

**LAPORAN HASIL PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN II**



NAMA : JORGI JACKO EXCEL
NIM : 193030503064
KELAS : A
**MODUL : PEMROGRAMAN BERORIENTASI
OBJEK**

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

**LAPORAN HASIL PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN II**



Nama : Jorgi Jacko Excel
NIM : 193030503064
Kelas : A
Modul : PEMROGRAMAN BERORIENTASI
OBJEK

Komposisi	MAX	Nilai
BAB I Tujuan dan Landasan Teori	10	8
BAB II Pembahasan	60	50
BAB III Kesimpulan	20	15
Daftar Pustaka	5	3
Lampiran	5	4
Jumlah	100	80

Penilai
Asisten Praktikum

Diana

BAB I

TUJUAN DAN LANDASAN TEORI

1. TUJUAN

Setelah menyelesaikan modul ini, mahasiswa diharapkan mampu:

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

2. LANDASAN TEORI

Pemrograman berorientasi objek (Inggris: **object-oriented programming** disingkat **OOP**) merupakan paradigma pemrograman berdasarkan konsep "objek", yang dapat berisi data, dalam bentuk field atau dikenal juga sebagai atribut; serta kode, dalam bentuk fungsi/prosedur atau dikenal juga sebagai method. Semua data dan fungsi di dalam paradigma ini dibungkus dalam kelas-kelas atau objek-objek. Bandingkan dengan logika pemrograman terstruktur. 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 peranti 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.

Berikut merupakan konsep-konsep yang terdapat di dalam ruang lingkup Pemrograman Berorientasi Obyek :



*Nomor
gambar*

a) Kelas

Class adalah salah satu dari konsep OOP yang digunakan untuk membungkus data abstraksi procedural sebagai deskripsi tergeneralisir atau rancangan dari sebuah object untuk mendefinisikan atau menggambarkan isi dan tingkah laku sebagai entitas dari object.

b) Objek

Objek adalah 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.

c) 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.

d) Enkapsulasi

Enkapsulasi adalah 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.

e) Polimorfisme

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.

f) Inheritas (Pewarisan)

Inheritas ini merupakan konsep yang memiliki fungsi mengatur polimorfisme dan enkapsulasi dengan mengizinkan objek didefinisikan dan diciptakan dengan jenis yang khusus dari objek yang sudah ada. Objek-objek ini dapat membagi dan memperluas perilaku mereka tanpa melakukan implementasi perilaku tersebut.

Class dan Object

Class dan Object merupakan fitur yang sangat membantu untuk mendirikan sebuah program besar, menjadikan sebuah code program yang ditulis oleh programmer mudah untuk dimengerti, lebih terstruktur, dan juga mudah dalam pemeliharaan program.

Terdapat banyak fasilitas yang disediakan oleh class, yaitu dapat menampung member variabel, function, constructor, destructor, overloading dan lain-lain. Diluar definisi class kita juga dimungkinkan untuk membuat relasi seperti inheritance dan overriding.

Dalam membuat member function atau method terdapat dua cara, yaitu:

- a. Pertama, mendirikan dan mendefinisi function di dalam definisi class. Seperti apa yang sudah penulis contohkan pada contoh-contoh program di atas.
- b. Kedua, Mendirikan di dalam definisi class dan mendefinisikan function di luar class. Mirip seperti kita membuat function dengan mendirikan function prototype terlebih dahulu dan mendefinisikan function tersebut sesudah deklarasi function.

Pada class kita juga dimungkinkan untuk mendirikan function prototype, dan mendefinisikanya di luar definisi class. Untuk memberi definisi pada function tersebut kita dapat melakukannya dengan menggunakan “Scope Resolution Operator ::”, yang diletakan di antara nama class dan nama member function.

