Pertemuan 9

PERANCANGAN BERORIENTASI OBJEK

1. PENDAHULUAN

- Sistem berorientasi objek terdiri dari **objek** yang berinteraksi yang mempertahankan keadaan (*state*) lokal dan menyediakan operasi pada *state* tersebut.
- Representasi *state* bersifat pribadi dan tidak dapat diakses langsung dari luar objek.
- Proses desain berorientasi objek melibatkan perancangan kelas objek dan hubungan antara kelas-kelas tersebut.

Pendahuluan (Lanjutan)

- Mengubah implementasi suatu objek atau menambahkan metode atau operasi tidak mempengaruhi objek lain dalam sistem.
- Hal-hal yang diperhatikan dalam desain berorientasi objek:
  1. Memahami dan mendefinisikan **konteks** dan **interaksi** eksternal dengan sistem.
  2. Desain arsitektur sistem.
  3. Identifikasi **objek utama** dalam sistem.
  4. Kembangkan model desain.
  5. Tentukan antarmuka.

Pendahuluan (Lanjutan)

- Desain berorientasi objek biasanya diimplementasikan dengan bahasa pemrograman berorientasi objek.
- Keuntungan utama dari desain OO adalah:
  a. Sistem Analis dapat **menghemat waktu** dan menghindari kesalahan dengan menggunakan objek secara modular
  b. Programer dapat menerjemahkan desain ke dalam kode. Objek yang baru dapat dibuat tanpa mengubah kode yang sudah ada
  c. Bekerja dengan modul program yang dapat digunakan kembali (*reuse)* yang telah diuji dan diverifikasi.

A. Identifikasi Kelas Objek

Tujuan mengidentifikasi kelas objek dalam sistem berorientasi objek:

1. Gunakan analisis gramatikal dari deskripsi bahasa alami. Objek dan atribut adalah kata benda; operasi atau layanan adalah kata kerja.
2. Gunakan entitas nyata (benda) dalam domain aplikasi seperti mobil, peran seperti manajer atau dokter, acara seperti permintaan, interaksi seperti pertemuan, lokasi seperti kantor, unit organisasi seperti perusahaan.
3. Gunakan analisis berbasis skenario dimana berbagai skenario penggunaan sistem diidentifikasi dan dianalisis secara bergantian.

B. Istilah dalam Objek Oriented

1. OBJEK (*Object*)

- Objek adalah konsep atau abstraksi tentang sesuatu yang memiliki arti untuk aplikasi yang akan dikembangkan
- Objek diwakili dengan kata benda
- Objek dapat berupa:
  - Objek orang/manusia: Karyawan, Mahasiswa
  - Objek tempat: Kantor, Gedung, Toko
  - Objek abstrak: Transaksi, Jadwal, Peminjaman
  - Objek organisasi: Divisi-IT, HRD
  - Objek peralatan/benda: Mobil, Buku, Baju

2. ATRIBUT (*Attribute*)

- Suatu objek memiliki atribut tertentu yang merupakan **karakteristik** yang menggambarkan objek.
- Suatu atribut dapat mengambil sebuah nilai yang ditentukan berdasarkan *domain* yang dihitung.
- *Domain* merupakan satu himpunan nilai-nilai spesifik.
- Contoh: kelas MOBIL memiliki sebuah atribut WARNA. Domain nilai untuk warna adalah {putih, hitam, perak, abu-abu, biru, merah, kuning, hijau}.
- Objek dapat memiliki atribut khusus yang disebut *state*. Keadaan suatu objek adalah kata sifat yang menggambarkan status objek saat ini.
- Misalnya rekening bank dapat aktif, tidak aktif, tertutup, atau dibekukan

3. METODE (*Method*)

- Suatu metode mendefinisikan **tugas-tugas** spesifik yang dapat dilakukan oleh suatu objek
- Metode dituliskan dengan kata kerja yang menggambarkan apa dan bagaimana suatu objek melakukan sesuatu.
- Misalnya: objek PELANGGAN dapat melakukan tugas-tugas tertentu seperti melakukan pemesanan, membayar tagihan, dan mengubah alamatnya.

