

LAPORAN HASIL PRAKTIKUM ALGORITMA DAN PEMROGRAMAN II



NAMA : Ahmad Daffa Fahrezi
NIM : 193010503008
KELAS : A
MODUL : II (PEWARISAN)

JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS PALANGKA RAYA
2020

**LAPORAN HASIL PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN II**



Nama : Ahmad Daffa Fahrezi
NIM : 193010503008
Kelas : A
Modul : II (PEWARISAN)

Komposisi	MAX	Nilai
BAB I Tujuan dan Landasan Teori	10	8
BAB II Pembahasan	60	50
BAB III Kesimpulan	20	10
Daftar Pustaka	5	5
Lampiran	5	4
Jumlah	100	77

Penilai
Asisten Praktikum

Diana

BAB I

TUJUAN DAN LANDASAN TEORI

1.1. TUJUAN

Setelah menyelesaikan modul ini, mahasiswa diharapkan mampu :

1. membuat kelas baru dari kelas yang sudah ada dengan pewarisan

1.2. LANDASAN TEORI

Dalam PBO, kita mengambil realita kehidupan sehari-hari. Kita melakukan pengamatan bahwa manusia secara alami sering melakukan pengelompokan atas objek atau benda. Se jauh ini kita mengetahui cara untuk melakukan pengelompokan-pengelompokan atas objek-objek yang serupa (menjadi kelas objek).

Selain melakukan kategorisasi terhadap objek yang memiliki sekumpulan atribut dan perilaku yang sama, manusia sering melakukan pengelompokan terhadap objek yang memiliki kesamaan atas beberapa (**tidak semua**) atribut/perilaku. Contoh : pengelompokan atas kendaraan bermotor, kemudian menggrupkannya berdasarkan suatu tipe atau jenis (mobil, truk, sepeda motor, dll.). Setiap subkategori ini merupakan kelas atas objek-objek yang serupa.

Ada beberapa karakteristik yang di-share oleh semua kelompok. Relasi antar kelas-kelas ini disebut dengan **relasi “is-a”**. Dalam setiap kasus, objek yang dikelompokkan bersama dalam sub-kategori merupakan anggota dari kategori yang lebih umum. Contohnya adalah seperti di bawah ini.

- Mobil adalah (“is – a”) kendaraan bermotor.
- Truk adalah (“is – a”) kendaraan bermotor.
- Sepeda motor adalah (“is – a”) kendaraan bermotor.

Objek yang dikelompokkan dalam suatu kelas men-share sekumpulan atribut dan perilaku. Jadi, seluruh objek kendaraan bermotor memiliki sekumpulan atribut dan perilaku yang juga dimiliki oleh objek dari mobil. Keterkaitan antar kelas dalam relasi “is – a” berasal dari kenyataan bahwa sub

kelas memiliki atribut dan perilaku yang dimiliki kelas induk, ditambah atribut dan perilaku yang dimiliki oleh sub kelas tersebut.

Superclass (“kelas dasar” atau “kelas induk”) merupakan kelas yang lebih general dalam relasi “is – a”. Subclass (“kelas turunan” atau “ kelas anak”) merupakan kelas yang lebih spesifik dalam relasi “is – a”. Objek yang dikelompokkan dalam sub kelas memiliki atribut atau perilaku kelas induk, dan juga atribut dan perilaku tambahan. (Jadi, kumpulan atribut dan perilaku sub kelas lebih besar dari super kelas-nya). Relasi “is – a” antar superclass dan subclasses-nya disebut dengan **pewarisan** atau *inheritance*.

Subclass “mewarisi” suatu superclass (atau juga bisa dikatakan sebuah subclass “turunan dari” suatu superclass) karena reuabilitas Perangkat Lunak, membuat kelas baru (kelas turunan) dari kelas yang sudah ada (kelas dasar), kelas turunan mewarisi kelas induk yang mendapatkan data dan perilaku, merupakan bentuk spesial dari kelas induk, dan diperluas dengan perilaku tambahan.

