

**LAPORAN HASIL PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN II**

6



NAMA : PRAYOGA BETRIO L
NIM : 193030503060
KELAS : A
**MODUL : I (DASAR PEMROGRAMAN
BERORIENTASI OBJEK)**

**JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS PALANGKA RAYA**

2020

**LAPORAN HASIL PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN II**



Nama : PRAYOGA BETRIO L
NIM : 193030503060
Kelas : A
Modul : (DASAR PEMROGRAMAN
BERORIENTASI OBJEK)

Komposisi	MAX	Nilai
BAB I Tujuan dan Landasan Teori	10	9
BAB II Pembahasan	60	56
BAB III Kesimpulan	20	15
Daftar Pustaka	5	4
Lampiran	5	3
Jumlah	100	87

Penilai

Asisten Praktikum

Diana

BAB I

TUJUAN DAN LANDASAN TEORI

1.1. TUJUAN

Setelah menyelesaikan modul ini, mahasiswa diharapkan mampu :

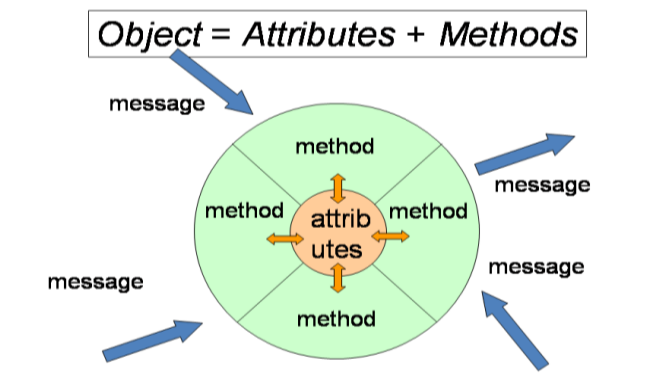
- Memahami dasar-dasar pemrograman berorientasi objek
- Memahami enkapsulasi
- Membuat kelas dan objek

1.2. DASAR TEORI

1.2.1. Perbedaan Pemrograman Tradisional dan Berorientasi Objek

Perbedaan pemrograman tradisional dan berorientasi objek adalah pada cara menyelesaikan suatu permasalahan. Pada pemrograman tradisional dalam memecahkan suatu masalah, masalah akan di bagi menjadi fungsi – fungsi yang lebih kecil, sedangkan pada pemrograman berorientasi objek (PBO) setiap masalah diselesaikan dengan cara di bagi-bagi ke dalam objek-objek.

Pada PBO di lakukan pembungkusan data (attributes) dan fungsi (behavior) ke paket yang di sebut kelas. Attributes merupakan data yang menggambarkan status internal sebuah objek dan biasanya merupakan “member variables” pada C++ , tidak dapat di akses dari luar (enkapsulasi), dan juga sebagai “state”, Methods merupakan fungsi yang mengakses status internal sebuah objek dan biasanya merupakan “ member functions” pada C++, dapat di akses dari luar, memanipulasi atribut, dan di sebut juga “behavior”. Berikut ini merupakan gambaran mengenai objek.



Kelas (class) terdiri dari model objek yang memiliki atribut (data members) dan *Behaviors* (*member function*), dan *Member functions* yaitu *Methods* yang dipanggil sebagai response sebagai pesan. Kelas didefinisikan dengan keyword **class**.

Mode Akses akses yang ada pada kelas ada tiga, yaitu **private** yang merupakan *default* mode akses dan dapat diakses oleh *member functions*, **public** yang dapat diakses oleh setiap Accesible fungsi dalam program, dan **protected** yang biasanya digunakan untuk pewarisan.

Fungsi *Constructor* merupakan *member functions* khusus yang menginisialisasi data members dan memiliki nama yang sama dengan nama kelas. Fungsi *Constructor* dipanggil saat membuat objek dari kelas dan tidak memiliki tipe balikan.

Member functions yang didefinisikan di luar kelas dilakukan dengan menggunakan *binary scope resolution operator* (::) yang berfungsi untuk “mengikat” nama fungsi ke nama kelas dan mengidentifikasi fungsi dari suatu kelas tertentu.

Berikut ini merupakan format dari *member functions*.

NilaiBalikan NamaKelas::NamaFungsi() {

...
}

Members functions yang didefinisikan di dalam kelas tidak membutuhkan scope resolution operator dan nama kelas.

1.2.2. Konsep Dasar Pemrograman Berorientasi Objek (PBO)

Konsep dasar Pemrograman Berorientasi Objek (object-oriented programming) merupakan pemrograman yang berorientasikan kepada objek. Semua data dan fungsi di dalam paradigma ini dibungkus dalam *kelas-kelas* atau *objek-objek*.

Setiap objek dapat menerima pesan, memproses data, dan mengirim pesan ke objek lainnya.