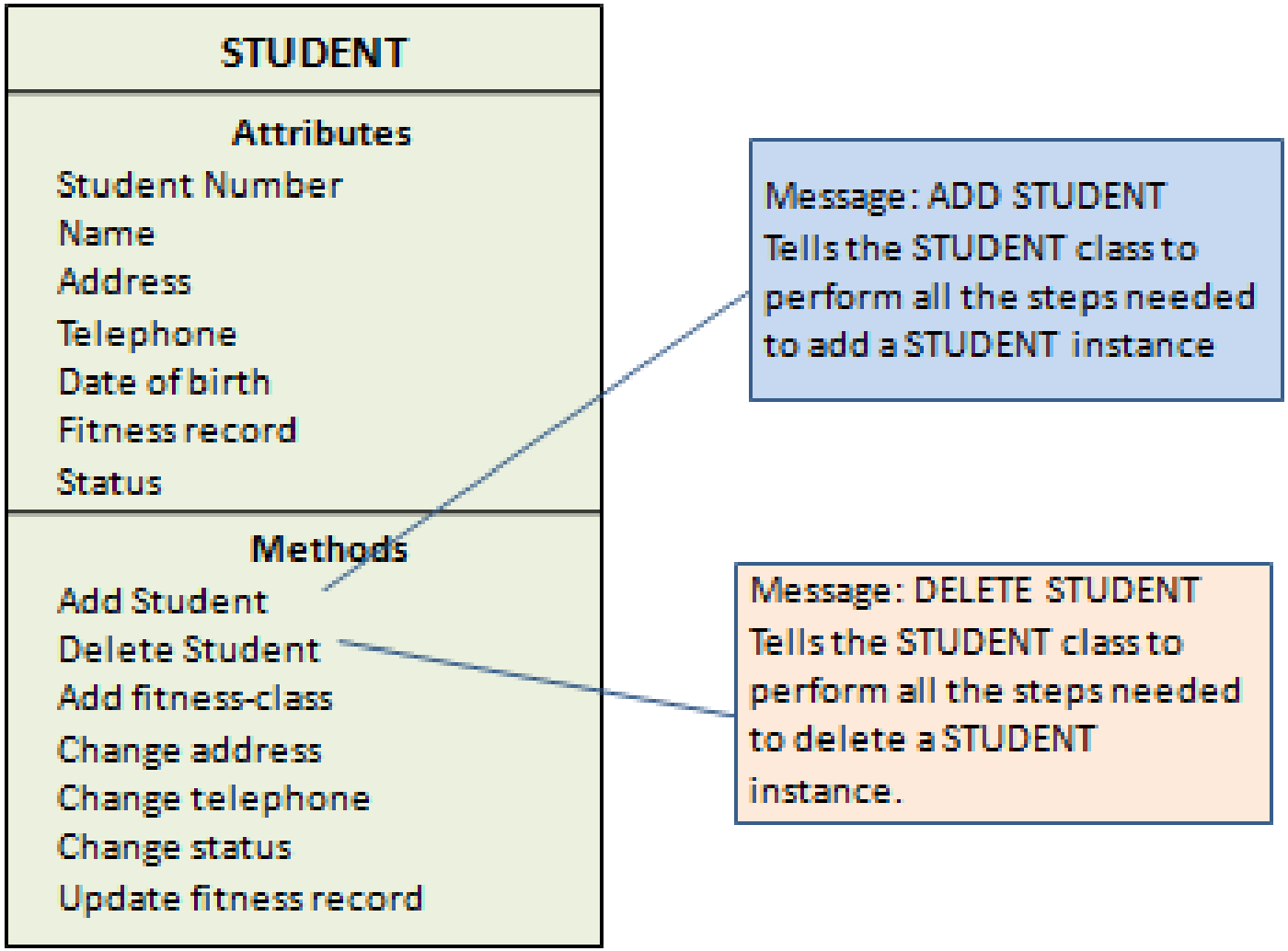
4. PESAN (*Message*)

- Pesan (*Message)* adalah perintah yang memberi tahu suatu objek untuk melakukan metode tertentu.
- Misalnya: pesan TAMBAHKAN SISWA mengarahkan kelas SISWA untuk menambahkan nomor siswa, nama, dan data lain tentang siswa itu. Demikian pula, pesan bernama HAPUS SISWA memberi tahu kelas SISWA untuk menghapus instance Siswa.
- Pesan yang sama untuk dua objek berbeda dapat menghasilkan hasil yang berbeda.

Contoh pesan

Objek INSTRUCTOR mengirim pesan ENTER GRADE ke instance kelas STUDENT RECORD.
Objek INSTRUCTOR dan kelas STUDENT RECORD dapat digunakan kembali dengan sedikit modifikasi, di sistem informasi sekolah lain dimana banyak atribut dan metode akan serupa