Pewarisan ada dua jenis, yaitu pewarisan tunggal dan pewarisan jamak. Pada *protected access*, **protected** members dapat diakses oleh member kelas dasar, friend kelas dasar, member kelas turunan, dan friend member kelas turunan. Kelas turunan dapat merujuk/mengakses langsung **public** dan **protected** data member kelas induk dengan menggunakan nam atribut yang diakses.

Contoh :

- o Kelas C adalah kelas anak dari kelas B
- o Kelas B merupakan kelas anak dari kelas A.
- o Maka sifat sifat yang diwariskan kelas A ke kelas B juga akan diwariskan ke kelas C.

Keyword untuk pewarisan : **extends**

Contoh Program:

```
class Induk {
```

```

private String var1; //hak akses private berarti tidak dapat diakses oleh kls
anak
public int var2; //hak akses public berarti dapat diakses oleh kls anak
//konstruktor
Induk () {
}
//method
public void cetakData() {
}
}
class Anak extends Induk {
private String dataAnak; //variabel
//konstruktor
Anak () {
}
public void cetak() {
cetakData();//method milik Induk
// ada jg yang memanggil dgn super.cetakData()
}
}
class DemoPewarisan {
public static void main(String args[]) {
Anak anak = new Anak();
anak.cetak();
anak.cetakData();
System.out.println("Demo Pewarisan");
System.out.println("isi var 2="+anak.var2);
}
}

```

A. Keuntungan Pewarisan



Bersifat reusable

Tidak harus menyalin semua data dan method dari suatu kelas jika akan menggunakannya lagi



Kemudahan dalam me-*manage* kelas yang memiliki data dan method yang sama

Untuk memodifikasi suatu data atau method untuk semua subkelas / kelas anak, maka tidak perlu melakukan perubahan di masing-masing kelas anak melainkan hanya pada kelas induk saja.

Dari mekanisme pewarisan yang sudah diuraikan diatas, dapat disimpulkan bahwa pewarisan ini dikelompokkan menjadi tiga, yaitu :

- a. Pewarisan Tunggal (*single inheritance*)
- b. Pewarisan Jamak (*multiple inheritance*), dan
- c. Pewarisan Jamak Maya (*virtual multiple inheritance*)

a. Pewarisan Tunggal (Single Inheritance)

Pewarisan Tunggal adalah pewarisan yang mana jumlah kelas dasarnya tunggal. Pada pewarisan ini, kelas turunan dapat berjumlah lebih dari satu. Pewarisan tunggal dapat digambarkan dengan sintak program sebagai berikut:

```
class A
{
    ...
};
class B : public A
{
    ...
}
```

Sintak di atas adalah mekanisme pewarisan secara public. Dengan implementasi di atas, kelas B merupakan kelas turunan dari kelas A. Selain pewarisan public, pewarisan juga dilakukan secara protected maupun private.

Contoh program pewarisan tunggal :

```
#include<iostream.h>

class makhluk
{
public:
    void berkembang();
};

class hewan : public makhluk
{
public:
    void bergerak();
};

class kuda : public hewan
{
public:
    void berlari();
};

main()
{
    makhluk mk; hewan hw; kuda kd;
    cout<<endl<<" Sifat-sifat dari Makhluk adalah : "<<endl;
    mk.berkembang();
    cout<<endl<<" Sifat-sifat dari Hewan adalah : "<<endl;
    hw.berkembang(); hw.bergerak();
    cout<<endl<<" Sifat-sifat dari Kuda adalah : "<<endl;
```

```

    mk.berkembang(); hw.bergerak(); kd.berlari();
}
void makhluk::berkembang()
{
cout<<" Berkembang biak"<<endl;
}
void hewan::bergerak()
{
cout<<" Bergerak berpindah tempat"<<endl;
}
void kuda::berlari()
{
    cout<<" Berlari sangat kencang seperti angin"<<endl;
}

