IPB University
Bogor Indonesia

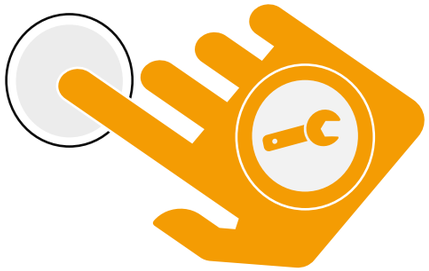
KOM120C -- BAHASA PEMROGRAMAN

Object and Class

- Review OOP
- Object and Class

Tim Pengajar Bahasa Pemrograman IPB University

Review : PRINSIP DASAR OOP

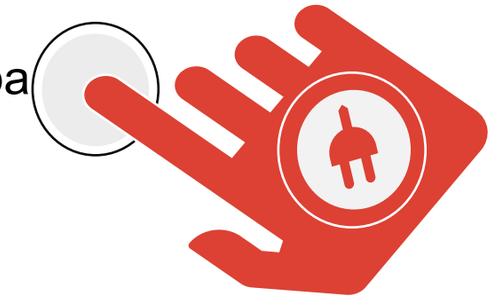


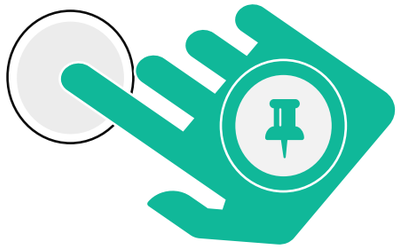
ENCAPSULATION

- Membungkus prosedur dan data dalam satu objek.
- OOP memodelkan objek yang ada di dunia nyata ke dalam software objek dalam pemrograman.
- Implementasi dalam bentuk **Class**.
- Berfungsi sebagai ADT (*Abstract Data Type*)

POLYMORPHISM

Tiap objek tahu siapa dirinya





INHERITANCE

- Pewarisan sifat objek
- Mengembangkan class baru dari class yang sudah ada.

Review : Membuat Objek dalam Pemrograman

Objek diimplementasikan dalam bentuk class.

Struktur class dalam C++:

```
class <class-name> {
    // data/atribut
    ...
    ...
    // prosedur/fungsi
    ...
    ...
}
```

Setiap anggota atau elemen dari class jenis akses (*access modifier*), yang menunjukkan bisa tidaknya elemen tersebut diakses suatu class:

- **private**
  hanya dapat diakses di dalam class itu sendiri
- **protected**
  dapat diakses oleh class itu sendiri beserta turunannya
- **public**
  dapat diakses di luar class

By default, akses terhadap elemen class adalah private.

Contoh Kasus: Objek "person“

Materi Praktikum 02 : Class Person

Deskripsi

Buatlah program C++ yang mengimplementasikan sebuah class bernama Person, yang memiliki atribut nama (string), usia (int), tinggi (int), dan berat (double). Default contructor class ini akan secara otomatis mengisi atribut nama dengan string kosong, dan atribut lainnya diisi dengan nilai 0 (nol). Class Person memiliki prosedur bernama setPerson untuk mengisi nilai setiap atribut. Tambahkan prosedur lainnya agar dapat memproses apa yang diminta oleh program utama (main) yang diuraikan di bawah ini.

Program akan membaca n baris data yang berisi nama, usia, tinggi badan, dan berat badan. Gunakan class Person untuk menyimpan n data tersebut. Selanjutnya program akan menuliskan seluruh nama dan usia, dan dilanjutkan dengan menuliskan rata-rata tinggi badan, serta banyaknya orang (person) yang tinggi badannya di atas nilai rata-rata. Program harus benar-benar mengimplementasikan class bagi obyek Person. Oleh karena itu, tidak diperkenankan mengolah data yang berasal dari variabel biasa (bukan atribut dari class). Semua atribut class dikelompokkan sebagai private.

Format Masukan

Baris pertama adalah sebuah bilangan bulat n, 1≤n≤100, yang menunjukkan banyaknya baris data yang akan dibaca. Dan n baris berikutnya berisi data nama, usia, tinggi badan, dan berat badan yang masing-masing dipisahkan oleh satu spasi. Nama orang berupa string dengan hanya terdiri atas satu kata.

Sebagai implementasi software dari objek "person" atau "orang".

Memiliki atribut : nama, usia, tinggi, berat.