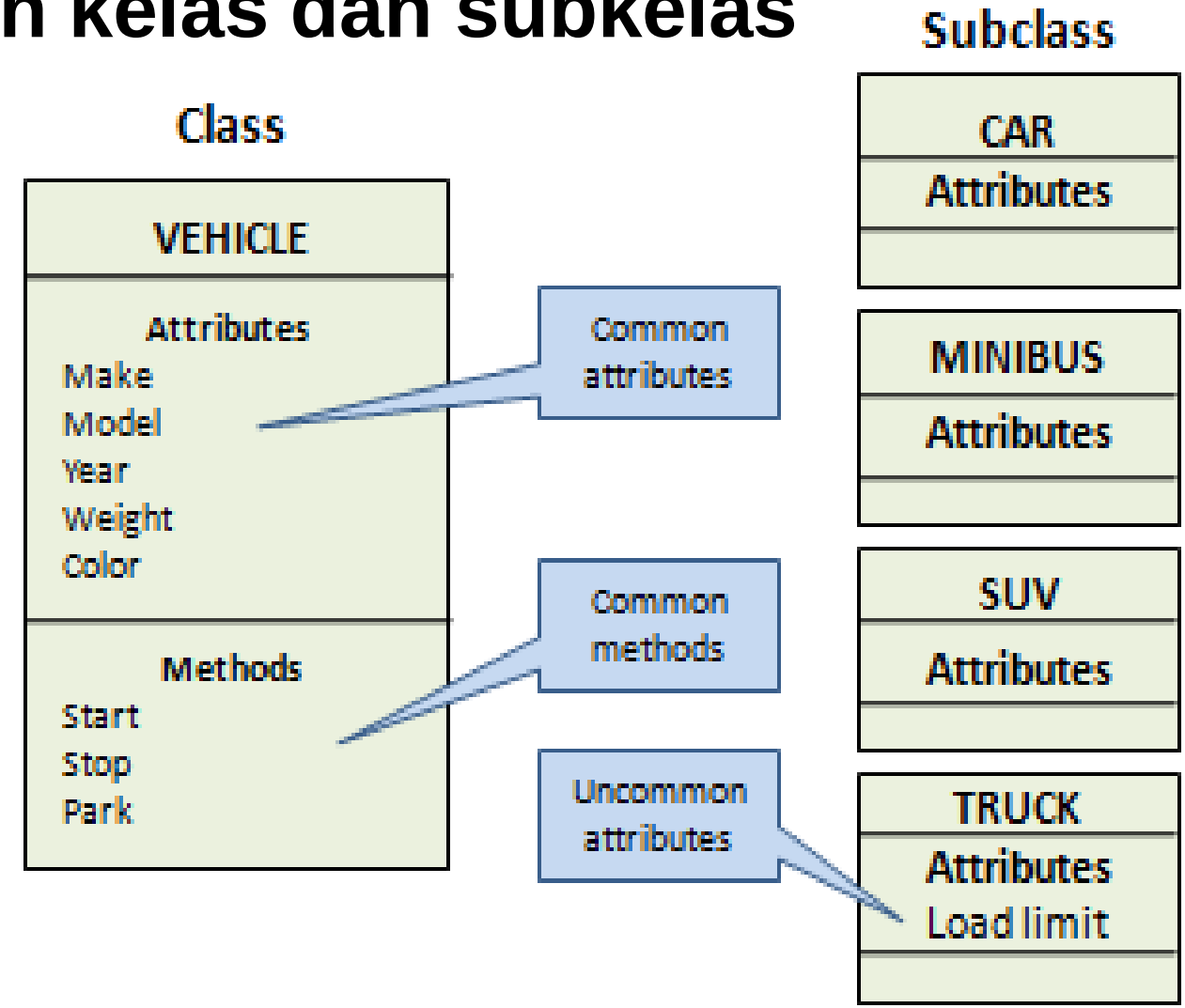
Contoh Pesan:



5. KELAS (*Class*)

- Kelas adalah deskripsi umum yang menggambarkan sebuah kumpulan berisi objek-objek yang sama.
- Semua objek dalam kelas berbagi atribut dan metode yang sama, sehingga kelas seperti *blue print*, atau *template* untuk semua objek di dalam kelas.
- *Superclass* adalah generalisasi dari satu himpunan kelas-kelas yang berhubungan.
- *Subclass* adalah spesialisasi dari *superclass*.
- Contoh: superclass kendaraanBermotor adalah generalisasi dari kelas Truk, SUV, Minibus dan Car. Subclass Minibus mewarisi semua atribut kendaraanBermotor, tetapi juga menggabungkan atribut tambahan yang spesifik hanya untuk Minibus.

