

**LAPORAN HASIL PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN II**



NAMA : EVAN ALPHARIO IMANUEL
NIM : 193030503059
KELAS : A
MODUL : II (PEWARISAN)

JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS PALANGKA RAYA
2020

LAPORAN HASIL PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN II



Nama : Evan Alphario Imanuel
NIM : 193030503059
Kelas : A
Modul : PEWARISAN

Komposisi	MAX	Nilai
BAB I Tujuan dan Landasan Teori	10	7
BAB II Pembahasan	60	50
BAB III Kesimpulan	20	13
Daftar Pustaka	5	4
Lampiran	5	5
Jumlah	100	79

Penilai
Asisten Praktikum

Diana

BAB I

TUJUAN DAN LANDASAN TEORI

I. TUJUAN

Setelah menyelesaikan modul ini, mahasiswa diharapkan mampu membuat kelas baru dari kelas yang sudah ada dengan pewarisan

II. DASAR TEORI

Dalam PBO, kita mengambil realita kehidupan sehari-hari. Kita melakukan pengamatan bahwa manusia secara alami sering melakukan pengelompokan atas objek atau benda. Sejauh 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 satu sub-kategori merupakan anggota dari kategori yang lebih umum. Contohnya adalah seperti di bawah ini.

1. Mobil adalah (“is-a”) kendaraan bermotor.
2. Truk adalah (“is-a”) kendaraan bermotor
3. Sepeda Motor adalah (“is-a”) kendaraan bermotor

Objek yang dikelompokkan dalam satu 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 dan 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 reusabilitas 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 kelas turunan. Kelas turunan dapat merujuk/mengakses langsung **public** dan **protected** data member kelas induk dengan menggunakan nama atribut yang diakses.

Seperti halnya dalam arti Pewarisan itu sendiri yang dimaksud dengan Inheritance adalah dimana suatu entitas/obyek dapat mempunyai entitas/obyek turunan. Dengan konsep inheritance, sebuah class dapat mempunyai class turunan.

Seperti halnya manusia pada umumnya terdapat orang tua dan anak pada PBO juga dikenal parent class atau base class dan subclass atau child class. Dimana subclass atau childclass mewarisi semua data yang ada di parent class atau base class atau dapat disimpulkan jika subclass atau childclass adalah perluasan dari parent class atau base class.

Dalam contohnya kita dapat mengambil contoh makhluk hidup sebagai parent class dengan method bernafas, bergerak, dan berkembang biak.

Dan kita menentukan manusia, hewan, dan tumbuhan sebagai childclass dan method dari parent class terdapat dalam childclass.

atau dalam deklarasinya dapat kita tulis `public class manusia extends makhluk_hidup { ... }`

KEUNTUNGAN INHERITANCE :

1. Subclass menyediakan state/behaviour yang spesifik yang membedakannya dengan superclass, hal ini akan memungkinkan programmer Java untuk menggunakan ulang source code dari superclass yang telah ada.
2. Programmer Java dapat mendefinisikan superclass khusus yang bersifat generik, yang disebut abstract class, untuk mendefinisikan class dengan behaviour dan state secara umum.

ISTILAH INHERITANCE :

Extends : ini adalah keyword agar sebuah class menjadi subclass.

Superclass : Menunjukkan class yang berarti parent class dari subclass/class anak.

Subclass : adalah class anak atau turunan dari superclass/parent class.

Super Keyword : untuk memanggil konstruktor dari superclass atau menjadi variabel yang mengacu pada superclass.

Method Overriding : Pendefinisian ulang method yang sama pada subclass

Dalam inheritance, method overriding berbeda dengan method overloading.

Jika method overriding adalah mendefinisikan kembali method yang sama, baik nama method maupun signature atau parameter yang diperlukan dalam subclass,

Sedangkan method overloading adalah mendefinisikan method yang memiliki nama yang sama saja.

PEWARISAN TUNGGAL (SINGLE INHERITANCE)

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.

PEWARISAN JAMAK (MULTIPLE INHERITANCE)

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.

