Tujuan Pembelajaran

- Mengetahui konsep pemrograman berorientasi obyek

- Mengetahui perbedaan antara pemrograman berorientasi obyek dan pemrograman prosedural

- Mengetahui keuntungan dari pemrograman berorientasi obyek
- Mengetahui kelas dan obyek
- Mengetahui fitur dari pemrograman berorientasi obyek

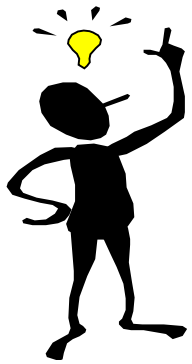
Ilustrasi

Bagaimana Anda menggambarkan ini dalam komputer ??

TABUNGAN

Tabungan

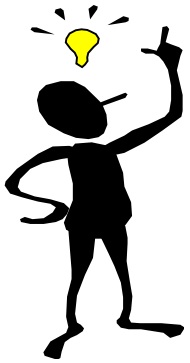
- Anda akan mencari atribut-atribut yang relevan dengan tabungan.
- Banyak ?? **Mungkin …**
- Tapi ambil saja 3 (sebagai contoh) untuk penyederhanaan dan memudahkan pemahaman ..



TABUNGAN
Nama Pemegang No Tabungan Saldo

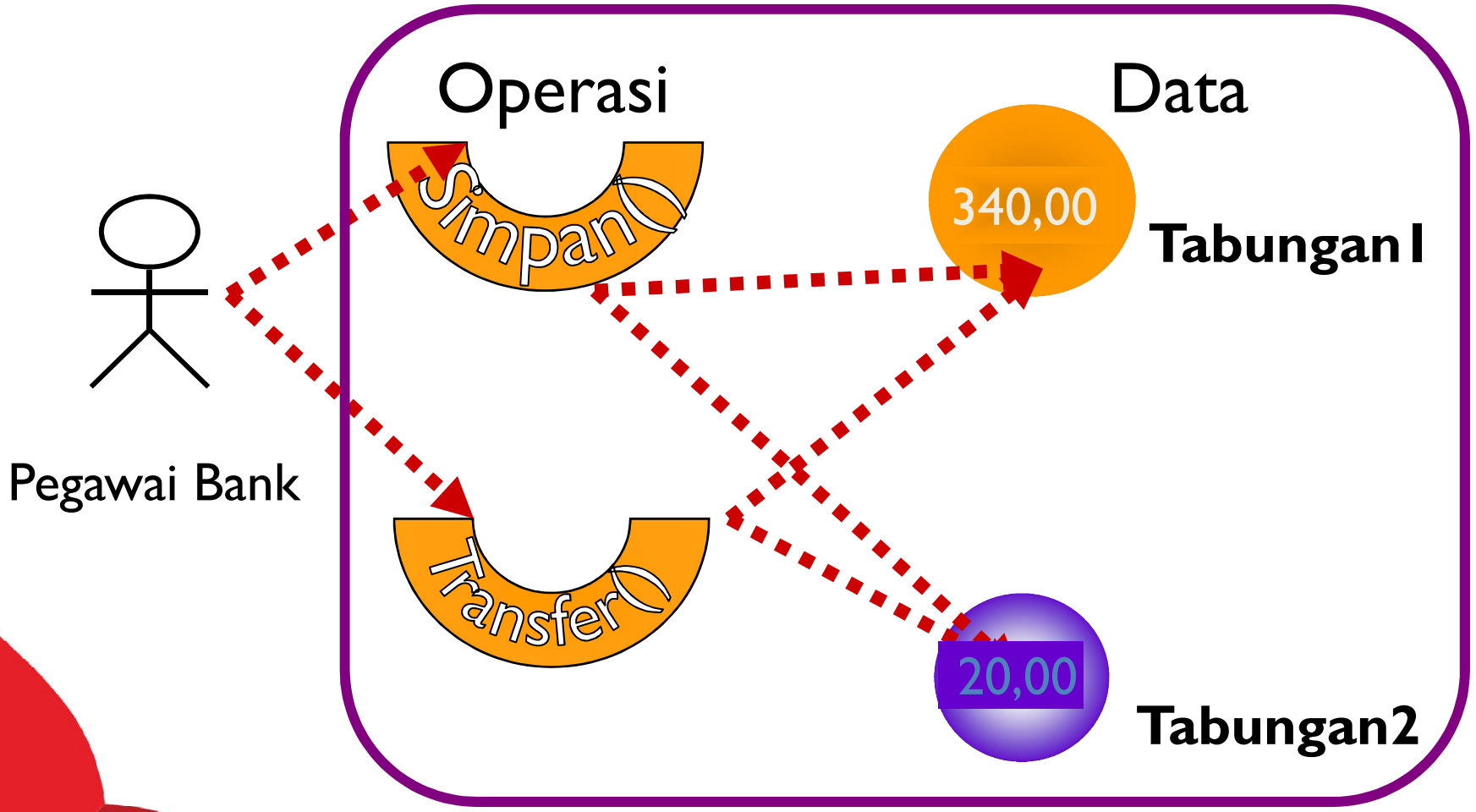
Tabungan

- **Operasi ??**
  - Simpan
  - Transfer
  - dlsb ….