```

b. Pewarisan Jamak (Multiple Inheritance)

Pewarisan Jamak adalah pewarisan dimana satu kelas diturunkan lebih dari satu kelas yang berbeda. Dalam pewarisan ini jumlah kelas dasarnya lebih dari satu, dan perlu dicatat bahwa kelas dasarnya bukan merupakan turunan dari satu kelas. Kelas turunan mewarisi seluruh sifat dari kelas dasarnya, sehingga sifat dari beberapa kelas dasar dan sifat khas dirinya. Perhatikan sintak dari pewarisan tunggal berikut ini:

```

class A
{
    ...
};
class B
{

```



```

    ...
}
class C: public A, public B
{
    ...
}

```

Pada bentuk tersebut terdapat dua kelas dasar yaitu kelas A dan kelas B yang menurunkan kelas C. Kelas C akan mewarisi sifat dari kelas A maupun sifat dari kelas B, tetapi tidak berlaku sebaliknya.

Perhatikan contoh program berikut ini:

```

#include<iostream.h>

class kuda
{
public :
void berlari()
{
cout<<" > Berlarinya sangat cepat"<<endl;
}
};

class burung
{
public:
void terbang()
{
cout<<" > Terbang menembus awan"<<endl;
}
};

class pegasus: public kuda, public burung

```

```

{
public:
void lariterbang()
{
cout<<" > Bersayap, lari dan dapat terbang ke
angkasa"<<endl;
}
};
main()
{
pegasus pg;
cout<<"Sifat dari PEGASUS yaitu : "<<endl;
pg.berlari();
pg.terbang();
pg.lariterbang();
}

```

c. Pewarisan Jamak Maya (Virtual Multiple Inheritance)

Pewarisan Jamak Maya adalah pewarisan yang mana kelas dasarnya lebih dari satu dan beberapa di antara kelas dasar tersebut merupakan kelas turunan dari kelas dasar yang sama. Mekanisme pewarisan sifat suatu kelas dasar kepada kelas turunan sama dengan pewarisan yang lain. Perhatikan sintak berikut ini:

```

class A
{
...
};
class B: virtual public A

```

```

{
    ...
};
class C: virtual public A
{
    ...
};
class D: public B, public C
{
    ...
};

```

Kelas D merupakan turunan dari kelas B dan C, sedangkan kelas B dan C merupakan kelas turunan dari kelas dasar yang sama yaitu kelas A. Supaya berjalan pewarisan dari kelas A kepada kelas B maupun C harus secara virtual. Kelas virtual A pertama menurunkan kelas B, sedangkan kelas virtual A kedua menurunkan kelas C.

Contoh program dalam pewarisan ini :

```

#include<iostream.h>

class hewan
{
public:
    void bergerak()
    {
        cout<<" # Bergerak berpindah tempat"<<endl;
    }
};

class kuda: virtual public hewan
{

```

```

public :
void berlari()
{
cout<<" # Berlarinya sangat cepat"<<endl;
}
};

class burung: virtual public hewan
{
public:
void terbang()
{
cout<<" # Terbang menembus awan"<<endl;
}
};

class pegasus: public kuda, public burung
{
public:
void lariterbang()
{
cout<<"# Bersayap, lari dan dapat terbang ke angkasa"<<endl;
}
};

main()
{
pegasus pg;
cout<<">> Sifat dari PEGASUS << "<<endl;
cout<<"===== "<<endl;
pg.bergerak();
pg.berlari();
}

```

```

pg.terbang();
pg.lariterbang();
}

```

B. Konstruktor dan Destruktor Pada Pewarisan

Di dalam contoh-contoh diatas belum melibatkan konstruktor dan destruktur secara eksplisit. Di dalam pewarisan suatu konstruktor perlu diperhatikan terutama konstruktor penyalinan. Jika konstruktor kelas dasar hanya berisi pernyataan memberi nilai data anggota private, maka akses private dapat diganti dengan akses protected agar data anggota pada kelas dasar dapat diakses dari kelas turunan.