Memiliki behaviour (perilaku, methods, etc):

- Inisialisasi (constructor)
- Dapat diberi nilai setiap atribut (mutator) : setPerson()
- Dapat diakses nilai setiap atribut (accessor)

Butuh sebuah class lain sebagai implementasi dari objek "beberapa person "

Desain Class Person

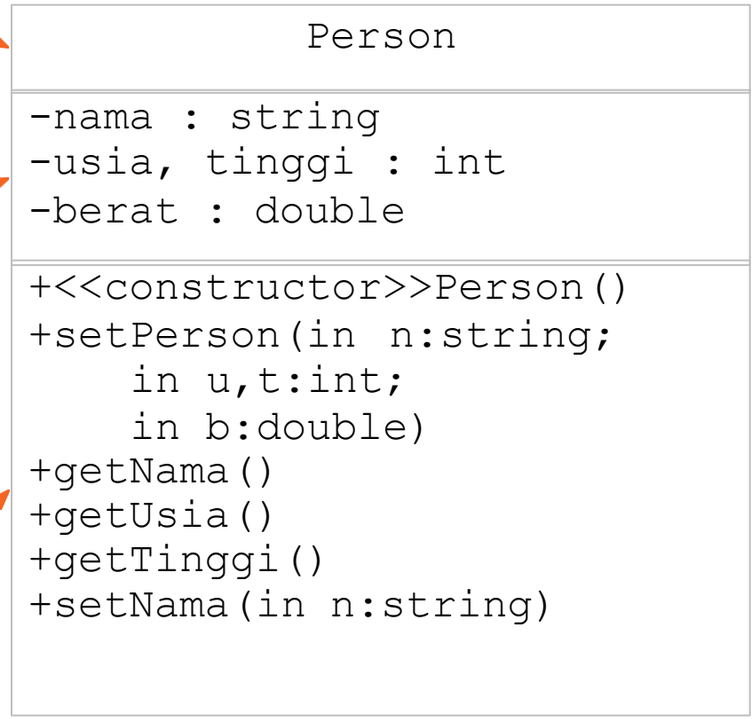
Materi Praktikum 02 : Class Person

Sebagai implementasi software dari objek "person" atau "orang".

Memiliki atribut : nama, usia, tinggi, berat.

Memiliki behaviour (perilaku, methods, etc) :

- Inisialisasi (constructor)
- Dapat diberi nilai atribut (mutator)
- Dapat diakses nilai atribut (accessor)



Presentasi berupa UML
(*Unified Modeling Language*)

Class Person

Materi Praktikum 02 : Class Person

```
class Person
{
    string nama;
    int usia, tinggi;
    double berat;

    public:
        Person() {nama=""; usia=tinggi=0; berat=0.0;};
        void setPerson(string n, int u, int t, double b)
            { nama=n; usia=u; tinggi=t; berat=b; }
        string getNama() { return nama; }
        int getUsia() { return usia; }
        int getTinggi() { return tinggi; }
        void setNama(string n) { nama=n;}
};
```

Desain Class nPerson → People

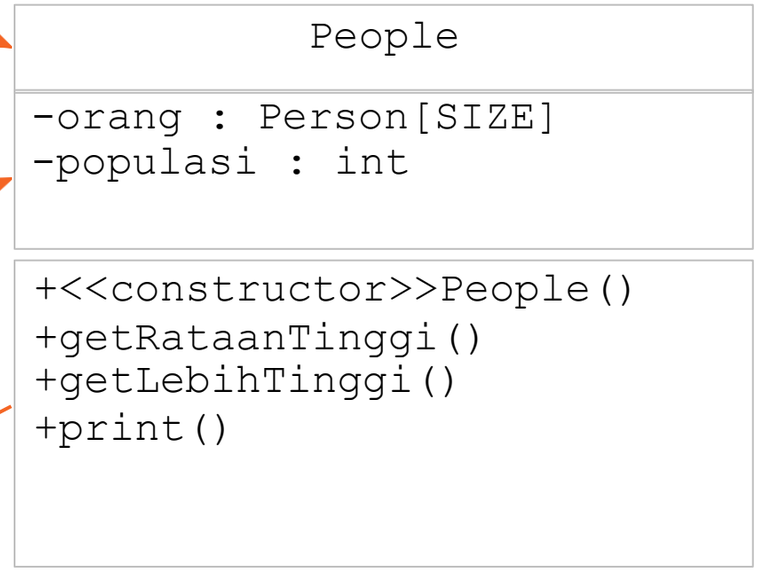
Materi Praktikum 02 : Class Person

Sebagai implementasi software dari objek "beberapa person"

Memiliki atribut : nama, usia, tinggi, berat.

