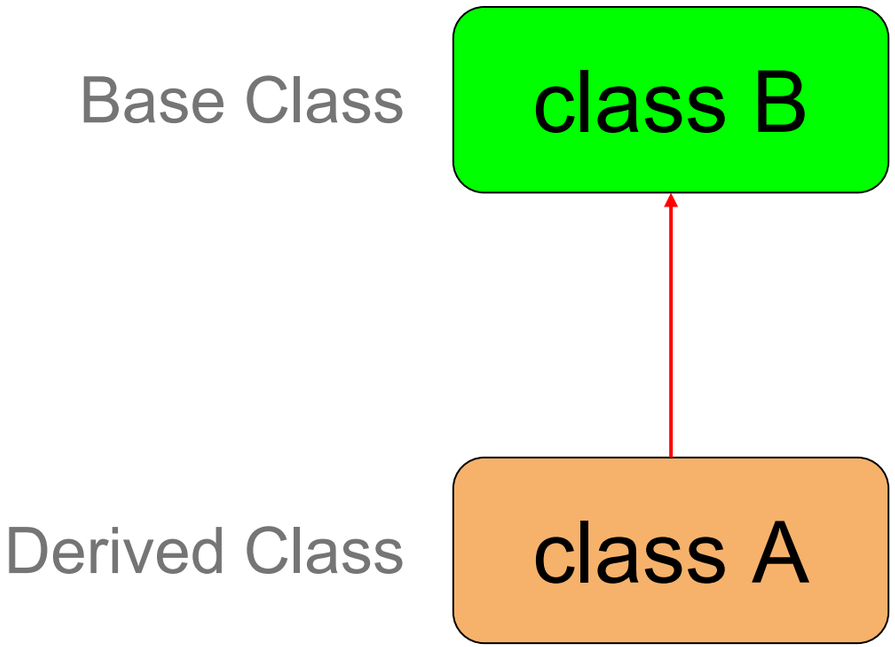
Tim Pengajar Bahasa Pemrograman IPB University

Jenis Pewarisan dalam C++

- Single Inheritance
- Multiple Inheritance
- Multilevel Inheritance
- Hierarchical Inheritance

Single Inheritance

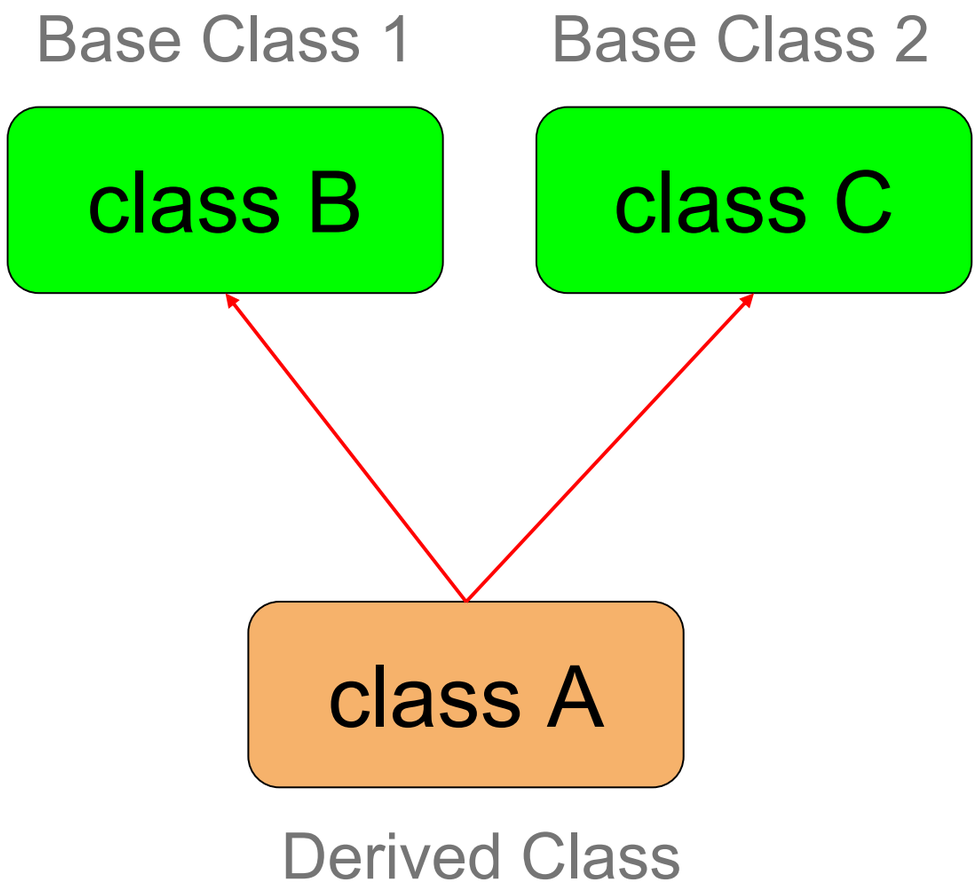
Base class hanya memiliki sebuah derived class




```
#include<iostream>
using namespace std;

class Kendaraan {
public:
    Kendaraan() { cout << "Ini kendaraan\n"; }
};

class Mobil : public Kendaraan {
    Mobil() { cout << "Ini mobil\n"; }
};

int main()
{
    Mobil obj;
    return 0;
}
```