III. LANGKAH KERJA

1. Buatlah 2 program dengan tema buah/hewan menggunakan prinsip pewarisan



BAB II

PEMBAHASAN

1. Program Buah Pertama

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

int b;

class buahJambu{
    public:
        buahJambu(int);
        int totBeli();
        int jumMatang();

    protected:
        int beli;
        int matang;
};

buahJambu::buahJambu(int b){
    beli=b;
    matang=b;
}

buahJambu::totBeli(){
    return beli;
}

buahJambu::jumMatang(){
    return matang;
}
```



```

}
class JambuBiji : public buahJambu{
    public:
        JambuBiji(int);
void JumMatang();
    float belijb;
};
JambuBiji::JambuBiji(int b):buahJambu(b){
    cout <<"Total Jambu Biji Dibeli"<<endl; belijb = b;
}
void JambuBiji::JumMatang(){
    buahJambu::totBeli();
    cout <<"Jambu"<< belijb << " Beli : [" << beli << ", " << matang << "]"
<< endl;
}

int main(){
    buahJambu bl(10);
    buahJambu mt(7);
    JambuBiji jb(8);
    cout << "====="<< endl;
    cout << "||Membeli Buah Jambu      ||"<< endl;
    cout << "====="<< endl;
    cout << "||Total Buah Yang Di Beli : " <<bl.totBeli()<<endl;
    cout << "||Jumlah Buah Yang Matang : " <<mt.jumMatang()<<endl;
    cout << "||Total Jambu Biji Dibeli : " <<jb.totBeli();
    return 0;
}

```

Gambar Input 2.1

Dari program di atas pada bagian di bawah ini

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

Fungsi `#include<iostream>` adalah agar kita dapat menggunakan file header seperti perintah (`cout`, `cin`, `endl`,). Sedangkan “`using namespace std;`” digunakan untuk memanggil namespace yang memiliki nama ‘`std`’. Namespace ‘`std`’ merupakan standar namespace dari C++ yang biasa kita gunakan untuk memanggil class/object/fungsi yang terdapat di dalam namespace tersebut.

```
class buahJambu{
    public:
        buahJambu(int);
        int totBeli();
        int jumMatang();
    protected:
        int beli;
        int matang;
};
```

Pada program di atas dideklarasikan “class buahJambu” yang berisi “Public” dan “Protected” yang dimana isi dari mode akses “public” tersebut terdiri dari `buahJambu(int);`, `int totBeli();` dan `int jumMatang();` yang akan digunakan untuk pendeklarasian constructor dan function pada program di bawahnya. Sedangkan untuk “protected” akan di akses sebagai member function pada bagian fungsi. Dan akan digunakan sebagai pewarisan pada bagian class turunan. Seperti pada bagian di bawah ini :

```
buahJambu::buahJambu(int b){
    beli=b;
    matang=b;
}
buahJambu::totBeli(){
    return beli;
}
```

```

buahJambu::jumMatang(){
    return matang;
}
class JambuBiji : public buahJambu{
    public:
        JambuBiji(int);
void JumMatang();
    float belijb;
};
JambuBiji::JambuBiji(int b):buahJambu(b){
    cout <<"Total Jambu Biji Dibeli"<<endl; belijb = b;
}
void JambuBiji::JumMatang(){
    buahJambu::totBeli();
    cout <<"Jambu"<< belijb << " Beli : [" << beli << ", " << matang << "]"
<< endl;
}

```

Pada bagian program di atas “buahJambu::buahJambu(int b)” merupakan bagian constructor yang berisi protected beli dan matang yang di deklarasikan sebagai variable “b”.

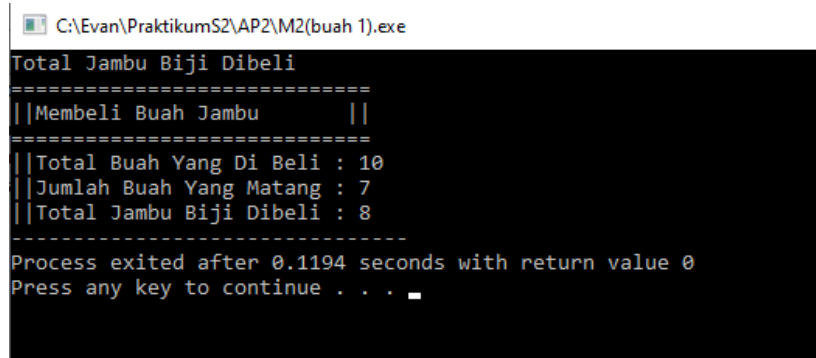
Pada bagian “buahJambu::totBeli()” dan “buahJambu::jumMatang()” yang dideklarasikan sebagai function dari class buahJambu yang berisi member function dari protected di mana berisi tipe balikan yang akan mengembalikannya ke beli dan matang pada bagian protected. Dan fungsi dari protected adalah untuk mempertahankan apa yang ada di dalamnya setelah program di jalankan sehingga tidak hilang dan bisa di gunakan lagi pada bagian class turunan seperti pada class turunan JambuBiji di atas. Yang dimana di setiap class turunan akan menggunakan bagian dari protected pada class induknya dan kemudian menambahkan variable baru pada bagian class turunan nya. Seperti pada bagian constructor JambuBiji::JambuBiji diamana di bagian bawahnya di deklarasikan variable baru yaitu “belijb = b;” .

