

**LAPORAN HASIL PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN II**



NAMA : M. Ade Shofy
NIM : 193010503006
KELAS : A
MODUL : I (DASAR PBO)

**JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS PALANGKA RAYA
2020**

**LAPORAN HASIL PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN II**



Nama : M. Ade Shofy
NIM : 193010503006
Kelas : A
Modul : DASAR PEMROGRAMAN
BERORIENTASI OBJEK

Komposisi	MAX	Nilai
BAB I Tujuan dan Landasan Teori	10	7
BAB II Pembahasan	60	48
BAB III Kesimpulan	20	13
Daftar Pustaka	5	3
Lampiran	5	3
Jumlah	100	74

Penilai

Asisten Praktikum

Diana

BAB I

TUJUAN DAN LANDASAN TEORI

I. TUJUAN

Setelah menyelesaikan modul ini, mahasiswa diharapkan mampu :

1. Memahami dasar-dasar pemrograman berorientasi objek
2. memahami enkapsulasi
3. membuat kelas dan objek

II. DASAR TEORI

Pemrograman berorientasi objek (object-oriented programming disingkat OOP) 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 OOP mengklaim bahwa OOP lebih mudah dipelajari bagi pemula dibanding dengan pendekatan sebelumnya, dan pendekatan OOP lebih mudah dikembangkan dan dirawat.

Istilah-istilah dalam OOP:

1. Kelas atau 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 OOP). 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.

2.Objek

Membungkus data dan fungsi bersama menjadi suatu unit dalam sebuah program komputer, objek merupakan dasar dari modularitas dan struktur dalam sebuah program komputer berorientasi objek.

3.Abstraksi

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.

4.Enkapsulasi

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.

5.Polimorfisme 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 OOP 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.