Multiple Inheritance

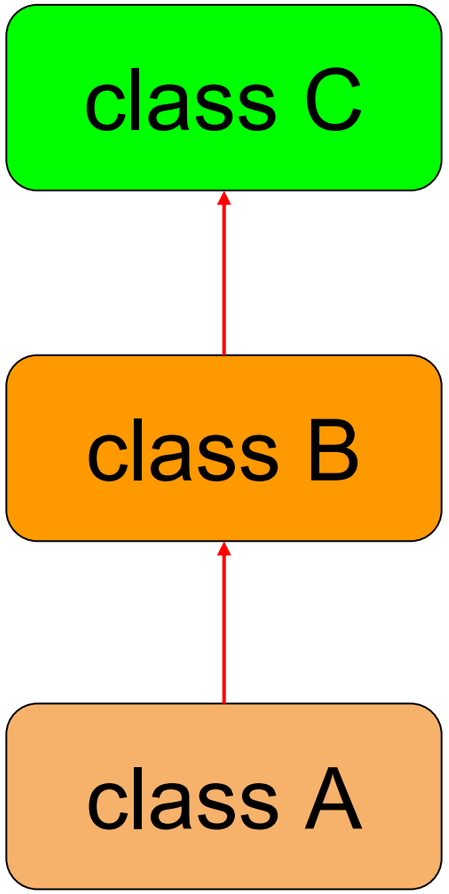
Sebuah derived class diturunkan lebih dari satu base class




```
class Kendaraan {
public:
    Kendaraan() { cout << "Ini kendaraan\n"; }
};

class Roda4 {
public:
    Roda4() { cout << "Ini kendaraan roda 4\n"; }
};

class Mobil : public Kendaraan, public Roda4 {
    Mobil() { cout << "Ini mobil\n"; }
};

int main()
{
    Mobil obj;
    return 0;
}
```

Multilevel Inheritance

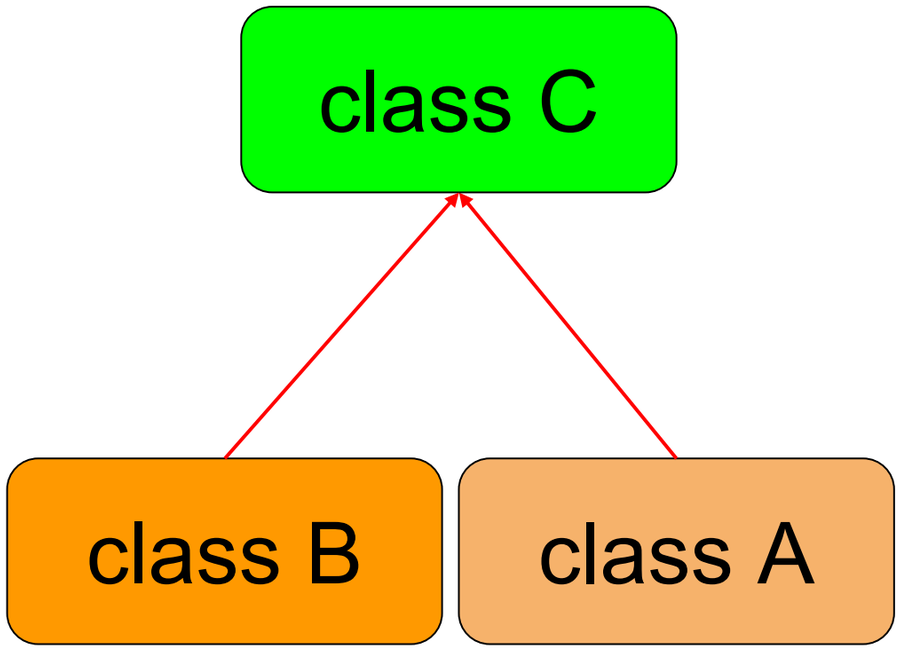
Sebuah derived class menjadi base class




```
class Kendaraan {
public:
    Kendaraan() { cout << "Ini kendaraan\n"; }
};

class Roda4 : public Kendaraan {
public:
    Roda4() { cout << "Ini kendaraan roda 4\n"; }
};

class Mobil : public Roda4 {
    Mobil() { cout << "Ini mobil\n"; }
};

int main()
{
    Mobil obj;
    return 0;
}
```