Contoh kelas dan subkelas



C. Hubungan Antara Objek dan Kelas

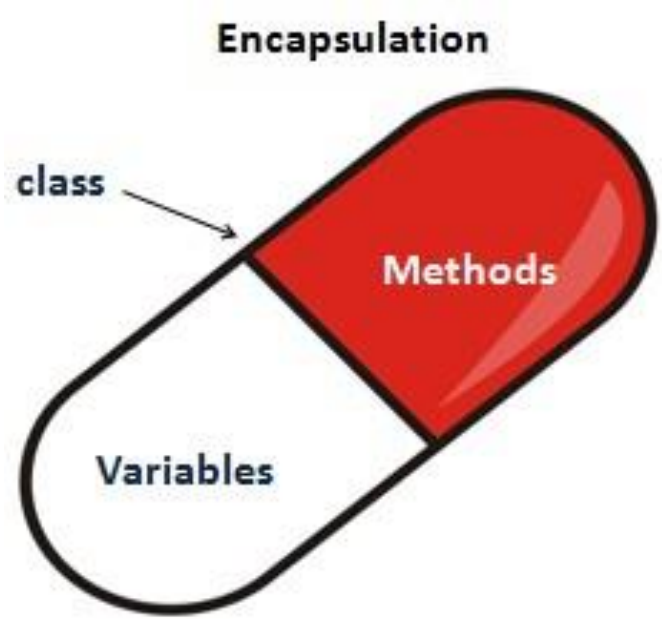
- Hubungan (*relationship*) memungkinkan objek untuk berkomunikasi dan berinteraksi ketika mereka melakukan fungsi bisnis dan transaksi yang diperlukan oleh sistem.
- Hubungan menggambarkan apa yang perlu diketahui objek satu sama lain, bagaimana objek merespon perubahan pada objek lain, dan efek keanggotaan dalam kelas, superclass, dan subclass.
- Beberapa hubungan lebih kuat daripada yang lain (seperti hubungan antara anggota keluarga lebih kuat dari satu hubungan antara kenalan biasa). Hubungan terkuat disebut **warisan**.

2. KARAKTERISTIK OBJEK

A. Enkapsulasi (*Encapsulation*)

- Data dan prosedur/fungsi dikemas bersama-sama dalam suatu objek, sehingga prosedur/fungsi lain dari luar tidak dapat mengaksesnya.
- Data terlindung dari prosedur atau objek lain kecuali prosedur yang berada dalam objek tersebut.
- Merupakan pembatasan ruang lingkup program terhadap data.
- Enkapsulasi memungkinkan objek untuk digunakan sebagai komponen modular di mana saja dalam sistem, karena objek mengirim dan menerima pesan tetapi tidak mengubah metode internal objek lain.