Contoh programnya :

```

#include<iostream.h>
#include<conio.h>
#include<string.h>
class Kendaraan
{
private:
char nama[15];
public:
Kendaraan(char *nama_kendaraan = "T1AS")
{
strcpy(nama, nama_kendaraan);
cout<<" Hidupkan mesin kendaraan anda ..."<<endl;
}
~Kendaraan()
{
cout<<" Matikan mesin kendaraan anda ..."<<endl;
}
}

```

```

    }

    void info_kendaraan()
    {
        cout<<nama<<" Sedang berjalan ..."<<endl;
    }
};

class Mercy : public Kendaraan
{
public:
    Mercy(char *nama_mercy) : Kendaraan(nama_mercy)
    {
        cout<<" Hidupkan mesin mobil merah ..."<<endl;
    }
    ~Mercy()
    {
        cout<<" Matikan mesin mobil merah itu ..."<<endl;
    }
};

void main()
{
    clrscr();

    Mercy mewah(" Mobil Yang Mewah");
    mewah.info_kendaraan();
    cout<<" Akhir dari permulaaan()..."<<endl;
}

```

Keluarannya :

Hidupkan mesin kendaraan anda ...

Hidupkan mesin mobil merah ...

Mobil Yang Mewah Sedang berjalan ...

Akhir dari permulaan()...

Matikan mesin mobil merah itu ...

Matikan mesin kendaraan anda ...

C. Pewarisan

Pemrograman C++ memungkinkan suatu class dapat mewarisi data ataupun keanggota class lain. Dalam hal ini class yang mewarisi sifat disebut class turunan, sedangkan kelas yang mewarisi disebut class dasar.

Untuk menurunkan class dari yang lain, kita menggunakan operator : colon dalam deklarasi class turunan dengan cara sebagai berikut :

Class derived_class_name: public base_class_name;

Dimana derived_class_name adalah nama class turunan dan base_class_name adalah nama class yang menjadi dasar. Public dapat diganti dengan akses lain misalnya protected atau private, dan menjelaskan akses untuk member yang diturunkan.

Kelas dasar :

- Sifat a
- Sifat b

Kelas turunan :

- Sifat a
- Sifat b
- Sifat c

Sifat a dan sifat b diwariskan pada kelas turunan.

Keuntungan pewarisan sifat adalah suatu kode yang telah ditulis mudah sekali untuk dipanggil kembali. Sifat yang dimaksud dalam class adalah private, protected dan public. Sifat yang diwariskan adalah protected dan public. Private bersifat khusus sehingga tidak dapat diwariskan.

BAB II

PEMBAHASAN

1. Program buah pewarisan pertama.

```
#include<iostream>

#include<string.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.

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

- a. Include merupakan salah satu jenis pengarah preprocessor yang digunakan untuk membaca file.
- b. `#include<iostream>` diperlukan pada program yang melibatkan objek `cout` dan `cin`.
- c. `#include <string>` diperlukan untuk memanggil nama-nama yang di deklarasikan.

```
class buah
{
protected:
char nama[20];
char warna[20];
```

Pada bagian program di atas, yaitu ***Class Buah {*** yang diperlukan untuk membuat objek dan memdefinisikan variabel dan method-method yang ada pada program, Tanda '{' dan '}' berfungsi untuk menandakan awal dan akhir sebuah definisi fungsi dan program.

protected. pada umumnya mirip dengan private namun perbedaannya ada pada bahwa member protected masih bisa diakses oleh class anak sedangkan private tidak.