Hierarchical Inheritance

Lebih dari satu derived class diturunkan dari sebuah base class




```
class Kendaraan {
public:
    Kendaraan() { cout << "Ini kendaraan\n"; }
};

class Sedan : public Kendaraan {
    Mobil() { cout << "Ini sedan\n"; }
};

class Bus : public Kendaraan {
    Bus() { cout << "Ini Bus\n"; }
};

int main()
{
    Sedan obj1;
    Bus obj2;
    return 0;
}
```

Static Keyword in C++

- Static variables
- Static members of class

Static Variables in a Function

- Ketika variabel dideklarasikan sebagai static, maka alokasi variabel tersebut berlaku selama program berjalan.
- Artinya, alokasi untuk variabel static di dalam fungsi akan dilakukan hanya sekali walaupun fungsi tersebut dipanggil berkali-kali.

```
#include <iostream>
using namespace std;

void demo()
{
    static int c = 4;
    cout << c << " ";
    c++;
}

int main()
{
    for (int i=0; i<5; i++)
        demo();
    return 0;
}
```

Static Variables in a Class

- Ketika variabel dideklarasikan sebagai static dalam class, maka alokasi variabel tersebut akan di-share oleh objek.
- Artinya, variabel static tidak dapat disalin untuk objek yang berbeda.
- Variabel static tidak dapat diinisialisasi menggunakan constructor.

```
#include<iostream>
using namespace std;

class myClass
{
public:
        static int i;

        myClass() { };
};

int main()
{
    myClass obj1;
    myClass obj2;
    obj1.i = 2;
    obj2.i = 3;

    cout << obj1.i<<" "<<obj2.i;
    return 0;
}
```

```
class myClass
{
public:
        static int i;
        myClass() {}
};

int myClass::i = 4;

int main()
{
        myClass obj;
        cout << obj.i;
    return 0;
}
```

Class Objects as Static

Seperti variabel, objek yang dideklarasikan static berlaku selama program berjalan

```
// Tanpa static
#include<iostream>

class myClass
{
   int i;
   public:
      myClass() {
         i = 0;
         std::cout << "A\n";
      }
      ~myClass() { cout << "B\n"; }
};

int main()
{
   int x = 0;
   if (x==0) myClass obj;
   std:: cout << "End of main\n";
   return 0;
}
```

```
// Dengan static
#include<iostream>

class myClass
{
    int i = 0;
    public:
    myClass() {
        i = 0;
        std::cout << "A\n";
    }
    ~myClass() { cout << "B\n"; }
};

int main()
{
    int x = 0;
    if (x==0) { static myClass obj; }
    std::cout << "End of main\n";
    return 0;
}
```

Static Functions in a Class

- Sama seperti anggota data static atau variabel static di dalam kelas, fungsi anggota static juga tidak bergantung pada objek kelas.
- Diizinkan untuk memanggil fungsi anggota statis menggunakan objek dan operator '.', tetapi disarankan menggunakan nama kelas dan operator '::'.

```
#include<iostream>
using namespace std;

class myClass
{
    public:
        static void print()
        {
            cout<<"Welcome!";
        }
};

int main()
{
    myClass::print();
    return 0;
}
```