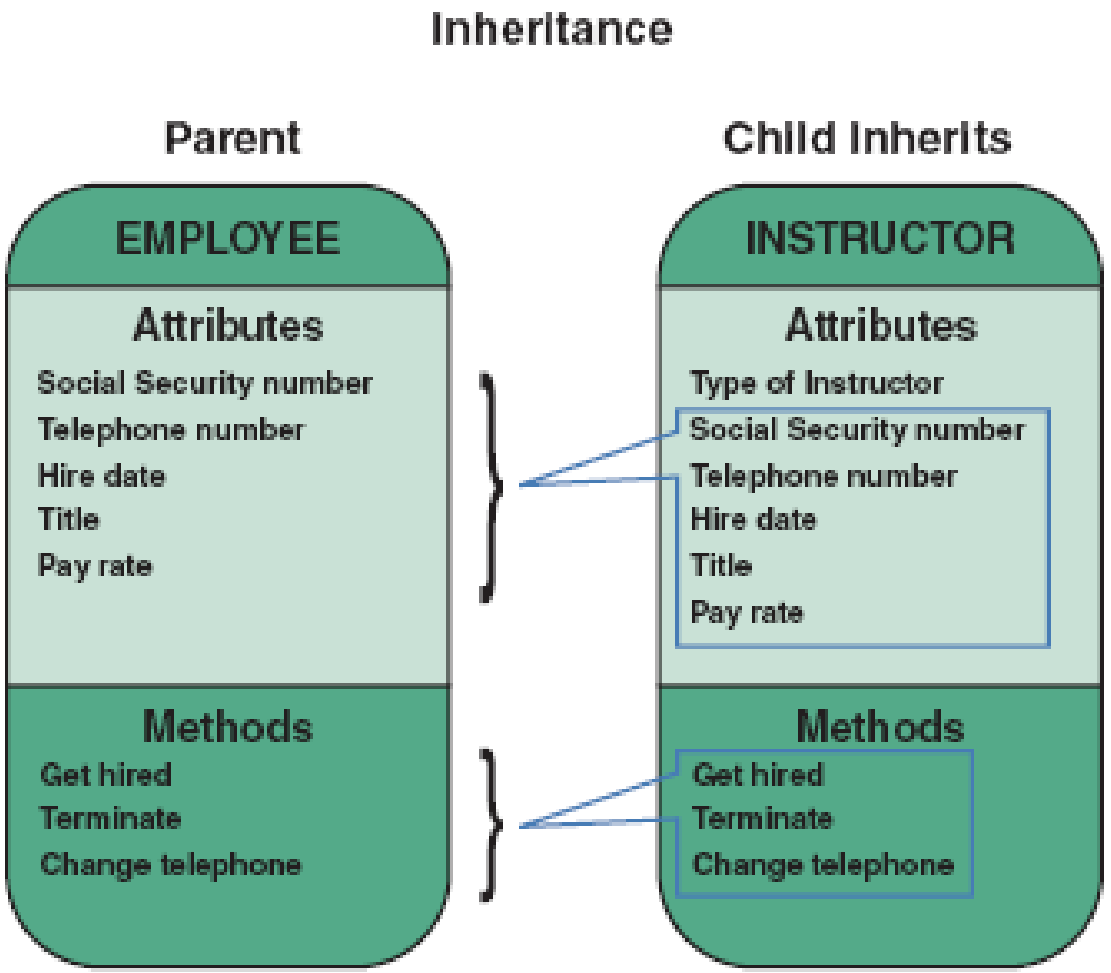
Contoh Enkapsulasi



B. Pewarisan (*Inheritance*)

- Pewarisan adalah salah satu pembeda utama antara
- sistem konvensional dan sistem berbasis objek.
- Subkelas **Y** mewarisi semua atribut dan operasi-operasi yang terkait dengan superkelas **X**. Ini berarti semua struktur dan algoritma data yang secara orisinal dirancang dan diimplementasikan untuk **X** segera tersedia untuk **Y**
- Perubahan apa pun pada atribut-atribut atau operasi-operasi yang dimuat ke dalam sebuah superkelas, akan diwarisi oleh semua subkelas.

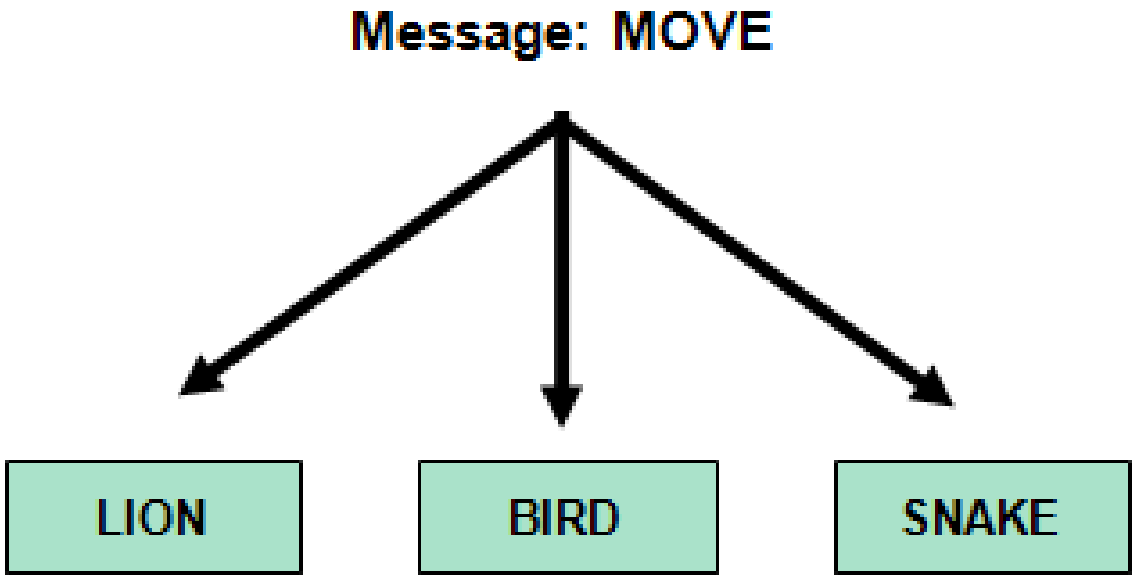
Contoh pewarisan:



C. Polimorfis (*Polymorphism*)

- Merupakan konsep yang menyatakan bahwa sesuatu yang sama dapat mempunyai bentuk dan perilaku berbeda.
- Polimorfis juga mempunyai arti bahwa operasi yang sama mungkin mempunyai perbedaan dalam kelas yang berbeda.
- Polimorfis sangat mengurangi usaha yang diperlukan untuk memperluas perancangan sistem berorientasi objek

Contoh Polimorfis



3. KELAS-KELAS PERANCANGAN

- Model kebutuhan menentukan serangkaian kelas-kelas analisis yang masing-masing kelas menggambarkan beberapa elemen masalah yang fokus pada masalah yang dilihat oleh pengguna.
- Himpunan kelas-kelas perancangan adalah

1. memperhalus kelas-kelas analisis dengan menyediakan detail perancangan yang memungkinkan kelas-kelas bisa diimplementasikan
2. menciptakan suatu himpunan kelas-kelas perancangan yang baru, yang mengimplementasikan suatu infrastruktur PL yang mendukung solusi bisnis.