```
public:
//setter
void setName(char n[20])
{
strcpy(nama,n);
}
void setWarna(char w[20])
{
strcpy(warna,w);
}
//getter
char *getNama()
{
return nama;
}
char *getWarna()
{
return warna;
}
};
```

pada program ini menggunakan mode akses yaitu public. Dan pada mode akses public ini menggunakan tipe data char, sedangkan mode akses. Tanda ‘;’ artinya menyatakan bahwa akhir dari sebuah pernyataan.

Void adalah sebuah fungsi (*function*) yang ada dalam sebuah bahasa pemrograman C, entah itu C++ atau C#. Fungsi ini juga disebut sebagai prosedur (*procedure*). Fungsi ini tidak mengembalikan nilai keluaran (*return output*) yang

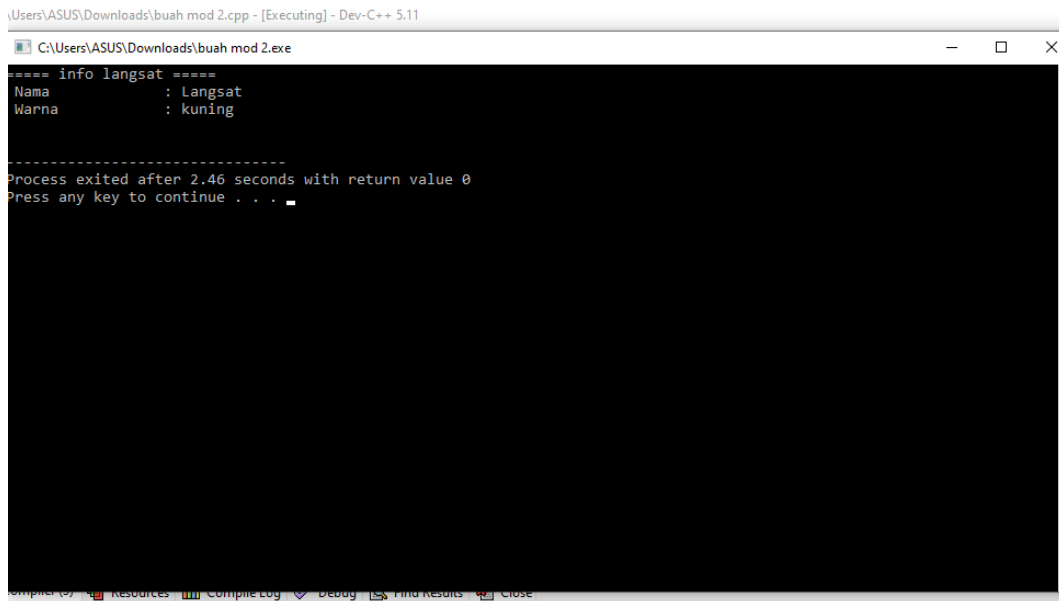
didapat dari hasil proses tersebut, ini kenapa fungsi ini disebut void, secara harfiah berarti kosong.

```
int main( )
{
    buah langsung;
    langsung.setNama("Langsat");
    langsung.setWarna("kuning");

    cout<<"===== info langsung ===== " <<endl;
    cout<<" Nama          : " <<langsang.getNama() <<endl;
    cout<<" Warna          : " <<langsang.getWarna()<<endl;
    cin.get();
}
```

Pada bagian program **Int main()** merupakan bagian utama dari program. **Return** digunakan karena pada main program menggunakan tipe data integer, dan tanda ‘**}**’ menandakan berakhirnya program. *cout* berfungsi untuk menampilkan keluaran atau output ke layar monitor.

Tanda << (dua buah tanda kurang dari berurutan) merupakan sebuah operator yang disebut operator “penyisipan/peletakan”. Operator ini akan mengarahkan operand (data) yang terletak di sebelah kanannya obyek yang terletak di sebelah kiri. *Cout* juga dapat melakukan output secara beruntun. Artinya, satu penulisan *cout* dapat diikuti oleh banyak operator. *Endl* merupakan suatu fungsi manipulator yang digunakan untuk menyisipkan karakter NewLine atau mengatur pindah baris



Gambar 1.1

Nah di output berikut ini dapat dilihat tampilan dari program yg pertama adalah cout yang menunjukkan “info langsung”, kemudian ada nama dari buah dan juga warnanya.

2. Program Buah Pewarisan Kedua.