Object of Self Type

Class dapat mengandung object dengan tipe dirinya sendiri, namun harus static atau pointer.

```
#include<iostream>
using namespace std;

class Test {
    static Test self;

};

int main()
{
    Test t;
    return 0;
}
```

```
#include<iostream>
using namespace std;

class Test {
    Test *self;

};

int main()
{
    Test t;
    return 0;
}
```

Virtual Function

Virtual Functions

- Fungsi virtual adalah fungsi anggota yang dideklarasikan dalam base class dan didefinisikan ulang (ditimpa) oleh kelas turunannya.
- Fungsi virtual tidak dapat static.
- Fungsi virtual harus diakses menggunakan pointer.
- Prototipe fungsi virtual harus sama, di dalam base class dan derived class.

```
class A {
public:
   virtual void print() { cout << "Print A\n"; }
   void show() { cout << "Show A\n"; }
};

class B : public A {
public:
   void print() { cout << "Print B\n"; }
   void show() { cout << "Show B\n";    }
};

int main()
{
   A *ptr;
   B d;
   ptr = &d;
   ptr->print();
   ptr->show();
   return 0;
}
```

Abstract Function
Abstract Class

Abstract Function & Abstract Class

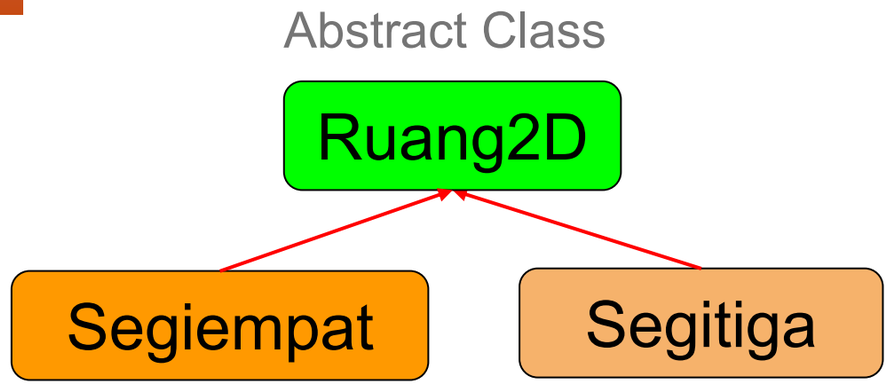
Dalam C++, abstract function disebut juga sebagai pure virtual function, yaitu virtual function yang harus di-override di dalam derived class.

Abstract function dapat dideklarasikan dengan memberikan assignment nilai 0.

```
class Ruang2D
{
public:
   virtual double luas() = 0;

};
```

Class yang sedikitnya mengandung sebuah abstract function disebut sebagai abstract class.

Latihan



Diketahui 3 class, yaitu Ruang2D (untuk mengolah data ruang 2 dimensi yang memiliki maksimum 4 sisi), Segiempat (untuk mengolah data segi empat), dan Segitiga (untuk mengolah data segitiga siku-siku). Class Ruang2D merupakan abstract class yang memiliki atribut 4 sisi bilangan bulat (integer), dan fungsi abstrak luas() dan keliling(), serta sebuah fungsi mutator set(). Lengkapi setiap class dan buat program OOP untuk membaca m sisi (panjang dan lebar) segiempat, dan n sisi (alas dan tinggi) segitiga siku-siku. Program menampilkan luas dan keliling segiempat diikuti luas dan keliling segitiga, terurut dimulai dari luas terbesar. Jika luas sama, data diurutkan berdasarkan keliling descending.

```
class Ruang2D
{
   protected:
      double sisi1, sisi2, sisi3, sisi4;
public:
      virtual double luas() = 0;
      virtual int keliling() = 0;
      void set(int s1=0,int s2=0,
               int s3=0,int s4=0);
};
```

Contoh Input:

```
2 3     // 2 segiempat, 3 segitiga
2 3
4 5
1 2
3 4
2 3
```

Contoh Input:

```
SEGIEMPAT
20.0 18.0
6.0 10.0
SEGITIGA
6.0 13.0
3.0 8.6
1.0 5.2
```