Pertemuan 1

Pengenalan Pemrograman Berbasis Obyek (PBO)

(Class & Object)

A. Tujuan

1. Mahasiswa memahami yang dimaksud Pemrograman Berbasis Obyek(PBO)
2. Mahasiswa memahami Class
3. Mahasiswa memahami Object

B. Dasar Teori

Pemrograman Berbasis Obyek(PBO) atau *Object Oriented Programming (OOP)* adalah paradigma pemrograman yang berorientasi kepada object. semua fungsi/subrutin, data, dan pengolahan data akan dibentuk dan dikelola dalam kelas-kelas dan object-object, sehingga setiap object dapat memiliki, data, sifat dan tugas masing-masing. Sebuah objek bisa memiliki **data member** dan **member function** (istilah lain dari variabel dan function yang berada di dalam objek). Object-object tersebut dapat berkerja sendiri dan juga saling bekerja sama, misalkan untuk saling berhubungan, menerima data, mengirim data, dan memproses data kepada object lainnya.

Dalam implementasi tidak semua bahsa pemrograman bisa diterapkan konsep Pemrograman Berbasis Obyek. Dalam beberapa kasus, membuat kode program dengan cara biasa (tanpa objek), bisa selesai lebih cepat dan lebih sederhana. “Pemrograman biasa” ini disebut juga dengan *pemrograman prosedural* (**Procedural Programming**) merupakan teknik pemrogram dengan cara membagi program menjadi bagian-bagian atau fungsi-fungsi kecil (*procedure*), kemudian menyatukannya untuk menghasilkan aplikasi secara utuh. sebuah objek bisa memiliki data dan function sendiri. Setiap objek ditujukan untuk mengerjakan sebuah tugas, dan menghasilkan nilai akhir untuk selanjutnya digunakan oleh objek lain.

Pada aplikasi yang cukup besar dan komplek hal ini akan menyulitkan dalam proses pemrograman/coding, salah satu masalah yang terjadi adalah timbulnya banyak fungsi dan variable yang digunakan. Konsep OOP ‘*Object Oriented Programming’* menawarkan metode yang bertujuan untuk memberikan pola dan kerangka dalam mengembangkan program(coding), pola tersebut dapat memberikan kemudahan,

fleksibilitas, kemudahan pembuatan, pengembangan, dan perawatan pada program(coding).

B.1. Class

Class adalah kerangka dasar / template yang akan digunakan oleh object. **class** bisa diibaratkan dengan **Mobil**. Kita tahu bahwa *mobil* memiliki ciri-ciri seperti *merk*, memiliki *mesin*, memiliki *roda*, dan beberapa bagian lain yang menyatakan sebuah benda tersebut adalah *Mobil*. Selain memiliki ciri-ciri, sebuah mobil juga bisa dikenakan tindakan, seperti: *menghidupkan mesin mobil* atau *membuka pintu, dll*. gambaran umum dari tentang sebuah benda. Di dalam pemrograman nanti, contoh lain dari class adalah **Mahasiswa, Sekolahan**, **Dosen, Makanan**, dll.

Penulisan **class** pada bahasa c++ diawali dengan *keyword* **class**, kemudian diikuti dengan *nama dari class tersebut*. Aturan penulisan nama **class** sama seperti aturan penulisan *variabel* dalam C++ (lebih tepatnya aturan **identifier**), yakni tidak boleh diawali angka dan tidak boleh mengandung spasi. Pada umumnya para programmer menulisakan nama class menggunakan **PascalCase** atau **UpperCamelCase**. Yakni cara penulisan dimana setiap kata diawali dengan huruf besar, termasuk huruf pertama. pendefinisian class juga harus dilakukan di luar function **main()**, sama seperti pendefinisian fungsi/subrutin.

```
1  #include <iostream>
2
3  using namespace std;
4
5  class Mobil {
6      // isi dari class Mobil...
7      // isi dari class Mobil...
8  };
9
10 int main(int argc, const char * argv[]) {
11
12     // kode program
13     return 0;
14 }
```

Akses specifier : digunakan untuk mendifinisikan hak akses kepada anggota yang ada didalam class. Hak akses terhadap isi dari class harus didefiniskan lebih dulu, Jika tidak didefinisikan maka anggota *class* secara otomatis memiliki hak akses *private*. Hak akses pada class terdiri dari:

1. Private, hanya member function atau data member yang ada didalam kelas itu yang dapat mengakses.

2. Public, member function atau data member dapat diakses didalam dan diluar class.
3. Protected, member function atau data member Hanya dapat diakses oleh fungsi tertentu.

```cpp
#include <iostream>

using namespace std;

class Mobil {
    public  :
        string merk;
        string tipe;
        double jumlah_kursi;
    private :
        string nomor_STNK;
};

int main(int argc, const char * argv[]) {

    // kode program
    return 0;
}
```

Data member (atau kadang juga dengan *atribut* atau *properti*) adalah data yang terdapat dalam sebuah **class**. **data member** adalah sebuah variabel yang terletak di dalam class. Seluruh aturan tata cara penamaan data member dan aturan tipe data member sama dengan aturan penamaan variabel biasanya.

Member function (kadang disebut juga sebagai *method*) adalah tindakan yang bisa dilakukan di dalam class. Pada dasarnya Member function sama seperti ***function*** / **subrutin** yang berada di dalam **class**. Seluruh sifat function bisa diterapkan ke dalam member function, seperti bisa di isi argumen/parameter, mengembalikan suatu data/nilai (dengan keyword *return*), dll.

```cpp
class Mobil {
    public  :
        string merk;
        string tipe;
        double jumlah_kursi;

    void hidupkan_mesin(){
        // isi program untuk menghidupkan mesin
    }
    void buka_pintu(){
        // isi program untuk buka pintu
    }

    private :
        string nomor_STNK;
};
```

B.2. Object

Object adalah komponen yang dibuat dari **class**, atau bisa juga disebut hasil konkrit dari **class**. Sebagai contoh, sebuah objek yang dibentuk dari dari **class Mobil** akan memiliki seluruh bagian dan fungsi yang ada di class mobil, termasuk data member dan member functionnya.

Proses pembuatan atau deklarasi Sebuah obyek dilakukan dengan menulis nama class, lalu diikuti dengan nama variabel di dalam function **main()**. Hampir sama seperti pembuatan variabel biasa, hanya saja dalam variabel biasa kita menulis tipe data terlebih dahulu. Sementara untuk pembuatan object, tipe data ini diganti dengan nama class. Berikut cara membuat object **kampus_A, kampus_B** dan **kampus_C** yang dibuat dari **class Mobil.**

```
#include <iostream>

using namespace std;

class Mobil {
    public  :
        string merk;
        string tipe;
        double jumlah_kursi;
};

int main(int argc, const char * argv[]) {

    Mobil kampus_A;
    Mobil kampus_B;
    Mobil kampus_C;

    return 0;
}
```

kampus_A, kampus_B dan **kampus_C** merupakan objek dari **class Mobil**. Ketiga objek akan memiliki seluruh data member dan member function yang telah dirancang dalam **class Mobil**.