Kelas perancangan yang merepresentasikan lapisan berbeda dari perancangan arsitektur:

a. **Kelas-kelas antarmuka.** Pengguna menentukan semua abstraksi yang diperlukan untuk interaksi manusia dengan komputer
b. **Kelas-kelas bisnis.** Kelas-kelas mengidentifikasi atribut dan operasi/metode yang dibutuhkan untuk mengimplementasikan beberapa elemen ranah bisnis.
c. **Kelas-kelas proses.** Mengimplementasikan abstraksi bisnis yang levelnya lebih rendah untuk sepenuhnya mengelola kelas-kelas ranah bisnis.
d. **Kelas-kelas persisten.** Merepresentasikan *data store* yang akan terus ada setelah eksekusi PL.
e. **Kelas-kelas sistem.** Mengimplementasikan manajemen PL dan mengendalikan fungsi-fungsi agar mampu mengoperasikan sistem dan berkomunikasi dengan dunia luar

A. Karakteristik Kelas Perancangan

a. Lengkap dan cukup

- Suatu kelas perancangan seharusnya menjadi enkapsulasi lengkap dari semua atribut dan metode yang dapat layak diharapkan.
- Cukup berarti memastikan bahwa kelas perancangan berisi hanya metode-metode yang cukup untuk mencapai tujuan kelas.
- Contoh: kelas Scene adalah lengkap hanya jika kelas ini berisi semua atribut dan metode yang dapat layak diasosiasikan dengan pembuatan suatu *scene video*.

Karakteristik Kelas Perancangan (Lanjutan)

b. Sederhana

- Metode-metode yang dihubungkan dengan sebuah kelas perancangan harus fokus ke pencapaian satu fungsi spesifik pada kelas.
- Contoh: kelas VideoClip memiliki atribut StartPoint dan EndPoint untuk mengindikasikan titik awal dan titik akhir.

Karakteristik Kelas Perancangan (Lanjutan)

c. Kohesi tinggi

- Kelas perancangan kohesif adalah *single minded.* Artinya kelas ini memiliki satu kumpulan kecil tanggung jawab yang fokus dan menerapkan atribut dan metode untuk menjalankan tanggung jawab tersebut.
- Contoh: kelas VideoClip dapat berisi satu kumpulan metode-metode untuk mengedit klip video. Kohesi dijaga asalkan setiap metode fokus semata-mata pada atribut-atribut yang diasosiasikan dengan klip video.

Karakteristik Kelas Perancangan (Lanjutan)

d. Keterhubungan rendah

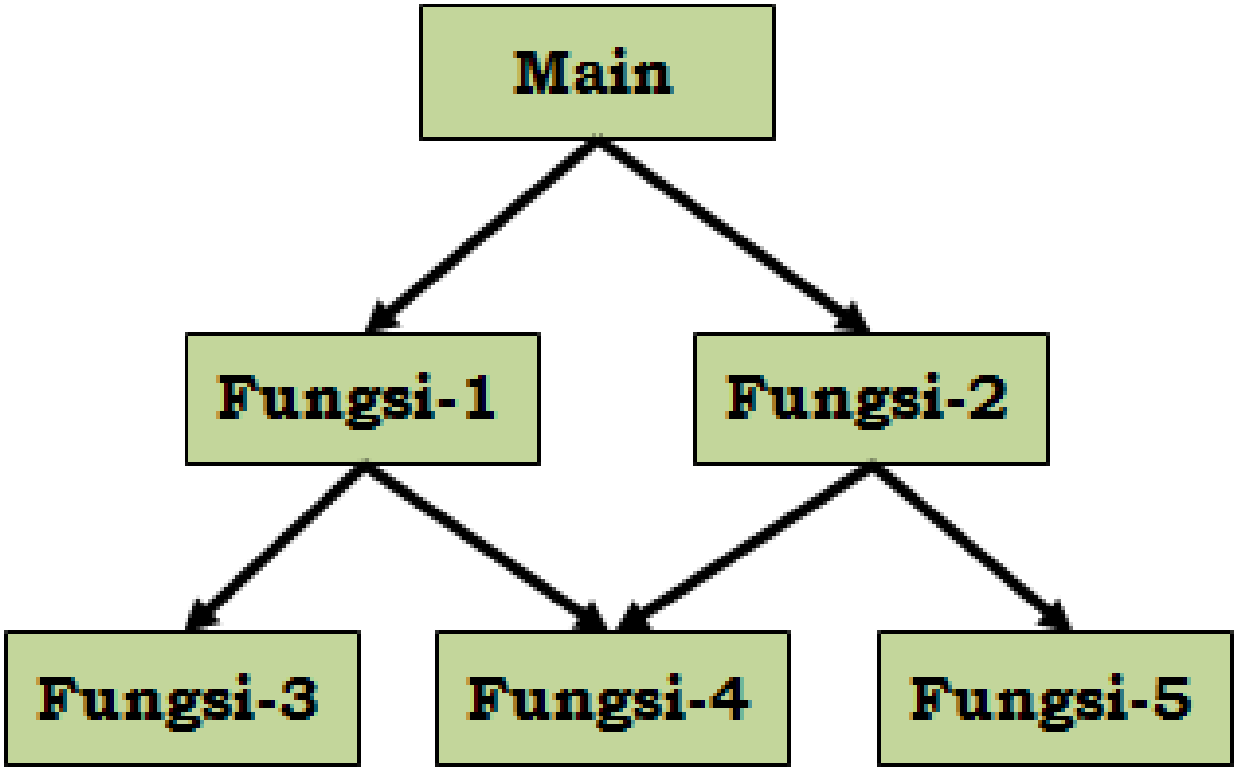
- Jika sebuah model perancangan memiliki keterhubungan tinggi (semua kelas perancangan berkolaborasi dengan semua kelas perancangan lainnya), sistem menjadi sulit diimplementasikan, diuji, dan dipelihara.
- Kelas perancangan pada subsistem memiliki hanya pengetahuan terbatas tentang kelas-kelas lain.
- Pembatasan ini dinamakan *Law of Demeter* yang menyatakan bahwa suatu metode seharusnya hanya mengirim pesan ke metode-metode pada kelas-kelas yang berdekatan.