```

int main(){
    buahJambu bl(10);
    buahJambu mt(7);
    JambuBiji jb(8);
    cout << "===== "<< endl;
    cout << "||Membeli Buah Jambu      ||"<< endl;
    cout << "===== "<< endl;
    cout << "||Total Buah Yang Di Beli : " <<bl.totBeli()<<endl;
    cout << "||Jumlah Buah Yang Matang : " <<mt.jumMatang()<<endl;
    cout << "||Total Jambu Biji Dibeli : " <<jb.totBeli();
    return 0;
}

```

Sedangkan untuk program di atas ini adalah program utama“(int main ())” Diamana pada program utama tersebut sudah di deklarasikan class dengan nama buahJambu bl dan buahJambu mt dengan nilai 10 dan 7 kemudian dari class turunan yaitu JambuBiji jb dengan nilai 8. Cout berfungsi untuk mencetak kalimat yang sudah di masukan pada program.pada cout “total buah yang di beli dan total buah yang matang” terdapat lanjutan yaitu <<bl.totBeli()<<endl; dan <<mt.jumMatang()<<endl; dimana tanda “<<” berfungsi untuk melanjutkan cout(character out) yang berisi bl.totBeli dan mt.jumBeli yang dimana mt dan bl nilai nya sudah ditentukan sebagai 10 dan 7. Dan pada bagian kelas turunan pada program utama yaitu JambuBiji yang bernilai 8 dengan cout<<” total jambu biji dibeli”<<jb.totBeli yang dimana saat program di jalankan akan mengambil dari deklarasi JambuBiji jb(8) yang akan menghasilkan output sebagai berikut :



```

C:\Evan\PraktikumS2\AP2\M2(buah 1).exe
Total Jambu Biji Dibeli
=====
||Membeli Buah Jambu ||
=====
||Total Buah Yang Di Beli : 10
||Jumlah Buah Yang Matang : 7
||Total Jambu Biji Dibeli : 8
-----
Process exited after 0.1194 seconds with return value 0
Press any key to continue . . .

```

Gambar output 2.2

2. Program Buah Kedua

```
#include <iostream>
using namespace std;
int b;
class buahAnggur{
    public:
        buahAnggur(int);
        int totBeli();
        int jumMatang();
    protected:
        int beli;
        int matang;
};
buahAnggur::buahAnggur(int b){
    beli=b;
    matang=b;
}
buahAnggur::totBeli(){
    return beli;
}
buahAnggur::jumMatang(){
    return matang;
}
class AnggurTanpaBiji : public buahAnggur{
    public:
        AnggurTanpaBiji(int);
void JumMatang();
```

```

        float belijb;
    };
    AnggurTanpaBiji::AnggurTanpaBiji(int b):buahAnggur(b){
        cout <<"Total AnggurTanpaBiji"<<endl; belijb = b;
    }
    void AnggurTanpaBiji::JumMatang(){
        buahAnggur::totBeli();
        cout <<"Anggur"<< belijb << " Beli : [" << beli << ", " << matang << "]" <<
endl;
    }
    int main(){
        buahAnggur bl(20);
        buahAnggur mt(18);
        AnggurTanpaBiji jb(15);
        cout << "===== "<< endl;
        cout << " || Membeli Buah Anggur      || "<< endl;
        cout << "===== "<< endl;
        cout << " || Total Buah Yang Di Beli : " << bl.totBeli()<<endl;
        cout << " || Jumlah Buah Yang Matang : " << mt.jumMatang()<<endl;
        cout << " || Total AnggurTanpaBiji : " << jb.totBeli();
        return 0;
    }

```

Gambar Input 2.3