Model data berorientasi objek dikatakan dapat memberi fleksibilitas yang lebih, kemudahan mengubah program, dan digunakan luas dalam teknik piranti lunak skala besar. Lebih jauh lagi, pendukung PBO mengklaim bahwa PBO lebih mudah dipelajari bagi pemula dibanding dengan pendekatan sebelumnya, dan pendekatan OOP lebih mudah dikembangkan dan dirawat.

a. Istilah-istilah dalam PBO:

1. Objek

Untuk mempermudah pemahaman, maka disini akan dijelaskan melalui analogi. Pada dasarnya semua benda yang ada di dunia nyata dapat dianggap sebagai objek. Misalnya rumah, mobil, sepeda, motor, gelas, komputer, meja, sepatu, dll. Setiap objek memiliki atribut sebagai status (state) dan tingkah laku sebagai behavior.

Contoh objek : Motor. Maka attribute (state) nya adalah pedal, roda, jeruji, speedometer, warna, jumlah roda. Sedangkan tingkah laku (behavior) nya adalah kecepatan menaik, kecepatan menurun, dan perpindahan gigi motor.

Analogi pemrograman berorientasi objek sama dengan penggambaran pada dunia nyata seperti contoh di atas. Dalam

PBO, state disimpan pada variabel dan tingkah laku disimpan pada method.

Dalam bahasa teoretis PBO, Objek berfungsi untuk membungkus data dan fungsi bersama menjadi satu unit dalam sebuah program komputer. Objek merupakan dasar dari modularitas dan struktur dalam sebuah program komputer berorientasi objek.

2. Class

Class yaitu template untuk membuat objek. Class merupakan prototipe atau blue prints yang mendefinisikan variabel-variabel dan method-method secara umum. Objek merupakan hasil instansiasi dari suatu class. Proses pembentukan objek dari suatu kelas disebut sebagai instantiation. Objek disebut juga sebagai instances.

Dalam bahasa teoretis PBO, class merupakan kumpulan atas definisi data dan fungsi-fungsi dalam suatu unit untuk suatu tujuan tertentu. Sebagai contoh 'class of dog' adalah suatu unit yang terdiri atas definisi-definisi data dan fungsi-fungsi yang menunjuk pada berbagai macam perilaku/turunan dari anjing. Sebuah class adalah dasar dari modularitas dan struktur dalam pemrograman berorientasi object.

Sebuah class secara tipikal sebaiknya dapat dikenali oleh seorang non-programmer sekalipun terkait dengan domain permasalahan yang ada, dan kode yang terdapat dalam sebuah class sebaiknya (relatif) bersifat mandiri dan independen (sebagaimana kode tersebut digunakan jika tidak menggunakan PBO). Dengan modularitas, struktur dari sebuah program akan terkait dengan aspek-aspek dalam masalah yang akan diselesaikan melalui program tersebut. Cara seperti ini akan

menyederhanakan pemetaan dari masalah ke sebuah program ataupun sebaliknya.

3. Attributes

Atribut adalah data yang membedakan antara objek satu dengan yang lainnya. Contoh Objek : VolcanoRobot (a volcanic exploration vehicle), mempunyai atribut sebagai berikut:

- Status ~> exploring, moving, returning home
- Speed ~> in miles per hour
- Temperature ~> in Fahrenheit degrees

Dalam class, atribut sering disebut sebagai variabel. Atribut dibedakan menjadi dua jenis yaitu Instance Variable dan Class Variable. Instance variable adalah atribut untuk tiap objek dari kelas yang sama. Tiap objek mempunyai dan menyimpan nilai atributnya sendiri. Jadi, tiap objek dari class yang sama boleh mempunyai nilai yang sama atau berbeda. Class Variable adalah atribut untuk semua objek yang dibuat dari class yang sama. Semua objek mempunyai nilai atribut yang sama. Jadi semua objek dari class yang sama mempunyai hanya satu nilai yang value nya sama.

4. Behavior

Behavior/tingkah laku adalah hal-hal yang bisa dilakukan oleh objek dari suatu class. Behavior dapat digunakan untuk mengubah nilai atribut suatu objek, menerima informasi dari objek lain, dan mengirim informasi ke objek lain untuk melakukan suatu tugas (task).