Untuk mendefinisiakan function menggunakan Scope Resolution Operator, kita bebas menaruh definisi tersebut dimana pun, seperti tepat di bawah deklarasi class tersebut atau di bawah main function dan juga bisa di definisikan diluar file, tapi kita tidak bisa melakukan definisi sebelum deklarasi function.

Access Modifier

Access Modifier (kadang juga disebut Access Specifier) adalah salah satu fitur penting dalam Object Oriented Programming (OOP) untuk melakukan Data Hiding (Menyembunyikan Data). Fitur ini memungkinkan kita untuk mengatur hak akses dari member class, digunakan agar tidak sembarangan perintah dapat mengakses, atau tidak bisa di akses secara langsung.

Fitur ini memiliki 3 tipe Access Modifier, yaitu:

- a. Public

Public adalah label yang berfungsi untuk menentukan sifat akses ke semua member yang mengikutinya (di bawahnya), sehingga memiliki sifat

dapat di akses dari manapun. Dapat di akses dari dalam class itu sendiri, dari anak class (derived class) dan juga dari luar class.

b. Private

Private adalah label yang berfungsi untuk menentukan sifat akses ke semua member yang mengikutinya menjadi memiliki sifat yang tidak dapat di akses dari manapun kecuali melalui friend function dan dari dalam class itu sendiri.

c. Protected

Protected adalah label yang berfungsi untuk menentukan sifat akses semua member yang mengikutinya, sehingga memiliki sifat yang tidak dapat diakses dari luar class tapi masih dapat di akses dari dalam class maupun anak class (derived class).

Ketiga Access Modifier mempunyai sifat mereka masing-masing . Semua baris deklarasi sebagai member class akan mengikuti sifat dari label di atasnya hingga label berikutnya atau tanda penutup class }; .Dan sifat access default adalah private, jadi jika tidak di dirikan sebuah access modifier di atasnya maka otomatis member akan memiliki tipe sifat private.

Constructor

Constructor adalah spesial member function yang akan dieksekusi terlebih dahulu saat pembuatan instance (object). Constructor biasanya digunakan untuk insialisasi member-member atau melakukan persiapan lainnya.

Dalam mendirikan Constructor mirip seperti kita mendirikan function hanya saja constructor harus menjadi member dari class, harus memiliki identitas sama dengan class tersebut dan tidak memiliki return type.

Karakteristik Constructor:

- a. Constructor memiliki identitas sama seperti identitas dari class.
- b. Constructor tidak memiliki return type.
- c. Constructor harus berada dalam label public.
- d. Constructor tidak bisa dipanggil dari luar class.

Constructor dapat didirikan tanpa menggunakan parameter atau menggunakan parameter. Constructor yang tidak mempunyai parameter disebut sebagai Default Constructor

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 bag 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.

Bahasa pemrograman yang mendukung OOP yaitu Visual Foxpro, Java, C++, Pascal (Bahasa pemrograman), SIMULA, Smalltalk, Ruby, Python, PHP, C#, Delphi, Eiffel, Perl, Adobe Flash 3.0.

BAB II

PEMBAHASAN

1. Program 1

Berikut adalah pembahasan program 1 yang diperintahkan membuat class buah Jeruk.

Source Code:

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

class jeruk
{
public:
jeruk(string, string, string);
string jenis;
string rasa;
string berat;
};
jeruk::jeruk(string inputjenis, string inputrasa, string inputberat){
jeruk::jenis = inputjenis;
jeruk::rasa = inputrasa;
jeruk::berat = inputberat;
}
int main()
{
jeruk jeruk1( "Jeruk Mandarin", "Manis", "1 Kg" );

cout<<"\nBuah Jeruk"<<endl;
cout<<"\nJenis Jeruk: "<<jeruk1.jenis<<endl;
cout<<"Rasa Jeruk: "<<jeruk1.rasa<<endl;
cout<<"Berat Jeruk: "<<jeruk1.berat<<endl;