```

#include <iostream>
using namespace std;

```

Fungsi #include<iostream> adalah agar kita dapat menggunakan file header seperti perintah (cout, cin, endl,). Sedangkan "using namespace std;" digunakan untuk memanggil namespace yang memiliki nama 'std'. Namespace 'std' merupakan standar namespace dari C++ yang biasa kita gunakan untuk memanggil class/object/fungsi yang terdapat di dalam namespace tersebut.

```

class buahAnggur{
    public:
        buahAnggur(int);
        int totBeli();
        int jumMatang();
    protected:
        int beli;
        int matang;
};

```

Pada program di atas dideklarasikan “class buahAnggur” yang berisi “Public” dan “Protected” yang dimana isi dari mode akses “public” tersebut terdiri dari buahAnggur(int); , int totBeli(); dan int jumMatang(); yang akan digunakan untuk pendeklarasian constructor dan function pada program di bawahnya. Sedangkan untuk “protected” akan di akses sebagai member fuction pada bagian fungsi. Dan akan digunakan sebagai pewarisan pada bagian class turunan. Seperti pada bagian di bawah ini :

```

buahAnggur::buahAnggur(int b){
    beli=b;
    matang=b;
}
buahAnggur::totBeli(){
    return beli;
}
buahAnggur::jumMatang(){
    return matang;
}
class AnggurTanpaBiji : public buahAnggur{
    public:
        AnggurTanpaBiji(int);
}

```

```

void JumMatang();

    float belijb;

};

AnggurTanpaBiji::AnggurTanpaBiji(int b):buahAnggur(b){

    cout <<"Total AnggurTanpaBiji"<<endl; belijb = b;

}

void AnggurTanpaBiji::JumMatang(){

    buahAnggur::totBeli();

    cout <<"Anggur"<< belijb << " Beli : [" << beli << ", " << matang << "]" <<
endl;

}

```

Pada bagian program di atas “buahAnggur::buahAnggur(int b) “ merupakan bagian constructor yang berisi protected beli dan matang yang di deklarasikan sebagai variable “b”.

Pada bagian “buahAnggur::totBeli()” dan “buahAnggur::jumMatang()” yang dideklarasikan sebagai function dari class buahAnggur yang berisi member function dari protected di mana berisi tipe balikan yang akan mengembalikannya ke beli dan matang pada bagian protected. Dan fungsi dari protected adalah untuk mempertahankan apa yang ada di dalamnya setelah program di jalankan sehingga tidak hilang dan bisa di gunakan lagi pada bagian class turunan seperti pada class turunan AnggurTanpaBiji di atas. Yang dimana di setiap class turunan akan menggunakan bagian dari protected pada class induknya dan kemudian menambahkan variable baru pada bagian class turunan nya. Seperti pada bagian constructor AnggurTanpaBiji:: AnggurTanpaBiji dimana di bagian bawahnya di delkarasikan variable baru yaitu “belijb = b; ” .

```

int main(){

    buahAnggur bl(20);

    buahAnggur mt(18);

    AnggurTanpaBiji jb(15);

```



```

        cout << "=====" << endl;
        cout << "||Membeli Buah Anggur    ||" << endl;
        cout << "=====" << endl;
        cout << "||Total Buah Yang Di Beli : " << bl.totBeli() << endl;
        cout << "||Jumlah Buah Yang Matang : " << mt.jumMatang() << endl;
        cout << "||Total AnggurTanpaBiji : " << jb.totBeli();
        return 0;
    }

```

Sedangkan untuk program di atas ini adalah program utama“(int main ())” Dimana pada program utama tersebut sudah di deklarasikan class dengan nama buahAnggur bl dan buahAnggur mt dengan nilai 20 dan 18 kemudian dari class turunan yaitu AnggurTanpaBiji jb dengan nilai 15. Cout berfungsi untuk mencetak kalimat yang sudah di masukan pada program.pada cout “total buah yang di beli dan total buah yang matang” terdapat lanjutan yaitu <<bl.totBeli()<<endl; dan <<mt.jumMatang()<<endl; dimana tanda “<<” berfungsi untuk melanjutkan cout(character out) yang berisi bl.totBeli dan mt.jumBeli yang dimana mt dan bl nilai nya sudah ditentukan sebagai 20 dan 18. Dan pada bagian kelas turunan pada program utama yaitu AnggurTanpaBiji yang bernilai 15 dengan cout<<” Total AnggurTanpaBiji”<<jb.totBeli yang dimana saat program di jalankan akan mengambil dari deklarasi AnggurTanpaBiji jb(15) yang akan menghasilkan output sebagai berikut :