4. PENDEKATAN PEMROGRAMAN TERSTRUKTUR

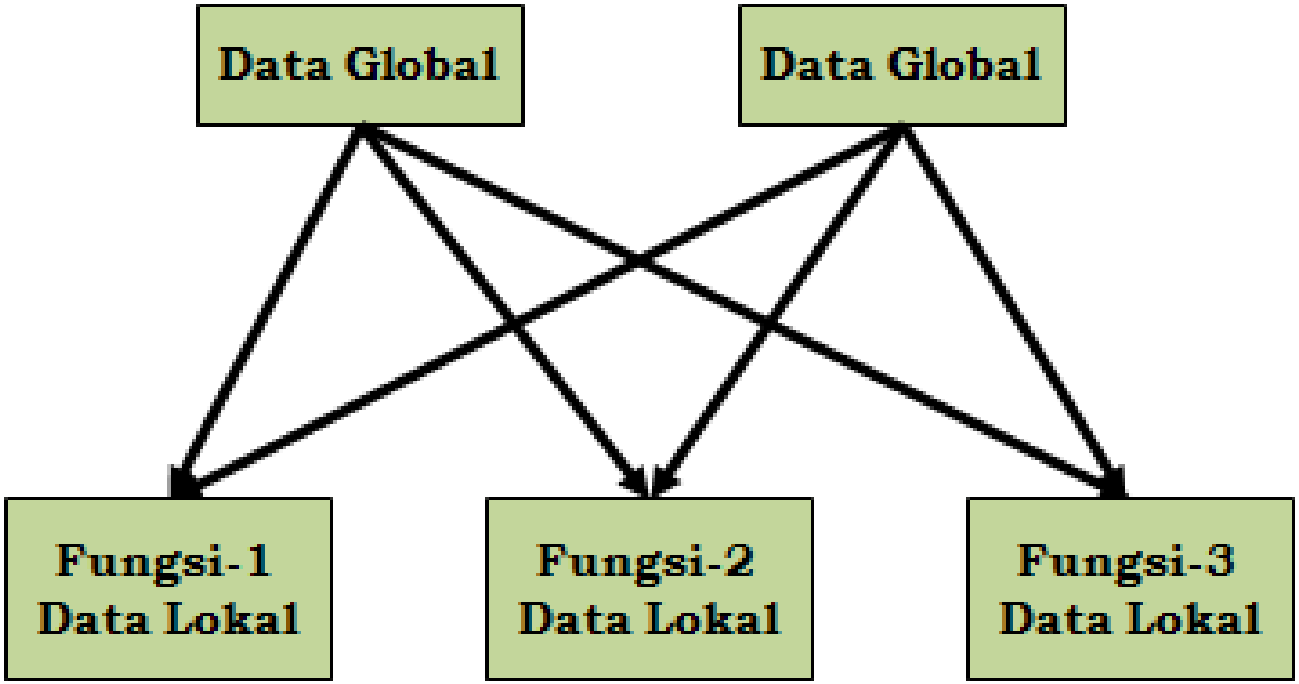
Karakteristik Pendekatan Berorientasi Prosedur/Fungsi:

a. Penekanan pada sesuatu yang harus dikerjakan (algoritma pemecahan masalah)
b. Program berukuran besar dipecah menjadi program-program yang lebih kecil
c. Kebanyakan fungsi/prosedur berbagi data global
d. Data bergerak secara bebas dalam sistem dari satu fungsi ke fungsi yang lain yang terkait
e. Fungsi-fungsi mentransformasi data dari satu bentuk ke bentuk yang lain
f. Menggunakan pendekatan *top-down*