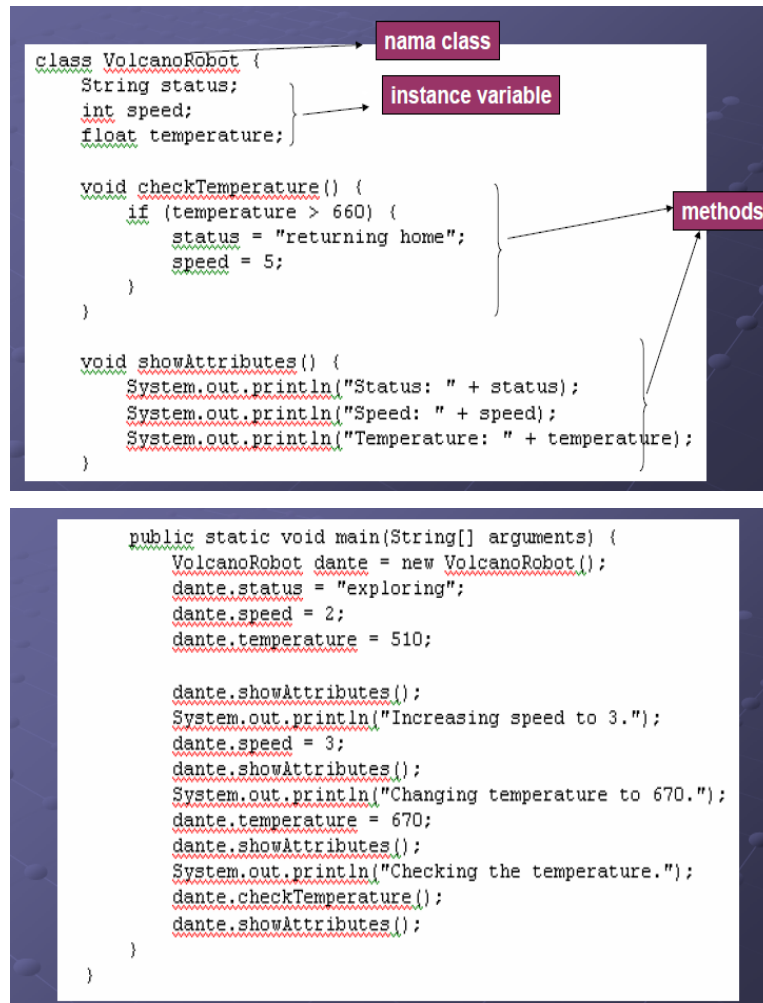
Contoh behavior untuk objek VolcanoRobot:

- check current temperature
- begin a survey
- report its current position

Dalam class, behavior disebut juga sebagai methods. Methods sendiri adalah serangkaian statements dalam suatu class yang handle suatu task tertentu. Cara objek berkomunikasi dengan objek yang lain adalah dengan menggunakan method.

- Contoh Object, class, attributes, dan behavior:

no mbr
gambar



5. Abstraksi

Abstraksi adalah kemampuan sebuah program untuk melewati aspek informasi yang diproses olehnya, yaitu kemampuan untuk memfokus pada inti. Setiap objek dalam sistem melayani sebagai model dari “pelaku” abstrak yang dapat melakukan kerja, laporan dan perubahan keadaannya, dan

berkomunikasi dengan objek lainnya dalam sistem, tanpa mengungkapkan bagaimana kelebihan ini diterapkan. Proses, fungsi atau metode dapat juga dibuat abstrak, dan beberapa teknik digunakan untuk mengembangkan sebuah pengabstrakan.



b. Konsep Konsep PBO:

1. Enkapsulasi (Encapsulation)

Enkapsulasi: Pembungkusan variabel dan method dalam sebuah obyek yang terlindungi serta menyediakan interface untuk mengakses variabel tersebut. Variabel dan method yang dimiliki oleh suatu objek, bisa ditentukan hak aksesnya. Dalam OOP, konsep enkapsulasi sebenarnya merupakan perluasan dari struktur dalam bahasa C.

Contoh: jam tangan. Dalam hal ini, penting sekali untuk mengetahui waktu, sedangkan cara jam mencatat waktu dengan baik antara jam bertenaga baterai atau bertenaga gerak tidaklah penting kita ketahui.

Dengan kata lain enkapsulasi berfungsi untuk memastikan pengguna sebuah objek tidak dapat mengganti keadaan dalam/dari sebuah objek dengan cara yang tidak layak; hanya metode dalam objek tersebut yang diberi izin untuk mengakses keadaannya. Setiap objek mengakses interface yang menyebutkan bagaimana objek lainnya dapat berinteraksi dengannya. Objek lainnya tidak akan mengetahui dan tergantung kepada representasi dalam objek tersebut.

2. Pewarisan (Inheritance)

Pewarisan merupakan pewarisan atribut dan method dari sebuah class ke class lainnya. Class yang mewarisi disebut

superclass dan Class yang diwarisi disebut subclass. Subclass bisa berlaku sebagai superclass bagi class lainnya, disebut sebagai multilevel inheritance.

Contoh : terdapat class sepeda dan sepeda gunung. Sepeda termasuk superclass. Sepeda gunung termasuk subclass. Hal ini dikarenakan sepeda gunung memiliki variabel dan method yang dimiliki oleh sepeda.

Prinsip dasar inheritance yaitu persamaan-persamaan yang dimiliki oleh beberapa kelas dapat digabungkan dalam sebuah class induk sehingga setiap kelas yang diturunkannya memuat hal-hal yang spesifik untuk kelas yang bersangkutan.