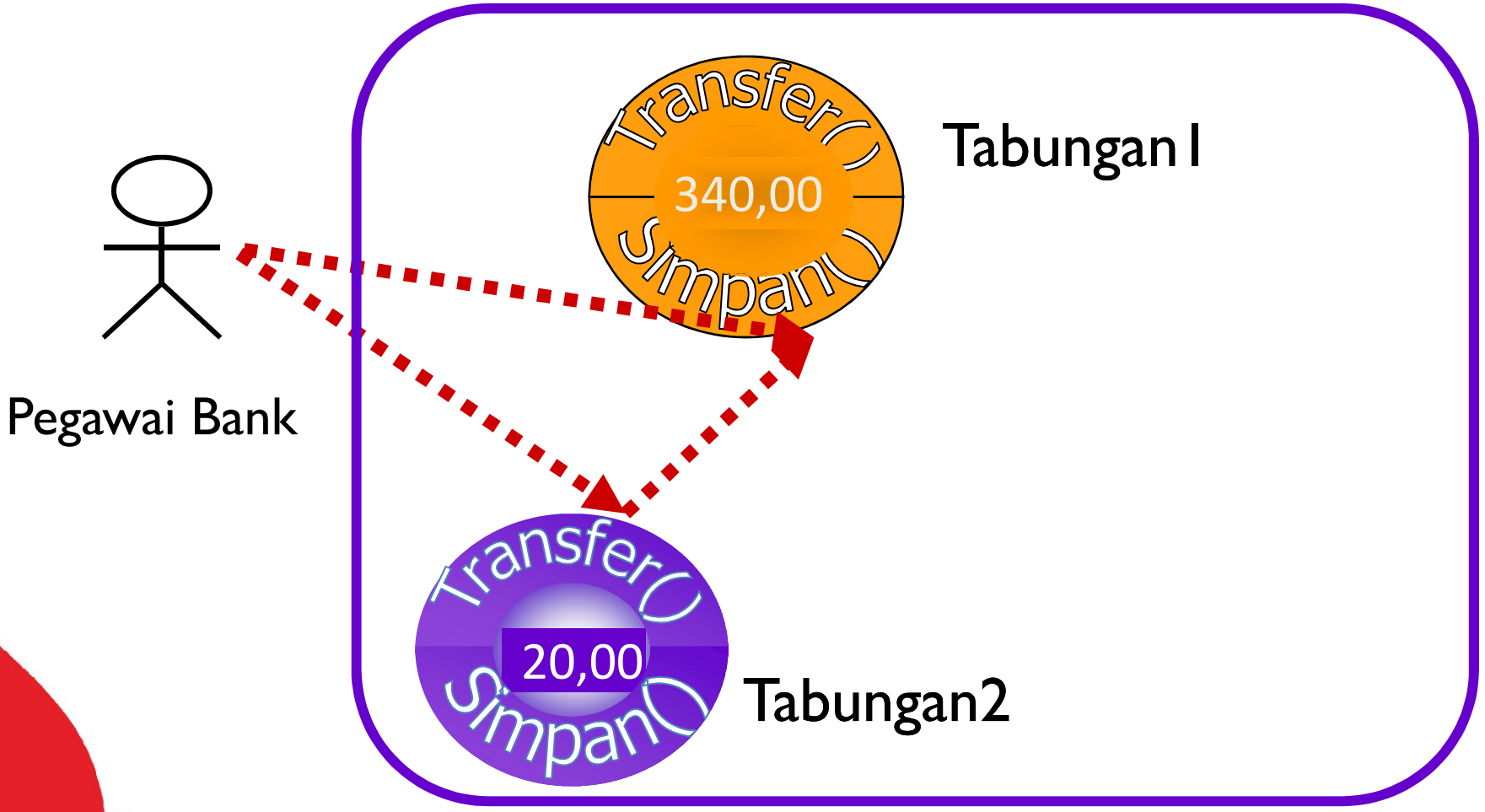


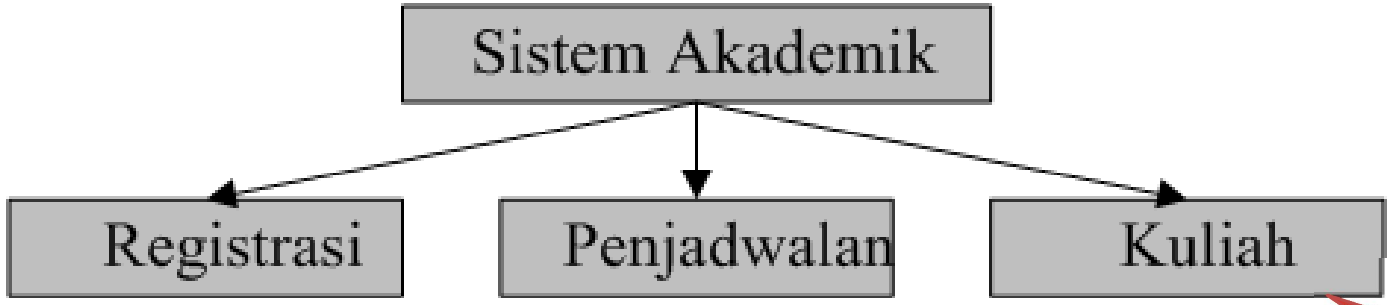
- **Implementasi**
  - Record – struktur
    - ✓Procedural style
  - Kelas
    - ✓Object Oriented style

Procedural System



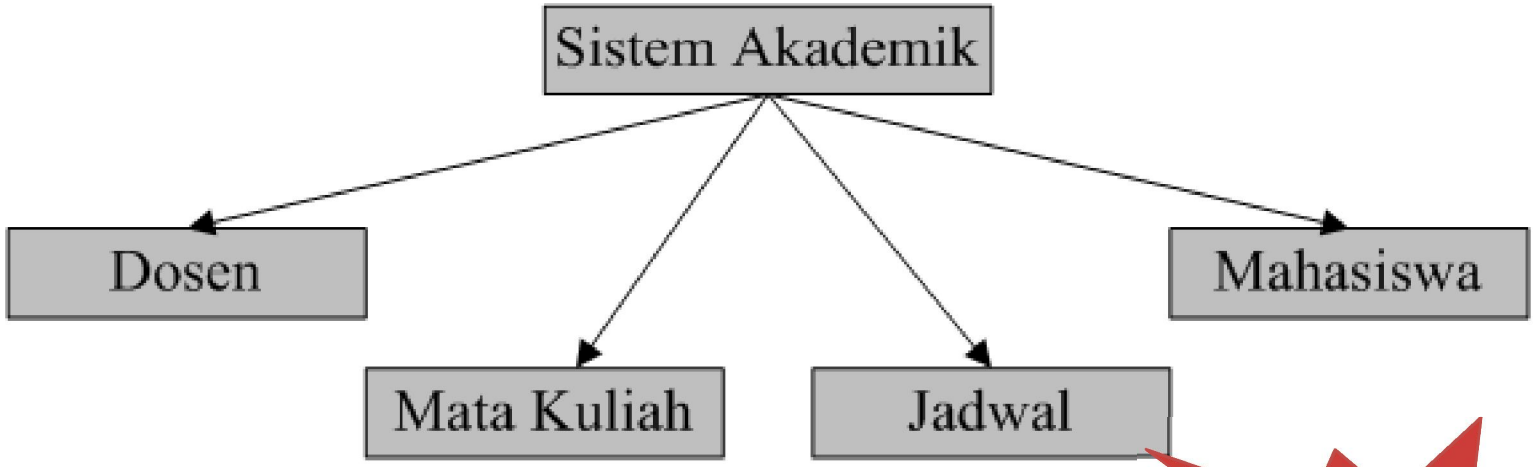
Object-Oriented System





Dekomposisi Berdasar Fungsi

Prosedural



Dekomposisi Berdasar Obyek

PBO

Sejarah PBO

- Dicetuskan pada tahun 1990-an yaitu memecahkan kebinggungan masalah pemrograman terstruktur dan ERD (Entity Relationship Diagram)