Struktur Umum Pemrograman Terstruktur



Hubungan Data dan Fungsi pada Pemrograman Terstruktur

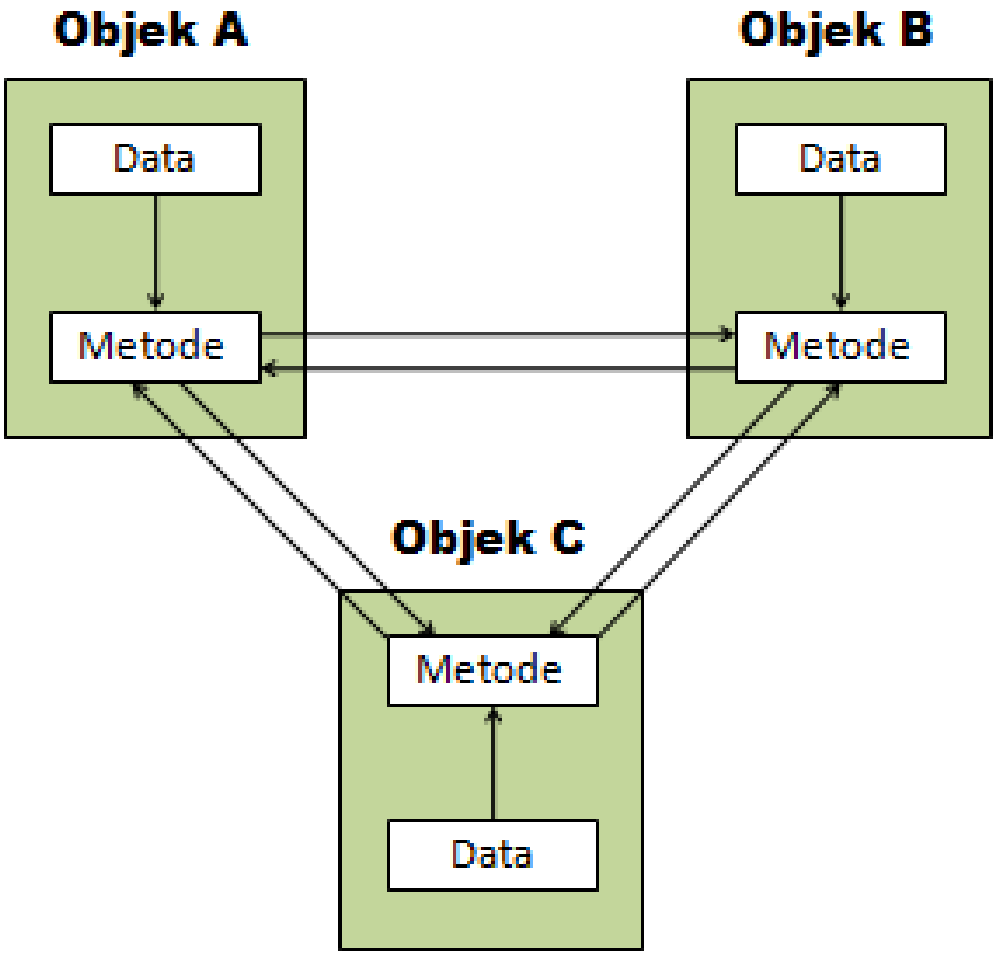


5. PENDEKATAN BERORIENTASI OBJEK

Karakteristik Pada Pendekatan Berorientasi Objek:

- Pendekatan lebih kepada data (bukan fungsi/prosedur)
- Program besar dibagi menjadi beberapa objek
- Struktur data dirancang dan menjadi karakteristik dari objek-objek
- Fungsi-fungsi yang mengoperasikan data tergabung dalam suatu objek yang sama
- Data tersembunyi dan terlindung dari fungs/prosedur yang ada di luar
- Objek-objek dapat saling berkomunikasi dengan saling mengirim *message* satu sama lain
- Menggunakan pendekatan *bottom-up*

Pengorganisasian Data dan Metode (Fungsi) pada Pendekatan Berorientasi Objek



6. Perbedaan Pemrograman Terstruktur dengan Pendekatan Berorientasi Objek

A. PEMROGRAMAN TERSTRUKTUR

- Permasalahan dilihat sebagai urutan sesuatu yang harus dikerjakan, seperti input – proses – output.
- Fokus utamanya pada fungsi atau prosedur
- Data global pada program yang sangat besar, sangat sulit untuk dilacak. Jika merevisi data global maka merevisi setiap fungsi yang menggunakan data global.
- Tidak menggambarkan kasus nyata dengan baik, karena fungsi-fungsi berorientasi pada aksi dan tidak berhubungan langsung dengan permasalahan.
- Kurang sempurna dalam menangkap kebutuhan *reusable components*, karena tidak ada standarisasi modul

B. PENDEKATAN BERORIENTASI OBJEK

- Penekanan pada “apa” yang dapat dilakukan oleh objek
- Pendekatan lebih kepada data
- Perubahan pada struktur data internal tidak mempengaruhi struktur data objek yang lain
- Penggunaan bersama untuk beberapa tingkat yang berbeda, seperti penggunaan bersama untuk disain dan kode
- Menggunakan pendekatan *bottom-up*
- Fungsi/prosedur berbagi data global

LATIHAN

- Dosen memberikan sebuah kasus untuk menjelaskan Encapsulasi, Pewarisan, dan Polimorfis, kemudian gambarkan dengan diagram objek
- Objek dilengkapi dengan atribut dan metode