return 0;
}
```

Pada baris 1 dan 2 disebut dengan header. Diawali dengan tanda # merupakan pernyataan untuk menyertakan preprocessor. Pernyataan ini bukan untuk dieksekusi. Lalu diikuti dengan kata “include” setelah tanda # tadi. include sendiri merupakan salah satu jenis pengarah praprosesor yang dipakai untuk membaca file-file header itu sendiri.

`include <iostream.h>`: diperlukan pada program yang melibatkan objek `cout` dan `cin`. `Cin` merupakan fungsi masukan (digunakan untuk menyimpan data dalam suatu variabel). Bentuk umum: `cin>>var x;`. `Cout` merupakan fungsi keluaran (digunakan untuk menampilkan data ataupun tulisan). Bentuk umum: `cout<<"tulisan";` atau `cout<<var x;`

Dan baris 3 terdapat **using namespace std** yang berfungsi untuk memanggil class/objek/fungsi yang terdapat dalam namespace `std`. Misalnya memanggil `cout` dan `cin`.

Class Jeruk merupakan class yang akan digunakan pada program ini, atau bisa dikatakan sebagai main class program. Penggunaan `Public` agar program di atas dapat di compile, perintah `public` menyatakan bahwa perintah-perintah yang ada di bawahnya dapat diakses diluar class. Perintah `public` termasuk access specifier (penentu akses). `Public` berisi fungsi yang mengakses status internal sebuah objek atau dapat dikatakan merupakan method atau perilaku dari sebuah objek.

Jeruk : : Jeruk(string inputjenis, string inputrasa, string inputberat) merupakan konstruktor adalah fungsi anggota dari suatu kelas yang secara otomatis dijalankan ketika suatu objek diciptakan. Konstruktor mempunyai aturan yaitu nama konstruktor harus sama dengan nama kelas yang mendefinisikannya. Member functions didefinisikan diluar kelas dengan menggunakan binary scope resolution operator (`::`) yang berfungsi untuk mrngikat nama fungsi ke nama kelas dan mengidentifikasi fungsi dari suatu kelas tertentu. Konstruktor biasanya digunakan untuk melakukan alokasi memori, memberikan nilai awal terhadap anggota kelas (atau disebut inisialisasi), dan melakukan tugas umum lainnya.

Dengan menggunakan konstruktor kita bisa sedikit 'berhemat' dengan tidak membuat suatu fungsi yang secara khusus diciptakan untuk melakukan suatu inisialisasi, namun cukup dengan menggunakan suatu konstruktor. Tanda `<<` merupakan karakter yang harus mengikuti syntax `cout`. Jadi dimana `cout` ditulis setelahnya kita wajib menuliskan karakter `<<`. Dan perintah `endl;` berfungsi ganti baris. `Int main()` suatu program dalam C++ dapat berisi banyak fungsi, fungsi adalah modul yang berisi kode-kode untuk menyelesaikan masalah-masalah tertentu. Dan diakhir listing program `return 0;` untuk menunjukkan program telah selesai dan berhasil.

2. Program 1

Berikut adalah pembahasan program 2 seperti diatas yang diperintahkan membuat class buah Semangka.

Source Code:

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

class semangka
{
public:
    semangka(string, string, string);
    string jenis;
    string rasa;
    string berat;
};

semangka::semangka(string inputjenis, string inputrasa, string
inputberat){
    semangka::jenis = inputjenis;
    semangka::rasa = inputrasa;
    semangka::berat = inputberat;
}

int main()
{
    semangka semangka1( "Semangka Merah", "Manis", "1.5 Kg" );

    cout<<"\nBuah Semangka"<<endl;
    cout<<"\nJenis semangka: "<<semangka1.jenis<<endl;
    cout<<"Rasa semangka: "<<semangka1.rasa<<endl;
    cout<<"Berat semangka: "<<semangka1.berat<<endl;

    return 0;
}
```