Memiliki behaviour (perilaku, methods, etc) :

- Inisialisasi (constructor)
- Dapat diberi nilai atribut (mutator)
- Dapat diakses nilai atribut (accessor)



IPB University
Bogor Indonesia

Class People

Materi Praktikum 02 : Class Person

```
class People
{
    Person orang[SIZE];
    int populasi;
    public:
        People() { populasi=0; }
        void add(Person p)
          { orang[populasi++]=p; }
        void print()
          {
             for(int i=0; i<populasi; i++)
             cout << orang[i].getNama()
                  << " " << orang[i].getUsia()
                  << endl;
          }
        double rataanTinggi();
        int lebihTinggi();
};
```

```
double People::rataanTinggi()
{
    int sum=0;
    for(int i=0; i<populasi; i++)
        sum+=orang[i].getTinggi();
    return (double)sum/populasi;
}
```

```
int People::lebihTinggi()
{
    double r = rataanTinggi();
    int res=0;
    for(int i=0; i<populasi; i++)
        if (orang[i].getTinggi()>r)
           res++;
    return res;

}
```

Driver

Materi Praktikum 02 : Class Person

```
int main()
{
    int n;
    cin >> n;
    People mhs;
    for(int i=0; i<n; i++)
    {
        string nama;
        int usia, tinggi;
        double berat;
        Person p;
        cin >> nama >> usia >> tinggi >> berat;
        p.setPerson(nama,usia,tinggi,berat);
        mhs.add(p);
    }
    mhs.print();
    printf("%.2lf\n",mhs.rataanTinggi());
    printf("%d\n", mhs.lebihTinggi());
    return 0;
}
```

MUTATOR and ACCESSOR

- **Mutator** merupakan **fungsi** yang dapat **memberi atau mengubah nilai** atribut/data dalam class. Fungsi ini diperlukan terutama bagi data yang memiliki akses private.
- **Accessor** merupakan **fungsi** yang digunakan untuk **membaca** nilai atribut/data pada class. Fungsi ini diperlukan karena data yang akan diakses merupakan private atau protected.

CONSTRUCTOR

Constructor

adalah fungsi khusus dalam class yang akan **dieksekusi terlebih dahulu saat pembuatan *instance*** (*object*).

Constructor biasanya digunakan untuk **insialisasi atribut.**
Misalnya memberi nilai 0 untuk usia, tinggi, dan berat.

Karakteristik Constructor

- Memiliki identitas (nama fungsi) yang sama dengan nama class.
- **Tidak memiliki return type.**
- **Tidak dapat dipanggil dari luar class.**

Problem

Bagaimana desain objek untuk contoh kasus data berikut?

Contoh kasus:
Membaca N data orang sesuai atribut yang ada dengan format CSV, dan menyimpan ke dalam array.

Contoh masukan:

```
5
Noh Jong Hyun,17,165,60.5
Yoon Ji Ho,25,170,48.7
Nam Se Hee,30,180,60.2
Ri Jung Hyuk,35,182,62.8
Yoon Se Ri,30,170,58.9
```

Pada class Person, tambahkan prosedur untuk menerima data string (satu baris data) sesuai format CSV. Di dalamnya ada parsing untuk menyimpan ke dalam setiap atribut. Misal:

```
void addPerson(string csv);
```

Class Person

Membaca data CSV

```
void Person::addPerson(string str)
{
    istringstream ss(str);
    string token;

    getline(ss, token, ','); nama=token;

    getline(ss, token, ',');
    stringstream d1(token); d1>>usia;

    getline(ss, token, ',');
    stringstream d2(token); d2>>tinggi;

    getline(ss, token, ',');
    stringstream d3(token); d3>>berat;
}
```

```
#include <iostream>
#include <cstdio>
#include <sstream>
#include <string>
#define SIZE 100
using namespace std;

class Person
{
    string nama;
    int usia, tinggi;
    double berat;

    public:
        Person() {nama=""; usia=tinggi=0; berat=0.0;};
        void setPerson(string n, int u, int t, double b)
            { nama=n; usia=u; tinggi=t; berat=b; }
        string getNama() { return nama; }
        int getUsia() { return usia; }
        int getTinggi() { return tinggi; }
        void setNama(string n) { nama=n;}
        void addPerson(string csv);
};
```

Driver

Membaca data CSV

```
int main()
{
    int n;
    scanf("%d ", &n);
    People mhs;
    for(int i=0; i<n; i++)
    {
        Person p;
        string line;
        getline(cin,line);
        p.addPerson(line);
        mhs.add(p);
    }
    mhs.print();
    printf("%.2lf\n",mhs.rataanTinggi());
    printf("%d\n", mhs.lebihTinggi());
    return 0;
}
```

LATIHAN

Salah Satu Materi Praktikum 03

Sebagai latihan, lengkapi member functions dari class Person tersebut untuk hal-hal berikut:

- mengakses data berat dengan fungsi asesor getBerat().
- menghitung indeks massa tubuh (IMT) menggunakan fungsi getIMT(), dimana nilai IMT adalah berat (kg) dibagi dengan kuadrat dari tinggi badan (meter).
- menentukan status gizi orang tersebut menggunakan fungsi getStatusGizi(), dimana seseorang berstatus "sangat kurus" jika IMT<17.0, "kurus" jika 17.0≤IMT<18.5, "normal" jika 18.5≤IMT<25.0, "gemuk" jika 25.0≤IMT<28.0, dan "sangatgemuk" jika 28.0≤IMT