```
#include <iostream>
#include <conio.h>
#include <string.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.

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

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

b. `#include<iostream>` diperlukan pada program yang melibatkan objek `cout` dan `cin`.

c. `#include <string>` diperlukan untuk memanggil nama-nama yang dideklarasikan.

d. `#include<conio.h>` yang digunakan untuk mengaktifkan fungsi `getch()` (get character and echo) dipakai untuk membaca sebuah karakter dengan sifat karakter yang dimasukkan tidak dimasukkan tidak perlu diakhiri dengan menekan tombol *enter*, dan karakter yang dimasukkan tidak akan ditampilkan di layar.

```
class Buah {
public:
    Buah(char *a,char *b);
    void cetakinfo();
protected:
    char nama[25],kulit[25];
};
Buah::Buah(char *a,char *b){
    strcpy(nama,a);
    strcpy(kulit,b);
}
void Buah::cetakinfo(){
    cout<<"Informasi Buah\n";
}
class buahlangsat:public Buah {
protected:
    char warna[25];
public:
    buahlangsat(char *a,char *b,char *w);
    void cetakinfo();
};
```

```

buahlangsats::buahlangsats(char *a,char *b, char *w):Buah(a,b) {
    strcpy(warna,w);
}
void buahlangsats::cetakinfo() {
    Buah::cetakinfo();
    cout<<"Nama          : "<<nama<<endl;
    cout<<"Warna Kulit    : "<<kulit<<endl;
    cout<<"Warna dalam buah : "<<warna<<endl<<endl;
}

```

Pada bagian program di atas, yaitu **Class Buah {** 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 dua mode akses yaitu public. Dan pada mode akses public ini menggunakan tipe data integer, sedangkan mode akses. Tanda ';' artinya menyatakan bahwa akhir dari sebuah pernyataan.

Void adalah sebuah fungsi (*function*) yang ada dalam sebuah bahasa pemrograman C, entah itu C++ atau C#. Fungsi ini juga disebut sebagai prosedur (*procedure*). Fungsi ini tidak mengembalikan nilai keluaran (*return output*) yang didapat dari hasil proses tersebut, ini kenapa fungsi ini disebut void, secara harfiah berarti kosong.

```

int main() {
int pilih;
do{
ulang:
system("cls");
cout<<"BUAH DAFFA\n\n"
<<"[1] Buah Langsung\n"
<<"[2] Buah Kelengkeng\n"

```

```
<<"[3] Keluar\n\n"  
<<"Masukan pilihan anda : "; cin>>pilih;  
cout<<endl;  
switch(pilih) {  
case 1:{  
buahlangsat bb("Buah Langsung","Kuning Kecoklatan","Putih Bening");  
bb.cetakinfo();  
system("pause");  
break;  
}  
case 2:{  
buahlangsat bb("Kelengkeng","kuning Kecoklatan","Putih Bening");  
bb.cetakinfo();  
system("pause");  
break;  
}  
case 3:{  
system("cls");  
break;  
}  
default:{  
cout<<"Pilihan tidak tersedia!\n"  
<<"Silahkan memilih sesuai nomor yang tersedia."  
goto ulang;  
break;  
}  
}  
}
```

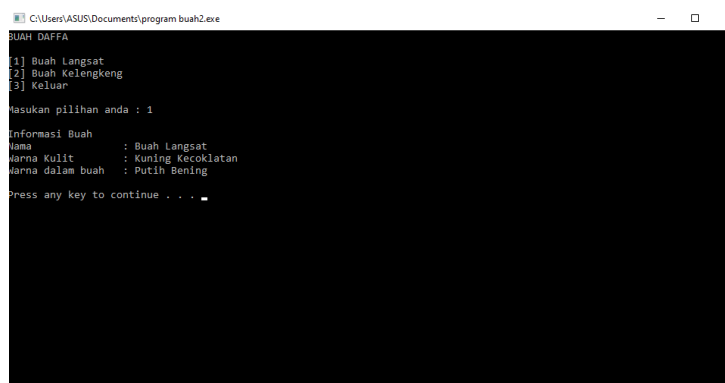
```
while(pilih!=3);  
return 0;  
}
```