Pada baris 1 dan 2 disebut dengan header. Diawali dengan tanda # merupakan pernyataan untuk menyertakan preprocessor. Pernyataan ini bukan untuk dieksekusi. Lalu diikuti dengan kata “include” setelah tanda # tadi. include sendiri merupakan salah satu jenis pengarah praprosesor yang dipakai untuk membaca file-file header itu sendiri.

include <iostream.h>: diperlukan pada program yang melibatkan objek cout dan cin. Cin merupakan fungsi masukan(digunakan untuk menyimpan data dalam suatu variabel). Bentuk umum: cin>>var x;. Cout merupakan fungsi keluaran(digunakan untuk menampilkan data ataupun tulisan). Bentuk umum: cout<<”tulisan”; atau cout<<var x;

Dan baris 3 terdapat **using namespace std** yang berfungsi untuk memanggil class/objek/fungsi yang terdapat dalam namespace std. Misalnya memanggil cout dan cin.

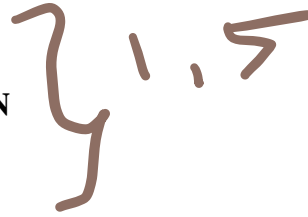
Class Jeruk merupakan class yang akan digunakan pada program ini, atau bisa dikatakan sebagai main class program. Penggunaan Public agar program di atas dapat di compile, perintah public menyatakan bahwa perintah-perintah yang ada di bawahnya dapat diakses diluar class. Perintah public termasuk access specifier (penentu akses). Public berisi fungsi yang mengakses status internal sebuah objek atau dapat dikatakan merupakan method atau perilaku dari sebuah objek.

Semangka : : Semangka(string inputjenis, string inputrasa, string inputberat) merupakan konstruktor adalah fungsi anggota dari suatu kelas yang secara otomatis dijalankan ketika suatu objek diciptakan. Konstruktor mempunyai aturan yaitu nama konstruktor harus sama dengan nama kelas yang mendefinisikannya. Member functions didefinisikan diluar kelas dengan menggunakan binary scope resolution operator (::) yang berfungsi untuk mrngikat nama fungsi ke nama kelas dan mengedentifikasi fungsi dari suatu kelas tertentu. Konstruktor biasanya digunakan untuk melakukan alokasi memori, memberikan nilai awal terhadap anggota kelas (atau disebut inisialisasi), dan melakukan tugas umum lainnya.

Dengan menggunakan konstruktor kita bisa sedikit 'berhemat' dengan tidak membuat suatu fungsi yang secara khusus diciptakan untuk melakukan suatu inisialisasi, namun cukup dengan menggunakan suatu konstruktor. Tanda << merupakan karakter yang harus mengikuti syntax cout. Jadi dimana cout ditulis setelahnya kita wajib menuliskan karakter <<. Dan perintah endl; berfungsi ganti baris. Int main() suatu program dalam C++ dapat berisi banyak fungsi, fungsi adalah modul yang berisi kode-kode untuk menyelesaikan masalah-masalah tertentu. Dan diakhir listing program return 0; untuk menunjukkan program telah selesai dan berhasil.

BAB III

KESIMPULAN



Class adalah salah satu dari konsep OOP yang digunakan untuk membungkus data abstraksi *procedural* sebagai deskripsi tergeneralisir atau rancangan dari sebuah *object* untuk mendefinisikan atau menggambarkan isi dan tingkah laku sebagai entitas dari *object*.

Objek adalah 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.

Class dan *Object* merupakan fitur yang sangat membantu untuk mendirikan sebuah program besar, menjadikan sebuah code program yang ditulis oleh programmer mudah untuk dimengerti, lebih terstruktur, dan juga mudah dalam pemeliharaan program.

Acces modifier memiliki 3 tipe yaitu *public*, *private*, dan *protected*.

Semua system terdiri dari class-class dan objek. Suatu system dicapai melalui kerjasama antar object, interaksi antar object disebut object relationship.