Contoh pewarisan:



❖ Keuntungan Pewarisan

- Subclass menyediakan state/behaviour yang spesifik yang membedakan dengan superclass, sehingga memungkinkan programmer untuk menggunakan ulang source code dari superclass yang telah ada.
- Programmer dapat mendefinisikan superclass khusus yang bersifat generik, yang disebut abstract class (abstraksi), untuk mendefinisikan class dengan tingkah laku dan state secara umum.

❖ Single & Multiple Inheritance

Bahasa C++ adalah contoh multiple inheritance. Suatu class diperbolehkan untuk mempunyai lebih dari satu superclass. Variabel dan method yang diwariskan merupakan kombinasi dari superclass-nya. Java adalah contoh single inheritance. Suatu class hanya boleh mempunyai satu superclass.

❖ **Multilevel Inheritance**

Suatu subclass bisa menjadi superclass bagi class yang lain.

- **Polomorfisme** melalui pengiriman pesan. Tidak bergantung kepada pemanggilan subrutin, bahasa orientasi objek dapat mengirim pesan; metode tertentu yang berhubungan dengan sebuah pengiriman pesan tergantung kepada objek tertentu di mana pesa tersebut dikirim. Contohnya, bila sebuah burung menerima pesan "gerak cepat", dia akan menggerakkan sayapnya dan terbang. Bila seekor singa menerima pesan yang sama, dia akan menggerakkan kakinya dan berlari. Keduanya menjawab sebuah pesan yang sama, namun yang sesuai dengan kemampuan hewan tersebut. Ini disebut polimorfisme karena sebuah variabel tunggal dalam program dapat memegang berbagai jenis objek yang berbeda selagi program berjalan, dan teks program yang sama dapat memanggil beberapa metode yang berbeda di saat yang berbeda dalam pemanggilan yang sama. Hal ini berlawanan dengan bahasa fungsional yang mencapai polimorfisme melalui penggunaan fungsi kelas-pertama.

Dengan menggunakan PBO maka dalam melakukan pemecahan suatu masalah kita tidak melihat bagaimana cara menyelesaikan suatu masalah tersebut (terstruktur) tetapi objek-objek apa yang dapat melakukan pemecahan masalah tersebut. Sebagai contoh anggap kita memiliki sebuah departemen yang memiliki manager, sekretaris,

petugas administrasi data dan lainnya. Misal manager tersebut ingin memperoleh data dari bag administrasi maka manager tersebut tidak harus mengambilnya langsung tetapi dapat menyuruh petugas bagian administrasi untuk mengambilnya. Pada kasus tersebut seorang manager tidak harus mengetahui bagaimana cara mengambil data tersebut tetapi manager bisa mendapatkan data tersebut melalui objek petugas administrasi. Jadi untuk menyelesaikan suatu masalah dengan kolaborasi antar objek-objek yang ada karena setiap objek memiliki deskripsi tugasnya sendiri.

Salah satu pemograman pendukung PBO adalah bahasa pemograman Java. Java adalah salah satu bahasa pemograman OOP. Bahasa ini awalnya dibuat oleh James Gosling. Bahasa ini banyak mengadopsi sintaksis yang terdapat pada C dan C++ namun dengan sintaksis model objek yang lebih sederhana serta dukungan rutin-rutin aras bawah yang minimal. Aplikasi-aplikasi berbasis java umumnya dikompilasi ke dalam p-code (*bytecode*) dan dapat dijalankan pada berbagai Mesin Virtual Java (JVM). Java merupakan bahasa pemrograman yang bersifat umum/non-spesifik (*general purpose*), dan secara khusus didisain untuk memanfaatkan dependensi implementasi seminimal mungkin. Karena fungsionalitasnya yang memungkinkan aplikasi java mampu berjalan di beberapa platform sistem operasi yang berbeda yang berbeda, java dikenal pula dengan slogannya, "*Tulis sekali, jalankan di mana pun*". Saat ini java merupakan bahasa pemrograman yang paling populer digunakan, dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi ataupun aplikasi berbasis web.

BAB II

PEMBAHASAN

2.1. Program pertama

```
#include <iostream>
#include <string>
using namespace std;
```

Pada program ini dengan menggunakan bahasa C++ yang bersifat yaitu case-sensitive, yang berarti bahwa huruf besar dan huruf kecil akan dianggap berbeda. Tidak seperti bahasa pemrograman Pascal, dimana penggunaan huruf besar dan kecil pada coding tidak akan mempengaruhi program. “Include” merupakan salah satu jenis pengarah preprocessor yang digunakan untuk membaca file.

Pada bagian program pertama ini digunakan dua header, yaitu `#include<iostream>` dan `#include<string>`.

Include merupakan salah satu jenis pengarah preprocessor yang digunakan untuk membaca file.