- Pemrograman berorientasi objek memungkinkan pengelolaan Data dan Fungsi dalam satu kesatuan yang disebut Objek

- Ide PBO diawali oleh ADT (Abstract Data Type) karena banyak kelemahan di sempurnakan menjadi konsep Objek

Karakteristik PBO

- Pendekatan lebih pada data bukan fungsi/prosedur
- Program besar dipecah-pecah menjadi objek-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 fungsi / prosedur yang ada diluar

- Objek-objek dapat saling berkomunikasi dengan saling mengirim pesan satu sama lain

Berorientasi Objek ?

Attribute :

Topi, Baju, Jaket,
Tas Punggung,
Tangan, Kaki, Mata

Behavior :

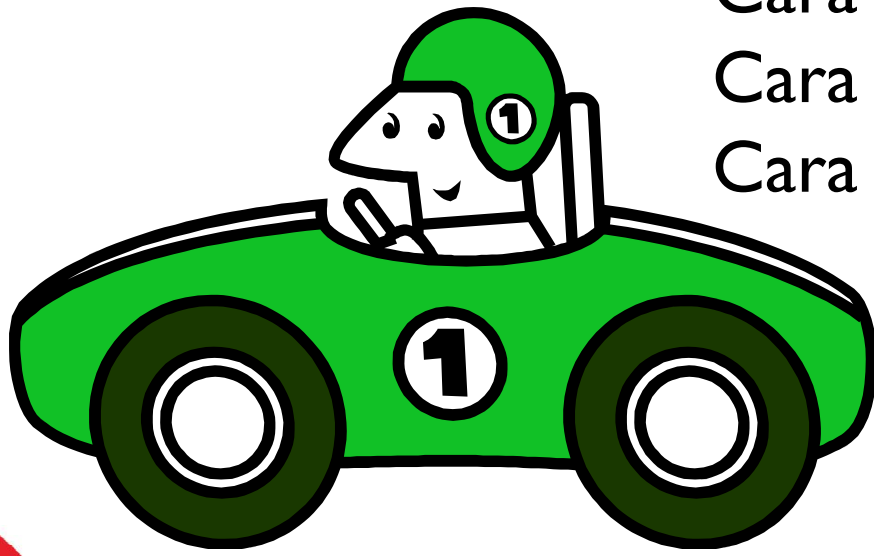
Cara Jalan ke Depan
Cara Jalan Mundur
Cara Belok ke Kiri
Cara Memanjat

Attribute (State):

Ban, Stir, Pedal Rem, Pedal Gas,
Warna, Tahun Produksi

Behavior:

Cara Menghidupkan Mesin
Cara Manjalankan Mobil
Cara Memundurkan Mobil



Attribute → Variable(Member)
Behavior → Method(Fungsi)

Perbedaan Class dan Object

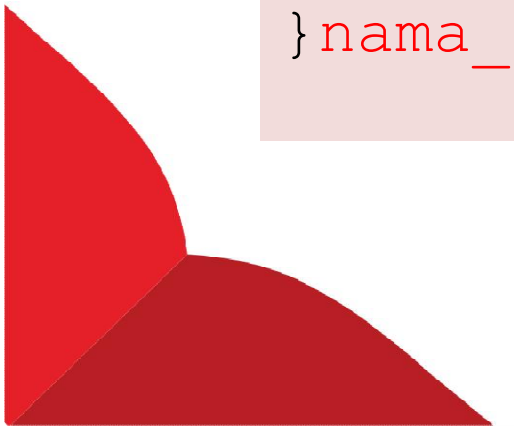
- **Class**: konsep dan deskripsi dari sesuatu
  - Class mendeklarasikan method yang dapat digunakan (dipanggil) oleh object

- **Object**: instance dari class, bentuk (contoh) nyata dari class
  - Object memiliki sifat independen dan dapat digunakan untuk memanggil method

- Contoh **Class** dan **Object**:
  - Class: mobil
  - Object: mobilnya pak Joko, mobilku, mobil berwarna merah