Method merepresentasikan operasi-operasi yang dapat dilakukan oleh objek. Yang membedakan antara variabel dan method adalah method selalu diakhiri dengan () atau {}.

Constructor merupakan suatu method yang akan memberikan nilai awal dari pada saat suatu objek dibuat.

DAFTAR PUSTAKA

Dosen Teknik Informatika. Modul Praktikum Algoritma Pemrograman II. Palangkaraya: Jurusan Teknik Informatika, 2020. Print.

<https://garudacyber.co.id/artikel/588-konsep-dasar-pemrograman-berorientasi-obyek>. Diakses 5 April 2020.

https://id.wikipedia.org/wiki/Pemrograman_berorientasi_objek. Diakses 5 April 2020.

<https://www.belajarcpp.com/tutorial/cpp/access-modifier/> Diakses 5 April 2020.

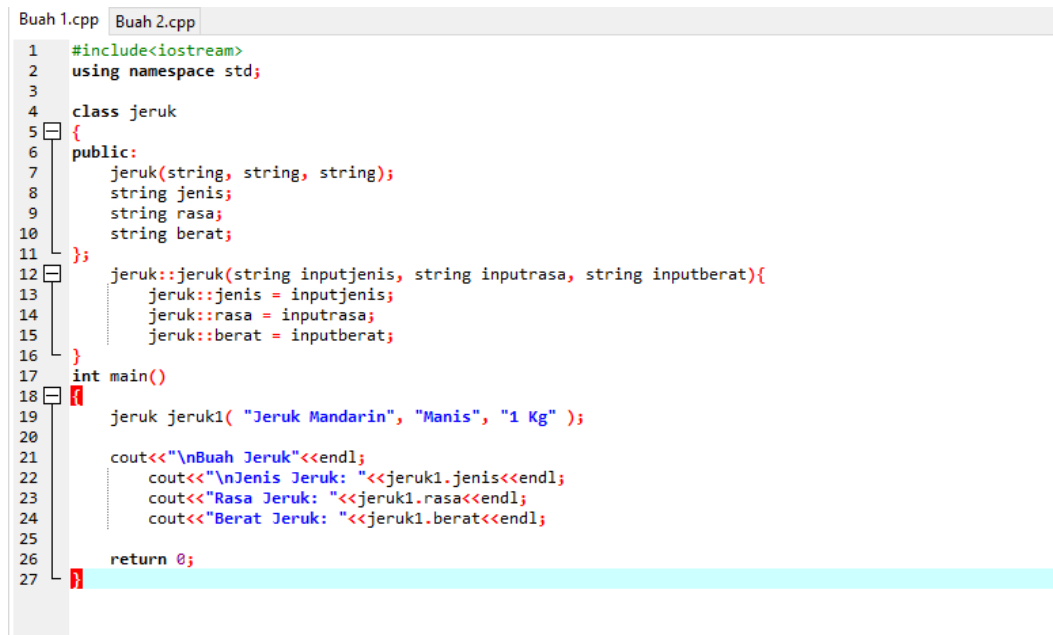
<https://www.belajarcpp.com/tutorial/cpp/class/> 5 April 2020.

<https://www.belajarcpp.com/tutorial/cpp/constructor/> 5 April 2020.