BAB II

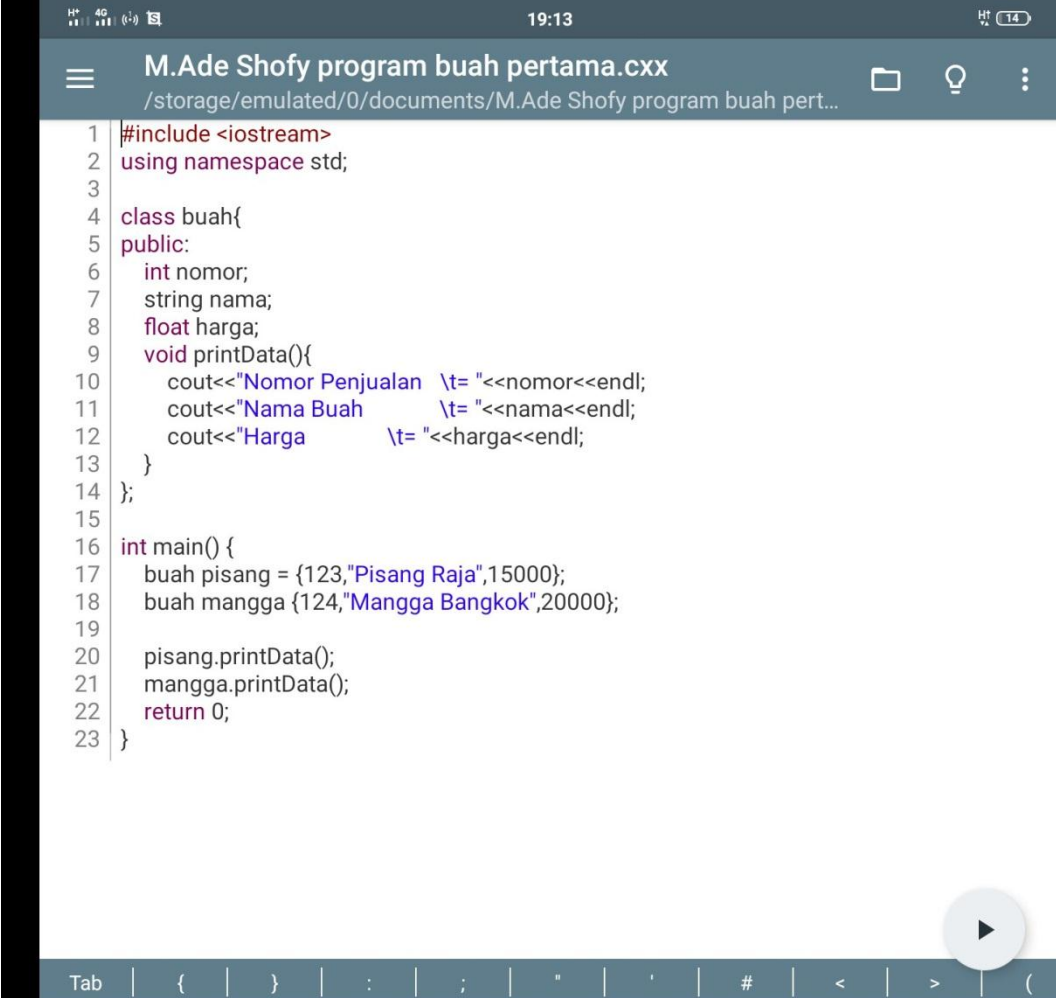
PEMBAHASAN

I. LANGKAH KERJA

Buatlah 2 program dengan tema buah bagi yang nomor ganjil dan jika nomor genap maka tema hewan dengan menggunakan prinsip DASAR PEMROGRAMAN BERORIENTASI OBJEK.

II. PEMBAHASAN

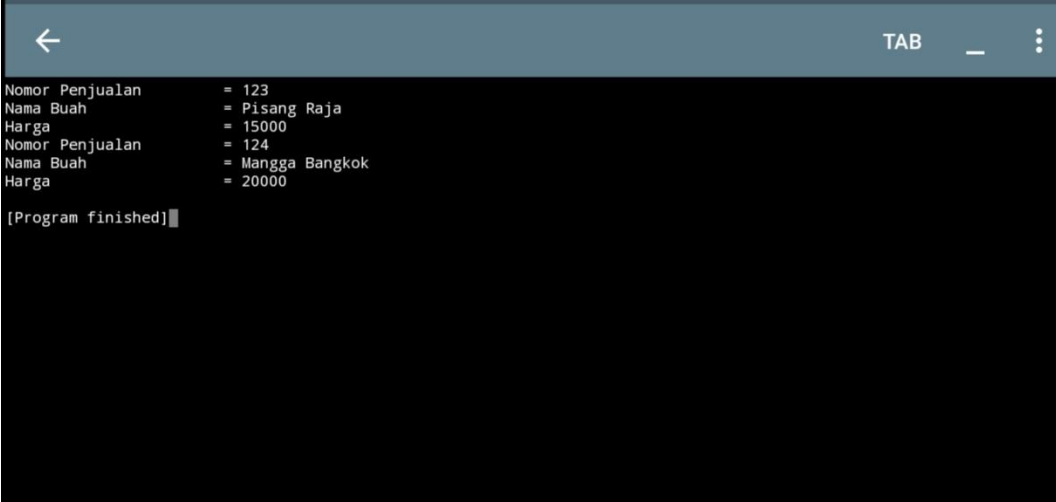
1. Program buah pertama.



```
1 #include <iostream>
2 using namespace std;
3
4 class buah{
5 public:
6     int nomor;
7     string nama;
8     float harga;
9     void printData(){
10         cout<<"Nomor Penjualan \t= "<<nomor<<endl;
11         cout<<"Nama Buah \t= "<<nama<<endl;
12         cout<<"Harga \t= "<<harga<<endl;
13     }
14 };
15
16 int main() {
17     buah pisang = {123,"Pisang Raja",15000};
18     buah mangga {124,"Mangga Bangkok",20000};
19
20     pisang.printData();
21     mangga.printData();
22     return 0;
23 }
```

