TIF1202 – Pemrograman Berorientasi Objek

HO 01 - Mengenal Kelas dan Objek

Opim Salim Sitompul

Department of Information Technology
Universitas Sumatera Utara

Outline

Pendahuluan

- C++ mungkin akan tetap menjadi 5 bahasa pemrograman terpopuler untuk tiga dekade yang akan datang - Richard Kenneth Eng (https://www.quora.com/)
- C++ sangat sering digunakan karena:
  - Menghasilkan kode program paling efisien dalam dunia bisnis.
  - Memiliki ekosistem *library* dan *tools* yang sangat kaya.
  - Bahasa pemrograman kelas-enterprise yang *solid*, dengan reputasinya yang kuat untuk *reliability* dan *robustness*; telah terbukti untuk rekayasa perangkat lunak skala besar selama lebih dari tiga dekade.

Pendahuluan

- Pemrograman berorientasi objek (*Object-Oriented Programming / OOP)* berdasarkan pada objek-objek yang membentuk suatu sistem.
- Di dalam sistem itu, objek-objek tersebut menyimpan jenis data tertentu dan memiliki operasi-operasi yang dapat dilakukan terhadap data tersebut.
- Objek menyajikan sebuah entitas dunia nyata (benda) yang dapat menyimpan data (disebut anggota variabel), dan memiliki sekumpulan operasi khusus yang dapat dikenakan pada objek itu (disebut anggota fungsi).

Sasaran OOP

- Sasaran (*goal*) yang ingin dicapai dalam pemrograman berorientasi objek adalah:
  - *Ease of design and code reuse*
    Setelah kode program teruji berjalan dengan benar, objek akan lebih mudah didesain ulang dan menggunakan kembali kode program yang telah dibuat untuk aplikasi berikutnya.
  - *Increased reliability*
    Setelah dilakukan pengujian terhadap pustaka-pustaka objek yang telah dibuat, penggunaan kembali kode-kode program yang sudah ada akan meningkatkan keandalan program.

Sasaran OOP

- *Ease of understanding*
  Desainer membangun sistem berdasarkan potongan-potongan kecil berbentuk objek. Hal ini memberikan kerangka bagi desainer untuk mengidentifikasi objek-objek yang diperlukan. Pemrogram akan terbantu untuk fokus dalam memahami komponen-komponen kunci yang membentuk sistem.
- *Increased abstraction*
  “Look at the big picture” - Mengabaikan sementara rincian dasar sehingga dapat bekerja pada elemen-elemen sistem yang lebih mudah dipahami.

Sasaran OOP

- *Increased encapsulation*
  Encapsulation adalah sebuah konsep yang mengikat semua data dan fungsi yang mengolah data itu ke dalam sebuah paket yang disebut kelas, dan menjaga keduanya agar aman dari interferensi dan penyalahgunaan.
- *Increased information hiding*
  Information hiding adalah teknik untuk mengurangi kompleksitas program. Mekanisme yang digunakan adalah encapsulation. Program memperlakukan variabel dan fungsi anggota di dalam sebuah kelas sebagai “black box” untuk melaksanakan operasi tertentu tanpa harus khawatir pada rinciannya yang tersembunyi.

Memahami Kelas dan Objek

- Kelas memberikan sebuah template bagi program untuk membuat objek.
  - Misal: Kelas file menjabarkan anggota variabel dan anggota fungsi yang akan digunakan untuk objek file yang akan dibuat dalam program.
- Objek adalah sebuah contoh (instance) dari sebuah template kelas.
  - Objek adalah sebuah entiti, contoh spesifik, atau sesuatu yang dikerjakan oleh program.

Memahami Kelas dan Objek

- Tiap-tiap objek yang diturunkan dari sebuah kelas akan memiliki anggota variabel dan anggota fungsi yang dimiliki oleh kelas tersebut.
- Menurunkan objek dari sebuah kelas adalah sama seperti mendeklarasikan variabel dari satu jenis data tertentu.

Memahami Kelas dan Objek

- Perhatikan definisi kelas-kelas berikut:
  - Sebuah kelas File Windows memiliki beberapa anggota variabel seperti name, date modified, type, dan size. Operasi-operasi yang dapat dilakukan pada file tersebut dinyatakan oleh anggota-anggota fungsi seperti copy, delete, rename, move, dll.
  - Sebuah kelas Buku memiliki anggota variabel judul, pengarang, penerbit, kota terbit, isbn, jenis kulit, dan jumlah halaman. Adapun beberapa operasi yang dapat dilakukan pada buku antara lain: membaca, menambah bookmark, menamatkan buku, membuat daftar buku.

Mengenal Kelas I/O Stream

- Pustaka iostream adalah pustaka berorientasi objek yang memberikan fungsionalitas input dan output menggunakan stream.
- Stream adalah abstraksi yang menyatakan alat (device) untuk melaksanakan operasi input dan ouput.
- Secara mendasar stream dapat disajikan sebagai sebuah sumber atau tujuan fisik dari tidak berhingga banyaknya karakter, misalnya file disk, keyboard atau layar monitor.
  - Banyaknya jumlah karakter untuk input atau output tidak perlu diketahui terlebih dahulu.

Mengenal Kelas I/O Stream

- Kelas **iostream** yang terdapat pada berkas judul <iostream> mendeklarasikan objek-objek yang digunakan untuk berkomunikasi melalui peralatan input dan output standard, yaitu **cin**, **cout** dan **cerr**.
- Kelas **iostream** terbagi atas dua kelas, yaitu **istream** dan **ostream**.