Pada bagian program **Int main()** merupakan bagian utama dari program. **Return** digunakan karena pada main program menggunakan tipe data integer, dan tanda ‘**}**’ menandakan berakhirnya program. *cout* berfungsi untuk menampilkan keluaran atau output ke layar monitor.

Tanda << (dua buah tanda kurang dari berurutan) merupakan sebuah operator yang disebut operator “penyisipan/peletakan”. Operator ini akan mengarahkan operand (data) yang terletak di sebelah kanannya obyek yang terletak di sebelah kiri. *Cout* juga dapat melakukan output secara beruntun. Artinya, satu penulisan *cout* dapat diikuti oleh banyak operator. *endl* merupakan suatu fungsi manipulator yang digunakan untuk menyisipkan karakter NewLine atau mengatur pindah baris

Switch — case merupakan jenis seleksi yang dirancang khusus untuk menangani pengambilan keputusan yang melibatkan sejumlah atau banyak alternatif penyelesaian. Pernyataan switch — case ini memiliki kegunaan sama seperti if — else bertingkat, tetapi penggunaannya untuk memeriksa data yang bertipe karakter atau integer.

Berikut output program :



```
C:\Users\ASUS\Documents\program buah2.exe  
BUAH DAFFA  
[1] Buah Langsung  
[2] Buah Kelengkeng  
[3] Keluar  
Masukan pilihan anda : 1  
Informasi Buah  
Nama : Buah Langsung  
Warna Kulit : Kuning Kecoklatan  
Warna dalam buah : Putih Bening  
Press any key to continue . . .
```

Gambar 1.2

Nah pada output tersebut dapat dilihat bahwasanya kita bisa memilih pilihan yang terdapat di menu untuk menampilkan program info buah. Sehingga muncul nama buah warna kulit dan warna dalam buah.

Paragraf

BAB III KESIMPULAN

Dari modul ini dapat ditarik kesimpulan sebagai berikut:

1. Pewarisan (*inheritance*) adalah penurunan sifat yang ada pada suatu kelas kepada kelas baru yang menjadi turunannya.
2. Superclass (“kelas dasar” atau “kelas induk”) merupakan kelas yang lebih general dalam relasi “is – a”.
3. Subclass (“kelas turunan” atau “ kelas anak”) merupakan kelas yang lebih spesifik dalam relasi “is – a”.
4. Objek yang dikelompokkan dalam sub kelas memiliki atribut atau perilaku kelas induk, dan juga atribut dan perilaku tambahan.
5. Dari mekanisme pewarisan yang sudah diuraikan diatas, dapat disimpulkan bahwa pewarisan ini dikelompokkan menjadi tiga, yaitu :
 1. Pewarisan Tunggal (*single inheritance*)
 2. Pewarisan Jamak (*multiple inheritance*), dan
 3. Pewarisan Jamak Maya (*virtual multiple inheritance*)