- `#include<iostream.h>` diperlukan pada program yang melibatkan objek `cout` dan `cin`.
- `#include<cstring>` digunakan untuk menampung dan memanipulasi data teks. Di dalam C++, `String` juga dapat dipergunakan dalam konstanta dan variabel.

Dan pada bagian `using namespace std;` adalah untuk memberitahukan kepada kompiler bahwa akan menggunakan semua fungsi, `class` atau `file` yang terdapat pada memori `namespace std`.

```
class cekBuah {
public:
    string Nama; string Buah;
```

```
cekBuah(string Nama, string Buah) {  
    cekBuah::Nama = Nama;  
    cekBuah::Buah = Buah;  
}
```

Pada bagian program di atas, yaitu **Class cekbuah{** yang diperlukan untuk membuat objek dan mendefinisikan variabel dan method-method yang ada pada program, Tanda ‘{‘ dan ‘}’ berfungsi untuk menandakan awal dan akhir sebuah definisi fungsi dan program, dan pada program ini juga menggunakan mode akses yaitu “public”. Dan pada mode akses public ini menggunakan tipe data string.

```
void tampilkanBuah() {  
    cout << "nama anda adalah : " << Nama << endl;  
    cout << "Buah favorit anda adalah : " << Buah << endl;  
}  
};
```

Dibagian member functions tampilInfo ini menggunakan **cout** yang berfungsi untuk menampilkan data ke dalam output (cetak pada layar). Tanda ‘<<’ merupakan operator yang fungsinya sebagai penghubung antara stream dengan kalimat inputan. Jadi, jika kita ingin menampilkan kalimat ke layar, kita perlu menghubungkan kalimat yang ingin ditampilkan dengan cout. Untuk itulah kita menggunakan operator << ini. Saat operator ini digunakan, maka kalimat akan dialirkan ke cout dan cout akan mencetaknya ke layar. Sedangkan **endl** merupakan manipulator yang digunakan untuk menyisipkan karakter mengatur pindah baris pada program. Tanda ‘;’ artinya menyatakan bahwa akhir dari sebuah pernyataan.

```
int main()  
{  
    string inputNama, inputBuah;  
    cout << "silahkan tulis nama anda   : "; getline(cin, inputNama);  
    cout << "masukkan buah favorit anda   : "; getline(cin, inputBuah);
```

```

        cout << endl << endl;

        cekBuah panggil = cekBuah(inputNama, inputBuah);

        panggil.tampilkanBuah();

        getchar();

        return 0;

    }

```

Pada bagian program **Int main()** merupakan bagian utama dari program. Pada bagian inilah nilai variabel objek-objek (cek buah) pada class diinputkan.

Getch digunakan untuk membaca karakter dengan sifat/jenis karakter yang dimasukkan tanpa perlu diakhiri dengan menekan tombol enter. Karakter akan langsung ditampilkan pada layar. **Return** digunakan karena pada main program menggunakan tipe data integer. Tanda ‘**}**’ menandakan berakhirnya program.

Program yang dijalankan :

Input :

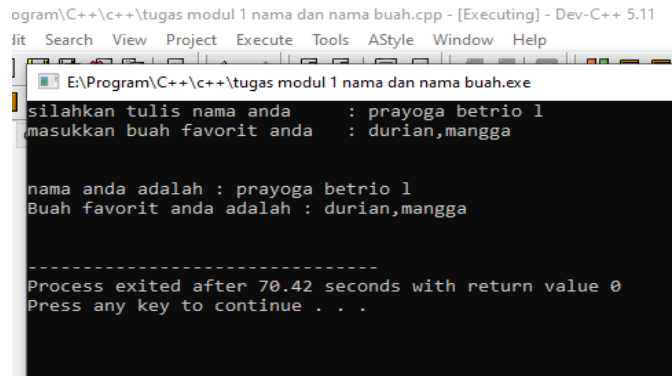
```

as modul 1 nama dan nama buah.cpp - Dev-C++ 5.11
Project  Execute  Tools  AStyle  Window  Help
[+] tugas modul 1 nama dan nama buah.cpp
1  #include <iostream>
2  #include <string>
3  using namespace std;
4
5  class cekBuah {
6  public:
7      string Nama; string Buah;
8
9      cekBuah(string Nama, string Buah) {
10         cekBuah::Nama = Nama;
11         cekBuah::Buah = Buah;
12     }
13
14     void tampilkanBuah() {
15         cout << "nama anda adalah : " << Nama << endl;
16         cout << "Buah favorit anda adalah : " << Buah << endl;
17     }
18 }
19
20 };
21
22 int main()
23 {
24     string inputNama, inputBuah;
25     cout << "silahkan tulis nama anda : "; getline(cin, inputNama);
26     cout << "masukkan buah favorit anda : "; getline(cin, inputBuah);
27     cout << endl << endl;
28     cekBuah panggil = cekBuah(inputNama, inputBuah);
29     panggil.tampilkanBuah();
30
31     getchar();
32     return 0;
33 }