ATUTAP Penulisan daftar pustaka

LAMPIRAN

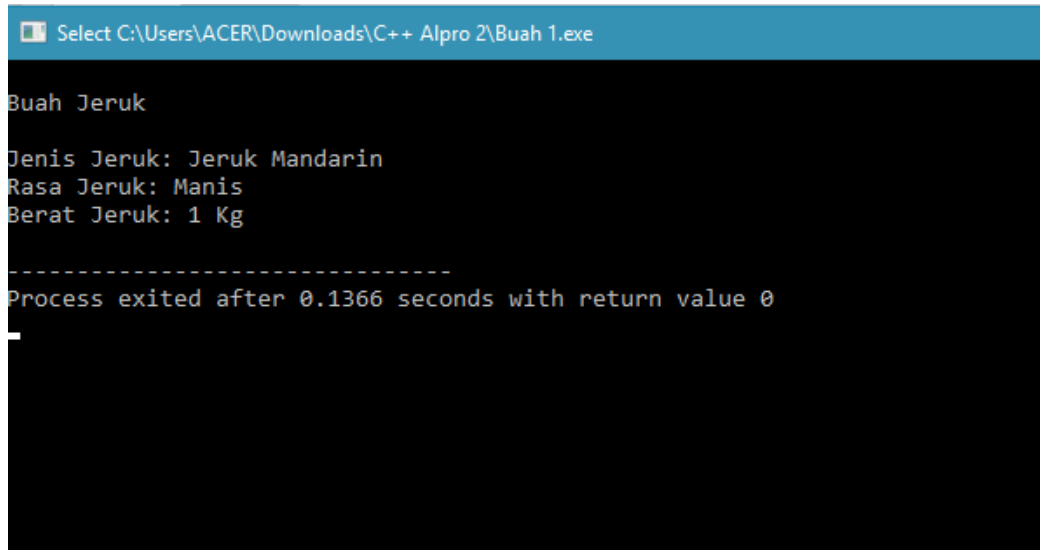
Tampilan program 1



```
Buah 1.cpp  Buah 2.cpp
1  #include<iostream>
2  using namespace std;
3
4  class jeruk
5  {
6  public:
7      jeruk(string, string, string);
8      string jenis;
9      string rasa;
10     string berat;
11 };
12 jeruk::jeruk(string inputjenis, string inputrasa, string inputberat){
13     jeruk::jenis = inputjenis;
14     jeruk::rasa = inputrasa;
15     jeruk::berat = inputberat;
16 }
17 int main()
18 {
19     jeruk jeruk1( "Jeruk Mandarin", "Manis", "1 Kg" );
20
21     cout<<"\nBuah Jeruk"<<endl;
22     cout<<"\nJenis Jeruk: "<<jeruk1.jenis<<endl;
23     cout<<"Rasa Jeruk: "<<jeruk1.rasa<<endl;
24     cout<<"Berat Jeruk: "<<jeruk1.berat<<endl;
25
26     return 0;
27 }
```

nomor gambar

Output Program 1



```
Select C:\Users\ACER\Downloads\C++ Alpro 2\Buah 1.exe

Buah Jeruk

Jenis Jeruk: Jeruk Mandarin
Rasa Jeruk: Manis
Berat Jeruk: 1 Kg

-----
Process exited after 0.1366 seconds with return value 0
```

Tampilan Program 2

```

Buah 2.cpp
1  #include<iostream>
2  using namespace std;
3
4  class semangka
5  {
6  public:
7      semangka(string, string, string);
8      string jenis;
9      string rasa;
10     string berat;
11 };
12 semangka::semangka(string inputjenis, string inputrasa, string inputberat){
13     semangka::jenis = inputjenis;
14     semangka::rasa = inputrasa;
15     semangka::berat = inputberat;
16 }
17 int main()
18 {
19     semangka semangka1( "Semangka Merah", "Manis", "1.5 Kg" );
20
21     cout<<"\nBuah Semangka"<<endl;
22     cout<<"\nJenis semangka: "<<semangka1.jenis<<endl;
23     cout<<"Rasa semangka: "<<semangka1.rasa<<endl;
24     cout<<"Berat semangka: "<<semangka1.berat<<endl;
25
26     return 0;
27 }

```

Output Program 2

```

C:\Users\ACER\Downloads\C++ Alpro 2\Buah 2.exe

Buah Semangka

Jenis semangka: Semangka Merah
Rasa semangka: Manis
Berat semangka: 1.5 Kg

-----
Process exited after 0.1283 seconds with return value 0
Press any key to continue . . .

```