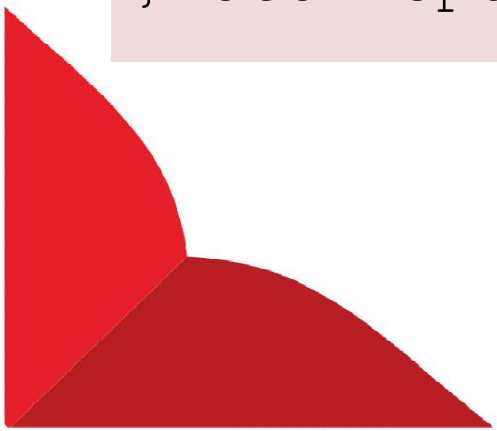
Deklarasi Kelas (*Class*) :

```
class nama_class
{
   Data elemen_class private;
   Data elemen_class private;
   ……
Public :
   Data elemen_class public;
   Data elemen_class public;
   ……
}nama_object;
```



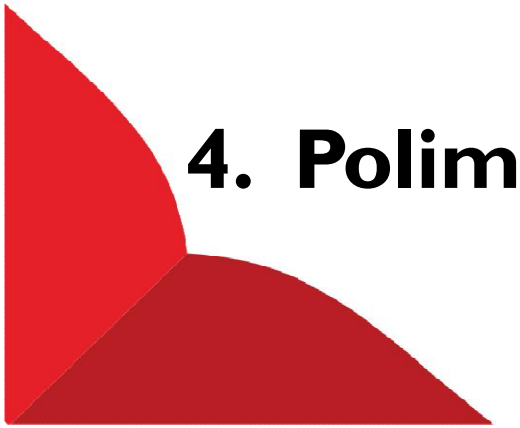
Listing Program Class Motor

```
class Motor //Nama Class
{
Public :
//Tipe anggota data bersifat public, default private
  char merk[50];
  char jenis[35]; //Nama anggota data
  float harga;
  int stock;
}Motor Sport; //Pendefinisian object
```



Fitur Pemrograman Berorientasi Obyek

1. Enkapsulasi (*Encapsulation*)
2. Abstraksi (*Abstraction*)
3. Pewarisan (*Inheritance*)
4. Polimorfisme (*Polymorphism*)



❑Enkapsulasi (*Encapsulation*)

- Enkapsulasi adalah suatu cara untuk menyembunyikan detail implementasi detail dari sebuah kelas

- Terdapat dua hal mendasar dari enkapsulasi yaitu :
  - *Information hiding* → penyembunyian detil dari atribut dan *method* pada sebuah kelas
  - *Interface* untuk pengaksesan data → suatu *method* untuk mengambil, memberikan atau mengubah suatu nilai

- **Contoh** : pada sebuah mesin ATM
  - Yang disediakan oleh mesin ATM adalah sebuah layar informasi dan perangkat untuk membaca kartu
  - Saat kartu ATM dimasukkan, maka pengguna akan memasukkan nomor pin

- Ketika pin dimasukkan, berarti telah terjadi proses pengaksesan data yang akan memberikan nomor pin untuk dikirimkan dan divalidasi pada server dari ATM tersebut
- Disini terjadi penyembunyian informasi tentang bagaimana cara kerja pengecekan validitas kartu, kecocokan pin yang dimasukkan, koneksi ke database server, dll, dimana hal-hal tersebut tidak perlu diketahui oleh pengguna tentang bagaimana cara kerjanya

❑Abstraksi

- Arti abstraksi mengacu kepada atribut dari sebuah obyek yang membedakan antara satu obyek dengan obyek yang lain

- Misalnya pada sebuah sistem akademik, dimana terdapat beberapa komponen misal saja mahasiswa

- Didalam obyek tersebut, terdapat suatu atribut yang akan saling membedakan antara satu obyek mahasiswa dengan mahasiswa lainnya, yaitu NIM, Nama, tanggal lahir, alamat, dan sebagainya.

- Dalam pemrograman berorientasi obyek konsep ini berada pada pembuatan sebuah kelas

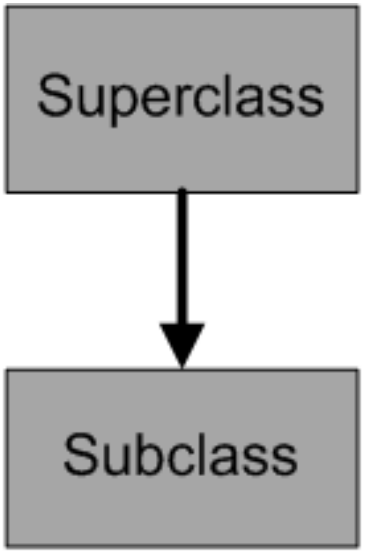
- Semua atribut dari obyek didefinisikan dalam sebuah kelas.