Mengenal Kelas I/O Stream

- Objek dari kelas ostream adalah **cout** dan **cerr**.
- Sebagai sebuah objek dari kelas ostream, karakter-karakter dapat dituliskan padanya sebagai data berformat menggunakan *insertion operator* (operator $<<$).
- Contoh:
  cout $<<$ "Hello, world!" $<<$ endl;
  cerr $<<$ "Error membuka file" $<<$ endl;

Mengenal Kelas I/O Stream

- Objek dari kelas istream adalah **cin**.
- Sebagai objek dari kelas istream, karakter-karakter dapat diambil sebagai data berformat menggunakan *extraction operator* (operator $>>$).
- Contoh:
  cin $>>$ usia;

Mendeklarasikan Kelas

- Sebuah kelas dideklarasikan menggunakan kata kunci **class**.
- Contoh:

```
class HelloWorld
{
    public:
        void hello()
        {
            cout << "Hello, world!" << endl;
        }
};
```

Mendeklarasikan Kelas

- Pada deklarasi class HelloWorld ini tidak terdapat anggota variabel.
- Anggota fungsi yang dimiliki adalah void hello().
- Definisi fungsi dapat dituliskan di dalam ataupun di luar kelas.
  - Tetapi sebaiknya penulisan definisi anggota fungsi di dalam kelas hanya untuk fungsi-fungsi yang singkat, yaitu fungsi-fungsi yang hanya terdiri satu atau dua baris.

Mendeklarasikan Kelas

- Apabila definisi anggota fungsi ditulis di luar kelas, harus dimulai dengan nama kelas dan menggunakan sebuah operator :: yang disebut *scope resolution operator*.
- Tanda :: pada definisi anggota fungsi bertujuan untuk menunjukkan bahwa fungsi itu adalah anggota dari kelas yang disebutkan.

Contoh Program

```
1  // Contoh1_1.cpp
2  #include <iostream>
3
4   class HelloWorld
5   {
6    public:
7      void hello()
8      {
9        std::cout << "Hello, world!" << std::
             endl;
10     }
11  };
```

Mendeklarasikan Kelas

```
1  int main()
2  {
3    HelloWorld myWorld;
4
5    myWorld.hello();
6
7    return 0;
8  }
```

Menurunkan Objek

- Menurunkan objek dari sebuah kelas berarti membuat contoh dari kelas itu.
- Objek yang diturunkan akan memiliki anggota-anggota yang terdapat di dalam kelas tersebut.
- Objek yang diturunkan dari sebuah kelas dituliskan setelah nama kelas.
- Contoh:
  HelloWorld myWorld;
  Ascii myTable;

Namespace

- **namespace** digunakan untuk mendefinisikan konteks dari nama anggota kelas yang digunakan dalam sebuah program.
- Pada dasarnya sebuah **namespace** mendefinisikan sebuah ruang lingkup (scope).
- Sebagai contoh, semua anggota fungsi pustaka standar termuat dalam **namespace std**, sehingga setiap fungsi yang ada didalamnya dapat digunakan dengan mendeklarasikannya seperti berikut:

  using namespace std;

Contoh Program

```
1  //Contoh1_2.cpp
2
3  #include <iostream>
4
5  using namespace std;
6
7  class Ascii
8  {
9    public:
10     void displayAscii();  // prototipe fungsi
11 };
```

Contoh Program

```
1  void Ascii::displayAscii()
2  {
3    cout << "TABEL␣ASCII" << endl;
4    for(int i=0; i<256; i++)
5    {
6      cout << i << "␣=␣" << char(i) << endl;
7      if((i>0) && (i%20 == 0))
8        getchar();
9    }
10   cout << "Press␣<ENTER>␣..." << endl;
11   getchar();
12 }
```

Contoh Program

```
1  int main()
2  {
3    Ascii myAscii;
4
5    myAscii.displayAscii();
6
7    return 0;
8  }
```

Mengakses Anggota Kelas

- Anggota-anggota kelas dapat diakses melalui objek-objek yang diturunkan dari kelas itu sendiri.
- Ada dua operator yang digunakan untuk mengakses anggota suatu kelas, tergantung dari jenis data anggota kelas tersebut.

  class.member atau

  class_ptr->member (indireksi)

Mengakses Anggota Kelas

- Untuk anggota kelas berjenis variabel biasa, digunakan operator titik (dot), sedangkan untuk anggota variabel pointer digunakan operator panah (kombinasi ->, ini akan dibahas belakangan).
- Contoh:

```
myWorld.hello();
myWorld->hello();
```

Latihan Pemrograman

- Buatlah sebuah program C++ yang memuat sebuah kelas dengan satu anggota variabel berjenis karakter. Kelas tersebut memiliki dua anggota fungsi berikut: sebuah fungsi untuk meminta input sebuah karakter dan sebuah fungsi untuk menampilkan kode ASCII karakter tersebut.
- Buatlah sebuah program C++ yang memiliki sebuah kelas dengan satu anggota variabel berjenis integer untuk menyimpan jari-jari lingkaran, dua anggota variabel berjenis float untuk menyimpan keliling dan luas lingkaran. Lengkapilah kelas tersebut dengan anggota-anggota fungsi yang diperlukan untuk menghitung keliling lingkaran dan luas lingkaran dan menampilkan hasilnya.