```

nomor gambar

Output :



```
ogram\C++\c++\tugas modul 1 nama dan nama buah.cpp - [Executing] - Dev-C++ 5.11
File Edit View Project Execute Tools AStyle Window Help

E:\Program\C++\c++\tugas modul 1 nama dan nama buah.exe

silahkan tulis nama anda : prayoga betrio l
masukkan buah favorit anda : durian,mangga

nama anda adalah : prayoga betrio l
Buah favorit anda adalah : durian,mangga

-----
Process exited after 70.42 seconds with return value 0
Press any key to continue . . .
```

2.2. Program kedua

```
#include <iostream>
#include <string>
#include <conio.h>
using namespace std;
```

Pada program ini dengan menggunakan bahasa C++ yang bersifat yaitu case-sensitive, yang berarti bahwa huruf besar dan huruf kecil akan dianggap berbeda. Tidak seperti bahasa pemrograman Pascal, dimana penggunaan huruf besar dan kecil pada coding tidak akan mempengaruhi program. “Include” merupakan salah satu jenis pengarah preprocessor yang digunakan untuk membaca file.

Pada bagian program pertama ini digunakan tiga header, yaitu `#include<iostream>`, `#include<string>`, dan `#include <conio h>`.

Include merupakan salah satu jenis pengarah preprocessor yang digunakan untuk membaca file.

- `#include<iostream.h>` diperlukan pada program yang melibatkan objek `cout` dan `cin`.
- `#include<string>` digunakan untuk menampung dan memanipulasi data teks. Di dalam C++, `String` juga dapat dipergunakan dalam konstanta dan variabel.

- c. `#include<conio.h>` diperlukan untuk program yang melibatkan `clrscr()`, yaitu perintah untuk membersihkan layar dan fungsi `getch()` untuk menerima sembarang input keyboard dari user pada program.

Dan pada bagian `using namespace std;` adalah untuk memberitahukan kepada kompiler bahwa akan menggunakan semua fungsi, *class* atau *file* yang terdapat pada memori *namespace std*.

```
class dataBuah
{
public:
    string nama;
    string buah;
    string rasa;
```

Pada bagian program di atas, yaitu ***Class databuah{*** yang diperlukan untuk membuat objek dan mendefinisikan variabel dan method-method yang ada pada program, Tanda '{' dan '}' berfungsi untuk menandakan awal dan akhir sebuah definisi fungsi dan program, dan pada program ini juga menggunakan mode akses yaitu "public". Dan pada mode akses public ini menggunakan tipe data string.

```
dataBuah(string inama, string ibuah, string irasa)
{
    nama = inama;
    buah = ibuah;
    rasa = irasa;
    cout << "Buah yang dipilih\t : " << dataBuah::nama << endl;
    cout << "Warna Buah\t      : " << dataBuah::buah << endl;
    cout << "rasanya\t          : " << dataBuah::rasa << endl;
    cout << " " << endl;
```

```
    }  
};
```

Dibagian member functions `tampilInfo` ini menggunakan **cout** yang berfungsi untuk menampilkan data ke dalam output (cetak pada layar). Tanda `<<` merupakan operator yang fungsinya sebagai penghubung antara stream dengan kalimat inputan. Jadi, jika kita ingin menampilkan kalimat ke layar, kita perlu menghubungkan kalimat yang ingin ditampilkan dengan `cout`. Untuk itulah kita menggunakan operator `<<` ini. Saat operator ini digunakan, maka kalimat akan dialirkan ke `cout` dan `cout` akan mencetaknya ke layar. Sedangkan **endl** merupakan manipulator yang digunakan untuk menyisipkan karakter mengatur pindah baris pada program. Tanda `;` artinya menyatakan bahwa akhir dari sebuah pernyataan.

```
int main()  
{  
    dataBuah buah1("jeruk", "Kuning", "manis"),  
                buah2("jambu", "hijau", "manis"),  
                buah3("Anggur", "Ungu", "manis");  
    getch();  
    return 0;  
}
```