- Sebenarnya kelas tidak memiliki data, tetapi sebuah obyek-lah yang akan menyimpan data, karena obyek diciptakan dari sebuah kelas dan oleh sistem operasi akan dialokasikan sejumlah memori kepada obyek tersebut

❑Pewarisan

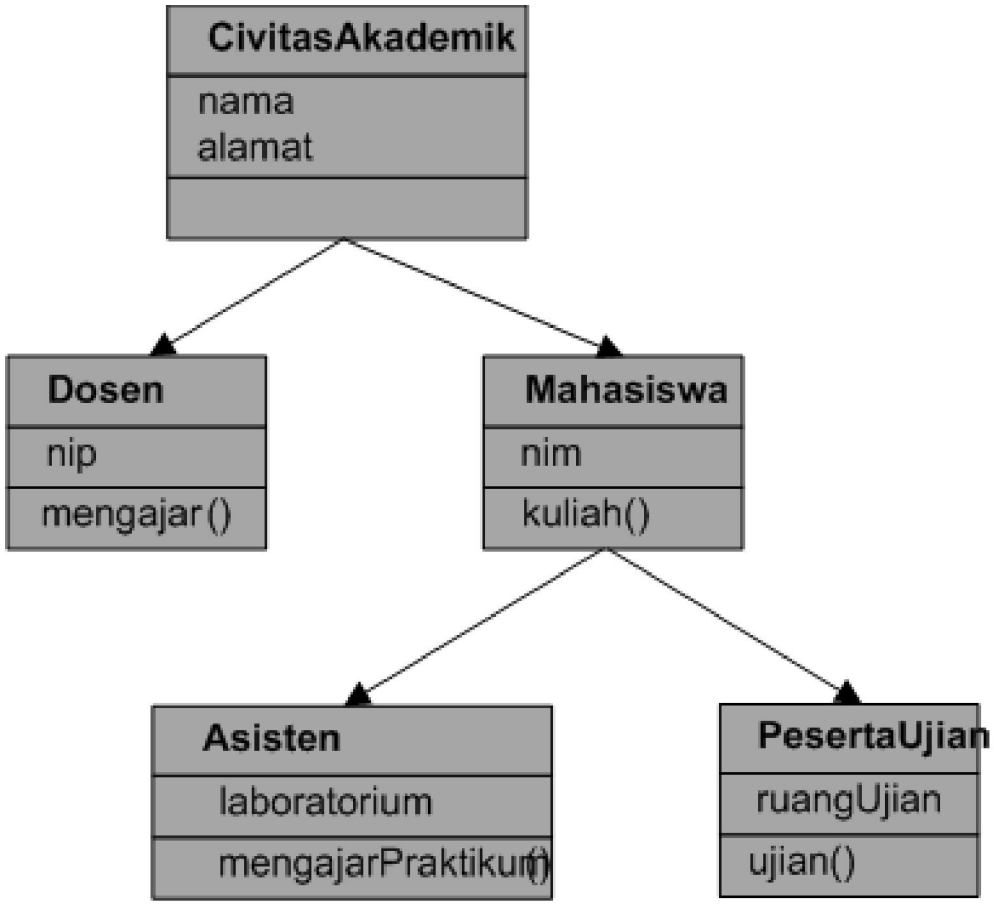
- Salah satu fitur yang paling kuat dalam pemrograman berorientasi obyek adalah penggunaan kode kembali (*code reuse*)

- Sekali sebuah prosedur dibuat, maka kita bisa menggunakannya berulang kali



1. Fitur pewarisan mengijinkan sebuah kelas yang dinamakan *superclass* untuk menurunkan atribut-atribut dan *methodnya* kepada yang lainnya, yaitu yang disebut *subclass* atau kelas turunannya

2. Hal ini akan mengijinkan pembuatan kelas baru yang didasarkan dari peng-abstrakan atribut-atribut dan *behaviour* yang sama



1. Dari contoh diatas bisa dilihat bahwa terdapat sebuah kelas CivitasAkademik yang memiliki 2 (dua) kelas turunan, yaitu Dosen dan Mahasiswa

2. Meskipun didalam kelas Dosen dan Mahasiswa tidak tercantum atribut Nama dan Alamat, tetapi keduanya tetap memiliki kedua atribut tersebut yang diturunkan dari kelas CivitasAkademik.

