PEMROGRAMAN LANJUT

12 | Polymorphism

Team Teaching Pemrograman Lanjut

Fakultas Ilmu Komputer UB

Review

- int A::b(int c) { }
  - Implementasi method di luar class
- a->b
  - Mengakses data member atau method menggunakan pointer
- class A: public B { };
  - Membuat derived class A dari base class B

Pointer Base Class

- Pada inheritance, pointer dari derived class dapat mengakses base class
- Polimorfisme mengambil keuntungan dari kasus di atas
  - Sederhana namun kuat dan serbaguna

Pointer Base Class


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

class Polygon {
  protected:
    int width, height;
  public:
    void set_values (int a, int b)
      { width=a; height=b; }
};

class Rectangle: public Polygon {
  public:
    int area()
      { return width*height; }
};

class Triangle: public Polygon {
  public:
    int area()
      { return width*height/2; }
};
```

```
int main () {
  Rectangle rect;
  Triangle trgl;
  Polygon * ppoly1 = &rect;
  Polygon * ppoly2 = &trgl;
  ppoly1->set_values (4,5);
  ppoly2->set_values (4,5);
  cout << rect.area() << '\n';
  cout << trgl.area() << '\n';
  return 0;
}
```

```
rect.set_values(4, 5);
```

Pointer Base Class

- `ppoly1` dan `ppoly2` merupakan pointer dari `Polygon`
  - Bukan `Rectangle` maupun `Triangle`
  - Hanya anggota yang diturunkan dari `Polygon` yang dapat diakses, bukan anggota class turunannya (`Rectangle` & `Triangle`)

- Oleh karena itu, pada contoh sebelumnya untuk mengakses method `area()` menggunakan object `rect` & `trgl` secara langsung
  - Pointer base class tidak dapat mengakses method `area()`

Dasar Polimorfisme

- Method `area()` dapat diakses dari pointer `Polygon` apabila `area()` merupakan anggota dari `Polygon`

- TETAPI
  - `Rectangle` dan `Triangle` memiliki implementasi `area()` yang berbeda
  - Oleh karena itu implementasi `area()` tidak dapat dilakukan di base class

Virtual Member

- Merupakan member function yang dapat didefinisikan ulang di derived class dengan menjaga referensi fungsinya
- Menggunakan keyword `virtual`

Virtual Member


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

class Polygon {
  protected:
    int width, height;
  public:
    void set_values (int a, int b)
      { width=a; height=b; }
    virtual int area () { return 0; }
};

class Rectangle: public Polygon {
  public:
    int area () { return width * height; }
};

class Triangle: public Polygon {
  public:
    int area () { return (width * height / 2); }
};
```

```
int main () {
  Rectangle rect;
  Triangle trgl;
  Polygon poly;
  Polygon * ppoly1 = &rect;
  Polygon * ppoly2 = &trgl;
  Polygon * ppoly3 = &poly;
  ppoly1->set_values (4,5);
  ppoly2->set_values (4,5);
  ppoly3->set_values (4,5);
  cout << ppoly1->area() << '\n';
  cout << ppoly2->area() << '\n';
  cout << ppoly3->area() << '\n';
  return 0;
}
```

Virtual Member

- Method area() dideklarasikan sebagai virtual di base class, sehingga dapat dideklarasikan ulang di derived class

- PENTING!
  - Apabila keyword virtual dihapus, maka hasil seluruh area() adalah 0, karena pointer base class pasti akan mengakses method area() yang terletak di base class (early binding)
  - Oleh karena itu, keyword virtual mengijinkan derived class memiliki method yang sama dan dapat diakses melalui pointer base class

Polimorfisme

- Suatu class yang mendeklarasikan fungsi virtual disebut sebagai polymorfisme class
- Polimorfisme
  - Suatu konsep yang mana terdapat beberapa class yang memiliki bentuk method yang sama dan cara pemanggilan methodnya dengan cara yang sama