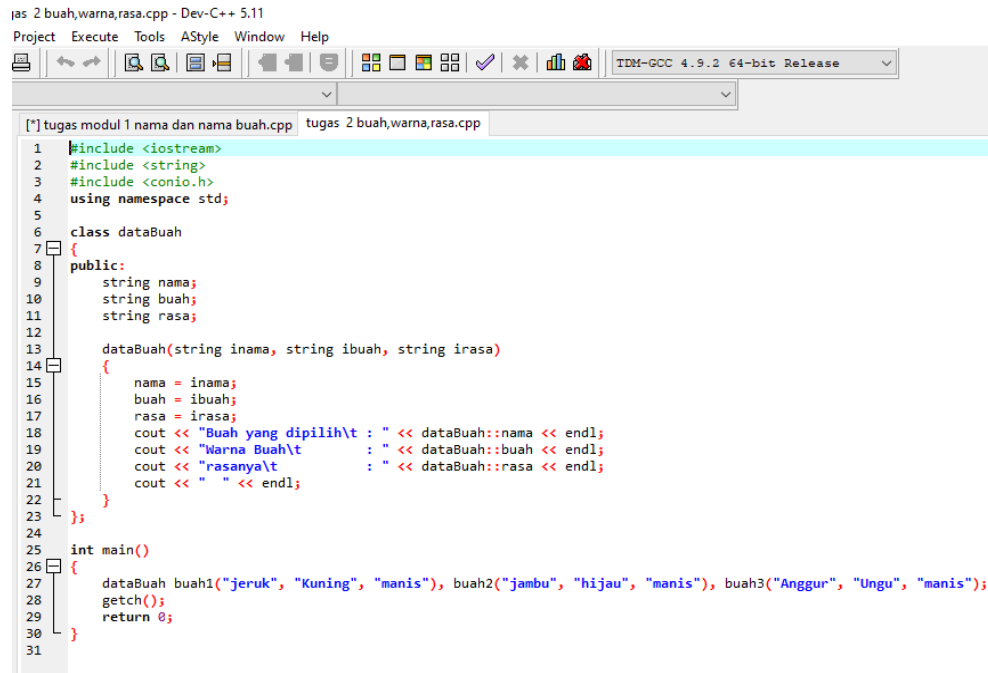
Pada bagian program **Int main()** merupakan bagian utama dari program. Pada bagian inilah nilai variabel objek-objek (buah 1, buah 2, buah 3.) pada class diinputkan.

Getch digunakan untuk membaca karakter dengan sifat/jenis karakter yang dimasukkan tanpa perlu diakhiri dengan menekan tombol enter. Karakter akan langsung ditampilkan pada layar.

Return digunakan karena pada main program menggunakan tipe data integer. Tanda '}' menandakan berakhirnya program.

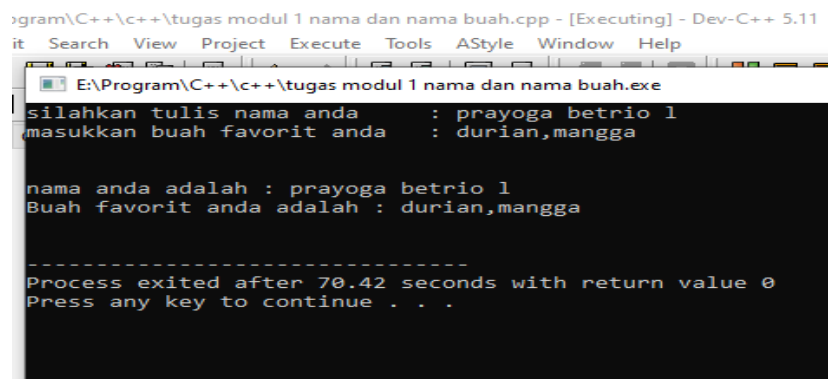
Program yang dijalankan :

Input :



```
1 #include <iostream>
2 #include <string>
3 #include <conio.h>
4 using namespace std;
5
6 class dataBuah
7 {
8 public:
9     string nama;
10    string buah;
11    string rasa;
12
13    dataBuah(string inama, string ibuah, string irasa)
14    {
15        nama = inama;
16        buah = ibuah;
17        rasa = irasa;
18        cout << "Buah yang dipilih\t : " << dataBuah::nama << endl;
19        cout << "Warna Buah\t\t\t : " << dataBuah::buah << endl;
20        cout << "rasanya\t\t\t\t\t : " << dataBuah::rasa << endl;
21        cout << " " << endl;
22    }
23 };
24
25 int main()
26 {
27     dataBuah buah1("jeruk", "Kuning", "manis"), buah2("jambu", "hijau", "manis"), buah3("Anggur", "Ungu", "manis");
28     getch();
29     return 0;
30 }
31
```

Output :



```
Program\C++\c++\tugas modul 1 nama dan nama buah.cpp - [Executing] - Dev-C++ 5.11
File Edit View Project Execute Tools AStyle Window Help

E:\Program\C++\c++\tugas modul 1 nama dan nama buah.exe

silahkan tulis nama anda      : prayoga betrio 1
masukkan buah favorit anda    : durian,mangga

nama anda adalah : prayoga betrio 1
Buah favorit anda adalah : durian,mangga

-----
Process exited after 70.42 seconds with return value 0
Press any key to continue . . .
```

BAB III

KASIMPULAN

Dapat di tarik kesimpulan sebagai berikut, Pemrograman Berorientasi Objek (PBO) merupakan pemrograman yang berorientasikan terhadap objek, dimana semua data dan fungsi yang dibungkus ke dalam kelas-kelas atau objek-objek. dan enkapsulasi adalah Pembungkusan variabel dan method dalam sebuah obyek yang terlindungi serta menyediakan interface untuk mengakses variabel tersebut. Variabel dan method yang dimiliki oleh suatu objek, bisa ditentukan hak aksesnya.