3. Begitu juga untuk kelas Asisten dan PesertaUjian. Keduanya tetap memiliki atribut Nama dan Alamat yang diturunkan dari kelas Mahasiswa, yang sebenarnya juga turunan dari kelas CivitasAkademik

4. Hal tersebut tidak hanya berlaku untuk atribut saja, tetapi juga *method-method* yang ada di kelas induknya

- Didalam pemrograman berorientasi obyek, terdapat beberapa jenis pewarisan, yaitu :
  1. ***Single inheritance***, yaitu hanya terdapat satu *superclass*
  2. ***Multiple inheritance***, yaitu terdapat lebih dari satu *superclass*

❑Polimorfisme

- Polimorfisme diturunkan dari bahasa latin yaitu *poly* yang berarti banyak dan *morph* yang berarti bentuk

- Polimorfisme sendiri berarti sesuatu yang memiliki banyak bentuk

- Sebagai contoh adalah sebuah obyek wanita, beberapa peran yang mungkin dimiliki adalah :
  - Bagi suami maka dia berperan sebagai seorang istri
  - Buat anak-anak berperan sebagai ibu
  - Di tempat kerja maka dia akan berperan sebagai seorang karyawan
  - Di tempat kuliah berperan sebagai mahasiswi
  - Di tempat arisan berperan sebagai ketua arisan

- Didalam pemrograman berorientasi obyek, polimorfisme adalah sebuah fitur yang memungkinkan kita untuk memberikan arti atau penggunaan yang berbeda bagi sebuah entitas dalam konteks yang berbeda-beda

- Polimorfisme bisa digunakan sebagai kategori umum bagi sebuah entitas dalam tindakan yang berbeda-beda.

Source Code 1

```
#include <iostream>
#include <conio.h>
 class Welcome
{
   public:
   void display()
   {
   cout<<" Selamat Datang di Praktikum PBO dengan C++"<<endl;
   }
};
 void main()
{
  Welcome W;
  W.display();
   getch();
}
```

Source Code 2

```
#include <iostream>
#include <conio.h>
class DemoObjectOriented
{
    public:
       void helloWorld(int jmlh_looping);
       int tambah(int bil1, int bil2);
};
void DemoObjectOriented::helloWorld(int jmlh_looping)
{
    for(int i=0;i<jmlh_looping;i++)
    {
        cout<<i+1<<". Hello World...."<<endl;
    }
}
int DemoObjectOriented::tambah(int bil1, int bil2)
{
   return(bil1+bil2);
}
```

```
int main()
{
    int bil1=5, bil2=2;
    int jmlh_looping=5;
    DemoObjectOriented Demo; //Pembentukan Objek
    cout<<endl;
    cout<<"-Display Hello World-"<<endl;
    Demo.helloWorld(jmlh_looping); //Pemanggilan
      Prosedur
    cout<<endl<<"-Pemanggilan Fungsi Tambah-
      "<<endl;
    //Pemanggilan Fungsi
       cout<<"Hasil Operasi Tambah :
       "<<Demo.tambah(bil1,bil2);
       cout<<endl;
    getch();
    return 0;
}
```

Source Code 3

```
#include<iostream.h>
#include<string.h>
#include <conio.h>
class Buku
{
      private :
            char judul[35];
            char pengarang[25];
            int  jumlah;
      public:
      void inisialisasiBuku(char *jdl,char *pngarang,int jmlh)
            {
                  strcpy(judul, jdl);
                  strcpy(pengarang, pngarang);
                   jumlah = jmlh;
            }
            void infoBuku()
            {
                  cout<<" Judul  :"<<judul<<endl;
                  cout<<" Pengarang :"<<pengarang<<endl;
                  cout<<" Jumlah buku :"<<jumlah<<endl;
            }
};
```

```
int main ()
{
    Buku novel,fiksi;
    novel. inisialisasiBuku("Meriam Benteng
navarone","Alistair
    Maclean",12);
    fiksi. inisialisasiBuku("Jurassic park","Michael
Crichton",3);
    novel. infoBuku();
    fiksi. infoBuku();
    getch();
    return 0;
}
```


Backspace
Thank You
Page Up
Home
Page Down
Shift
Insert