```

C:\Evan\PraktikumS2\AP2\M2(buah 2).exe
Total AnggurTanpaBiji
=====
||Membeli Buah Anggur    ||
=====
||Total Buah Yang Di Beli : 20
||Jumlah Buah Yang Matang : 18
||Total AnggurTanpaBiji : 15
-----
Process exited after 0.1656 seconds with return value 0
Press any key to continue . . .

```

Gambar Output 2.4

BAB III

KESIMPULAN

Dapat ditarik kesimpulan sebagai berikut, dari program pewarisan di atas dimana suatu class memiliki class turunan yang dimana isi dari class induk nya yang dinyatakan sebagai variabel protected dapat diwariskan pada class turunan nya, dan pada bagian class turunan nya dapat dideklarasikan function class baru. Untuk membuat class turunan maka bentuk nya adalah “class B : public A” (nama class turunan baru kemudian public[nama class induk]).

DAFTAR PUSTAKA

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

NoName,2015.Compiler File Header dan Fungsinya.

<https://almuhtadi93.wordpress.com/2015/01/25/compiler-file-header-dan-fungsinya-using-namespace-std-cara-membuat-komentar-integer-string-dan-char/>.

Diakses 12 April 2020 jam 22.00.

Pascal,2014.Macam-macam file header dan fungsinya di bahasa c++.

http://pascaldhika.blogspot.com/2014/03/macam-macam-file-header-dan-fungsinya_22.html.

Diakses 12 April 2020 jam 22.30.

Unknown,2013.Pengertian PBO.

<http://hadiprojek.blogspot.com/2013/03/pemrograman-berorientasi-objek.html>.

Diakses 12 April 2020 jam 23.00.

Wilmansalawu,2017.Pewarisan Dalam PBO.

<https://wilmansalawu.wordpress.com/2017/12/13/inheritance-pewarisan-dalam-pemrograman-berorientasi-objek-pbo/>.

Diakses 13 April 2020 jam 00.01.

LAMPIRAN

```
#include <iostream>
using namespace std;
int b;
class buahJambu{
    public:
        buahJambu(int);
        int totBeli();
        int jumMatang();
    protected:
        int beli;
        int matang;
};
buahJambu::buahJambu(int b){
    beli=b;
    matang=b;
}
buahJambu::totBeli(){
    return beli;
}
buahJambu::jumMatang(){
    return matang;
}
class JambuBiji : public buahJambu{
    public:
        JambuBiji(int);
    void JumMatang();
    float belijb;
};
JambuBiji::JambuBiji(int b):buahJambu(b){
    cout <<"Total Jambu Biji Dibeli"<<endl; belijb = b;
}
void JambuBiji::JumMatang(){
    buahJambu::totBeli();
    cout <<"Jambu"<< belijb << " Beli : [" << beli << ", " << matang << "]"
<< endl;
}
int main(){
    buahJambu bl(10);
    buahJambu mt(7);
    JambuBiji jb(8);
    cout << "====="<< endl;
    cout << "||Membeli Buah Jambu      ||"<< endl;
    cout << "====="<< endl;
    cout << "||Total Buah Yang Di Beli : " <<bl.totBeli()<<endl;
```

```

        cout << "Jumlah Buah Yang Matang : " << mt.jumMatang() << endl;
        cout << "Total Jambu Biji Dibeli : " << jb.totBeli();
        return 0;
    }

```

Gambar Input 5.1