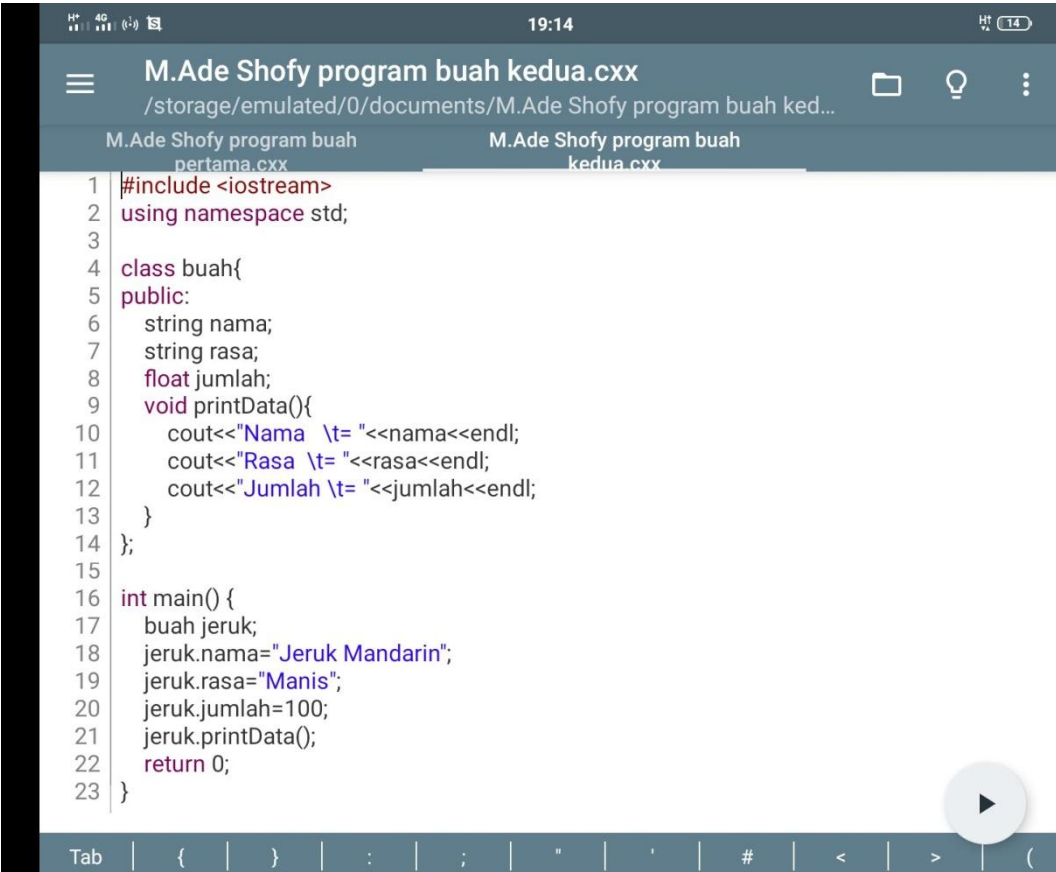
nomor gambar

Hasil output.



```
← TAB ⋮
Nomor Penjualan    = 123
Nama Buah          = Pisang Raja
Harga              = 15000
Nomor Penjualan    = 124
Nama Buah          = Mangga Bangkok
Harga              = 20000
[Program finished]
```

2.Program buah kedua.



```
19:14
M.Ade Shofy program buah kedua.cxx
/storage/emulated/0/documents/M.Ade Shofy program buah ked...
M.Ade Shofy program buah pertama.cxx M.Ade Shofy program buah kedua.cxx

1 #include <iostream>
2 using namespace std;
3
4 class buah{
5 public:
6     string nama;
7     string rasa;
8     float jumlah;
9     void printData(){
10         cout<<"Nama \t= "<<nama<<endl;
11         cout<<"Rasa \t= "<<rasa<<endl;
12         cout<<"Jumlah \t= "<<jumlah<<endl;
13     }
14 };
15
16 int main() {
17     buah jeruk;
18     jeruk.nama="Jeruk Mandarin";
19     jeruk.rasa="Manis";
20     jeruk.jumlah=100;
21     jeruk.printData();
22     return 0;
23 }
```

Hasil output.

A screenshot of a terminal window with a dark background. The title bar at the top is light blue and contains a back arrow icon on the left, the word 'TAB' in the center, and a minus sign icon on the right. The terminal text is white and shows the output of a program: 'Nama = Jeruk Mandarin', 'Rasa = Manis', 'Jumlah = 100', and '[Program finished]' followed by a cursor.

```

Nama = Jeruk Mandarin
Rasa = Manis
Jumlah = 100

[Program finished]

```

Pembahasan :

Program buah pertama : Pada program pertama saya menggunakan class buah dan tipe data yang saya gunakan adalah integer,string,float, dan fungsi assign yang merupakan fungsi bernilai void dan semuanya bersifat public. Artinya dapat diakses diluar class. Setelah itu saya memasukan “nomor penjualan”, “nama buah”, dan “harga”. Lalu memasukan fungsi “main” yang merupakan fungsi utama diluar class. Lalu setelah itu memasukan nomor penjualan buah, nama buah, serta harga buah yang telah ditentukan. Dan yang terakhir fungsi print data agar menampilkan semua data yang telah dimasukan.