~~DAFTAR~~

DAFTAR PUSTAKA

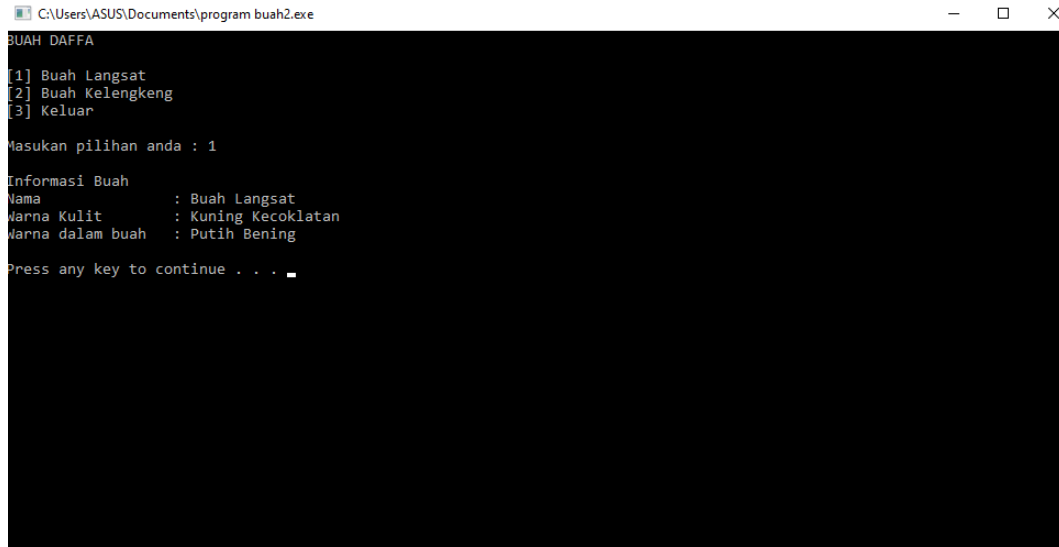
Tim Dosen Algoritma Pemrograman II. 2020. *Modul Praktikum Algoritma Pemrograman II*. Universitas Palangka Raya : UPR. Fakultas Teknik.

Movita09. 2013. *Pewarisan*. <http://blogspot.com/2013/12/pewarisan.html>

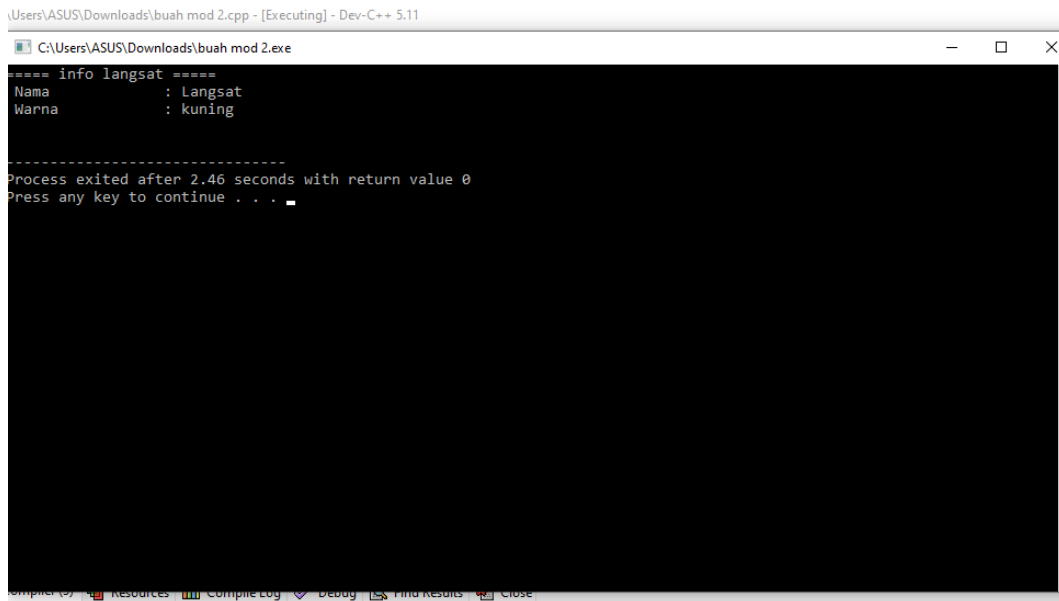
Diakses 15 April 2020 jam 18.20 WIB.

~~BAB V~~

LAMPIRAN



Gambar 1.2



Gambar 1.1