```

#include <iostream>
using namespace std;
int b;
class buahAnggur{
    public:
        buahAnggur(int);
        int totBeli();
        int jumMatang();
    protected:
        int beli;
        int matang;
};
buahAnggur::buahAnggur(int b){
    beli=b;
    matang=b;
}
buahAnggur::totBeli(){
    return beli;
}
buahAnggur::jumMatang(){
    return matang;
}
class AnggurTanpaBiji : public buahAnggur{
    public:
        AnggurTanpaBiji(int);
    void JumMatang();
        float belijb;
};
AnggurTanpaBiji::AnggurTanpaBiji(int b):buahAnggur(b){
    cout << "Total AnggurTanpaBiji" << endl; belijb = b;
}
void AnggurTanpaBiji::JumMatang(){
    buahAnggur::totBeli();
    cout << "Anggur" << belijb << " Beli : [" << beli << ", " << matang << "]" <<
endl;
}
int main(){
    buahAnggur bl(20);

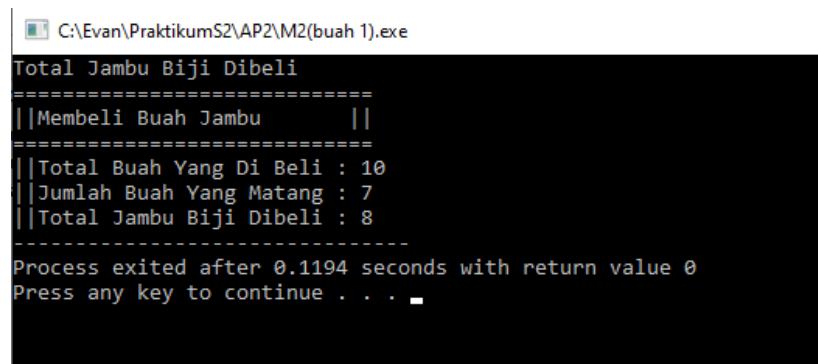
```

```

    buahAnggur mt(18);
    AnggurTanpaBiji jb(15);
    cout << "===== "<< endl;
    cout << " || Membeli Buah Anggur      || "<< endl;
    cout << "===== "<< endl;
    cout << " || Total Buah Yang Di Beli : " << bl.totBeli() << endl;
    cout << " || Jumlah Buah Yang Matang : " << mt.jumMatang() << endl;
    cout << " || Total AnggurTanpaBiji : " << jb.totBeli();
    return 0;
}

```

Gambar Input 5.2

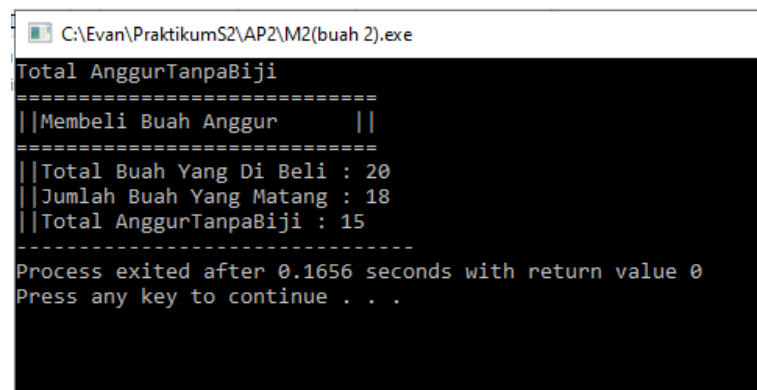


```

C:\Evan\PraktikumS2\AP2\M2(buah 1).exe
Total Jambu Biji Dibeli
=====
|| Membeli Buah Jambu      ||
=====
|| Total Buah Yang Di Beli : 10
|| Jumlah Buah Yang Matang : 7
|| Total Jambu Biji Dibeli : 8
-----
Process exited after 0.1194 seconds with return value 0
Press any key to continue . . .

```

Gambar Output 5.3



```

C:\Evan\PraktikumS2\AP2\M2(buah 2).exe
Total AnggurTanpaBiji
=====
|| Membeli Buah Anggur      ||
=====
|| Total Buah Yang Di Beli : 20
|| Jumlah Buah Yang Matang : 18
|| Total AnggurTanpaBiji : 15
-----
Process exited after 0.1656 seconds with return value 0
Press any key to continue . . .

```

Gambar Output 5.4