Program buah kedua : Pada program kedua ini saya juga menggunakan class dan tipe data yang digunakan string,float,dan void yang juga bersifat public. Yang artinya semuanya dapat diakses diluar class. Lalu memasukan “nama”, “rasa”, dan “jumlah”. Setelah itu memasukan fungsi “main” yang merupakan fungsi utama diluar class. Dan setelah itu memasukan nama buah, rasa buah tersebut dan jumlah buah yang tersedia yang telah saya tentukan sendiri. Lalu yang terakhir menambahkan fungsi print data agar menampilkan semua data yang sudah saya masukan sebelumnya. Sebenarnya program kedua ini tidak jauh beda dengan program yang pertama.

Paragraf

BAB III KESIMPULAN

- Pemrograman berorientasi objek (object-oriented programming disingkat OOP) merupakan pemograman yang berorientasikan kepada objek.
- Semua data dan fungsi di dalam paradigma dibungkus dalam kelas-kelas atau objek-objek. Setiap objek dapat menerima pesan, memproses data, dan mengirim pesan ke objek lainnya.
- Class atau kelas merupakan kumpulan atas definisi data dan fungsi-fungsi dalam suatu unit untuk suatu tujuan tertentu.
- Fungsi objek adalah untuk membungkus data dan fungsi bersama menjadi suatu unit dalam sebuah program komputer, objek merupakan dasar dari modularitas dan struktur dalam sebuah program komputer berorientasi objek.
- Kemampuan sebuah program untuk melewati aspek informasi yang diproses olehnya, yaitu kemampuan untuk memfokus pada inti disebut abstraksi.

DAFTAR PUSTAKA

<http://technopark.surakarta.go.id/id/media-publik/komputer-teknologi-informasi/189-konsep-dasar-pemrograman-berorientasi-objek> (Diakses pada tanggal 7 April 2020).

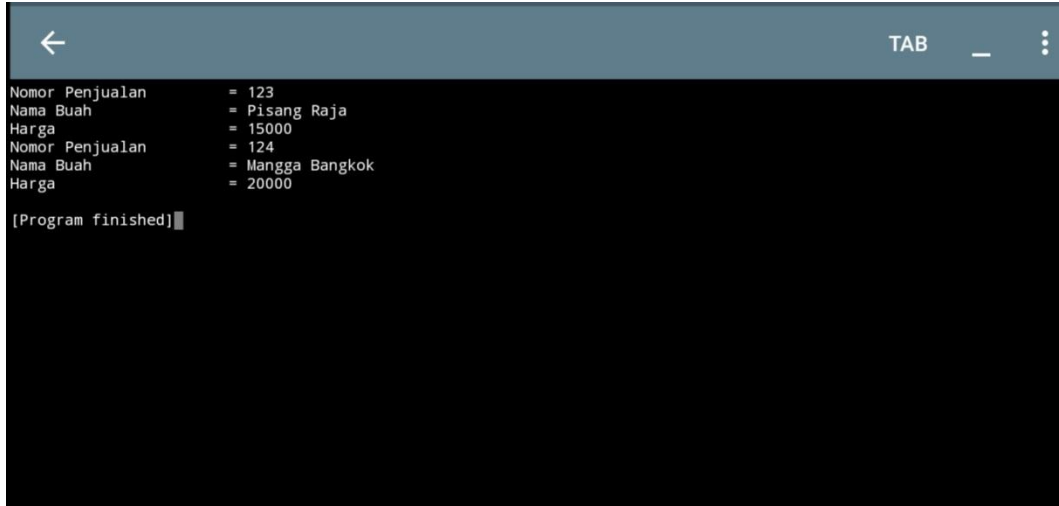
<http://kodedasar.com/class-cpp/&ved=2ahUKEwiM> (Diakses pada tanggal 7 April 2020).

<https://www.belajarcpp.com/tutorial/cpp/class/&ved=2ahUKEwiM> (Diakses pada tanggal 7 April 2020).

Akuran Permisar

LAMPIRAN

Hasil output program pertama.



```
← TAB ⋮
Nomor Penjualan = 123
Nama Buah      = Pisang Raja
Harga          = 15000
Nomor Penjualan = 124
Nama Buah      = Mangga Bangkok
Harga          = 20000
[Program finished]
```

gambar . . .

Hasil output program kedua.



```
← TAB ⋮
Nama    = Jeruk Mandarin
Rasa    = Manis
Jumlah  = 100
[Program finished]
```