Objek berfungsi untuk membungkus data dan fungsi bersama menjadi satu unit dalam sebuah program komputer. Objek merupakan dasar dari modularitas dan struktur dalam sebuah program komputer berorientasi objek.

Class merupakan prototipe atau blue prints yang mendefinisikan variabel-variabel dan method-method secara umum.

DAFTAR PUSTAKA

Tim Dosen Algoritma Pemrograman II. 2020. *Modul Praktikum Algoritma Pemrograman II*. Universitas Palangka Raya : UPR. Fakultas Teknik.

Auliarosalinda. 2013. *Konsep Dasar Pemrograman Berorientasi*.
<http://blogspot.com/3013/03/konsep-dasar-pemrograman-berorientasi.html>
Diakses 05 April 2020 jam 19.07 WIB.

Adityarizki. 2012. *Konsep Dasar Pemrograman Berorientasi Objek*.
<http://www.net/2012/6/konsep-dasar-pemrograman-berorientasi-objek/>
Diakses 05 April 2020 jam 20.30 WIB.

Urutan sesuai abjad

-

isi lampiran Skema gambar yg ada di laporan

LAMPIRAN

Program pertama yang dijalankan :

Input :

```
as modul 1 nama dan nama buah.cpp - Dev-C++ 5.11
Project Execute Tools AStyle Window Help

1  #include <iostream>
2  #include <string>
3  using namespace std;
4
5  class cekBuah {
6  public:
7      string Nama; string Buah;
8
9      cekBuah(string Nama, string Buah) {
10         cekBuah::Nama = Nama;
11         cekBuah::Buah = Buah;
12     }
13
14     void tampilkanBuah() {
15         cout << "nama anda adalah : " << Nama << endl;
16         cout << "Buah favorit anda adalah : " << Buah << endl;
17     }
18
19 };
20
21 int main()
22 {
23     string inputNama, inputBuah;
24     cout << "silahkan tulis nama anda : "; getline(cin, inputNama);
25     cout << "masukkan buah favorit anda : "; getline(cin, inputBuah);
26     cout << endl << endl;
27     cekBuah panggil = cekBuah(inputNama, inputBuah);
28     panggil.tampilkanBuah();
29
30     getchar();
31     return 0;
32 }
33
```

Output :

```
ogram\C++\c++\tugas modul 1 nama dan nama buah.cpp - [Executing] - Dev-C++ 5.11
File Edit View Project Execute Tools AStyle Window Help

E:\Program\C++\c++\tugas modul 1 nama dan nama buah.exe
silahkan tulis nama anda : prayoga betrio l
masukkan buah favorit anda : durian,mangga

nama anda adalah : prayoga betrio l
Buah favorit anda adalah : durian,mangga

-----
Process exited after 70.42 seconds with return value 0
Press any key to continue . . .
```

Program kedua yang dijalankan :

Input :

```
jas 2 buah,warna,rasa.cpp - Dev-C++ 5.11
Project Execute Tools AStyle Window Help

[*] tugas modul 1 nama dan nama buah.cpp | tugas 2 buah,warna,rasa.cpp

1  #include <iostream>
2  #include <string>
3  #include <conio.h>
4  using namespace std;
5
6  class dataBuah
7  {
8  public:
9      string nama;
10     string buah;
11     string rasa;
12
13     dataBuah(string inama, string ibuah, string irasa)
14     {
15         nama = inama;
16         buah = ibuah;
17         rasa = irasa;
18         cout << "Buah yang dipilih\t : " << dataBuah::nama << endl;
19         cout << "Warna Buah\t\t\t : " << dataBuah::buah << endl;
20         cout << "rasanya\t\t\t\t\t : " << dataBuah::rasa << endl;
21         cout << " " << endl;
22     }
23 };
24
25 int main()
26 {
27     dataBuah buah1("jeruk", "Kuning", "manis"), buah2("jambu", "hijau", "manis"), buah3("Anggur", "Ungu", "manis");
28     getch();
29     return 0;
30 }
31
```

Output :

```
rogram\C++\c++\tugas modul 1 nama dan nama buah.cpp - [Executing] - Dev-C++ 5.11
it Search View Project Execute Tools AStyle Window Help

E:\Program\C++\c++\tugas modul 1 nama dan nama buah.exe

silahkan tulis nama anda      : prayoga betrio l
masukkan buah favorit anda    : durian,mangga

nama anda adalah : prayoga betrio l
Buah favorit anda adalah : durian,mangga

-----
Process exited after 70.42 seconds with return value 0
Press any key to continue . . .
```

