LAPORAN HASIL PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN II

NAMA : **M. ADE SHOFY**
NIM : **193010503006**
KELAS : **A**
MODUL : **II (PEWARISAN)**

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 : PEWARISAN

Komposisi	MAX	Nilai
BAB I Tujuan dan Landasan Teori	10	8
BAB II Pembahasan	60	50
BAB III Kesimpulan	20	13
Daftar Pustaka	5	5
Lampiran	5	5
Jumlah	100	81

Penilai

Asisten Praktikum

Diana

BAB I
TUJUAN DAN LANDASAN TEORI

I. TUJUAN

Setelah menyelesaikan modul ini, mahasiswa diharapkan mampuu membuat kelas baru dari kelas yang sudaah ada dengan pewarisan.

II. DASAR TEORI

1. Pengertian Inheritance Atau Pewarisan

Inheritance atau Pewarisan adalah salah satu konsep pada object oriented programming, yang mengadopsi konsep pewarisan yang di miliki oleh object pada dunia nyata. Konsep Pewarisan ini memiliki model relasi "adalah", dimana sebuah class akan dimungkinkan untuk mengambil isi dari class lain sebagai isi dari class tersebut.

Konsep pewarisan akan mempermudah kita dalam membuat dan perawatan sebuah program, dengan menghilangkan "penggunaan kode berulang" saat pembuatan banyak class yang serupa.

Contoh kasus yang tepat untuk penggunaan konsep pewarisan adalah ketika kita sedang membuat class macam-macam orang, dimana pertama-tama anda akan membuat sebuah class orang sebagai base class yang memiliki attribute dan method layaknya sebagai manusia pada dunia nyata, lalu base class tersebut isinya dapat di warisi ke derived classes, dimana derived merupakan macam-macam golongan dari aktivitas manusia, misalnya class pekerja, class pelajar, class pengusaha dan lain sebagainya.

Dalam kasus tersebut, class dari macam-macam orang (pelajar, pengusaha, pekerja) harus mewarisi isi dari class orang, karena beberapa class tersebut memang membutuhkan identitas (attribute) dan sifat (method) dasar menjadikanya layaknya orang, dan hal itu dimiliki oleh class orang. Dengan hal ini kita tidak perlu mengetikan kode berulang kali yang mirip seperti isi dari class orang ke beberapa class yang membutuhkan isi tersebut.

2. Cara Mendirikan Inheritance atau pewarisan

Untuk mendirikan sebuah base class, sama seperti layaknya kita membuat class biasa. Untuk mendirikan derived class dan mewarisi informasi dari base class, mendirikan class sama seperti kita mendirikan class seperti biasanya dan akan dibutuhkan tambahan berupa daftar class penurunan (base class) yang ditulis setelah tanda :.

Bentuk penulisan.

```
1.class nama_class : access_specifier base_class{
2.//…
3.};
```

Penjelasan.

- class nama_class : adalah tempat dimana anda membuat sebuah *class* dengan nama yang anda butuhkan yang diikuti sebuah *keyword* class.
- Tanda : : adalah tanda untuk memisahkan *derived class* dan *class* turunan-nya.
- access_specifier base_class : adalah tempat dimana kita mendaftarkan *base class* yang ingin kita turunkan informasinya, dengan di ikuti *access specifier* yang akan menentukan sifat dari *base class* tersebut dalam *derivered class* itu.

3. Tipe-Tipe Inheritance Atau Pewarisan

Pemberian access specifier pada daftar base class dalam mendirikan derived class dapat berupa public, protected atau private, masing-masing akan memberikan efek berbeda pada derived class tersebut. Dan efek tersebut adalah :

- **Public Inheritance :** Jika mendaftarkan base class pada derived class menggunakan access specifier public maka akan membuat member dari base class yang memiliki sifat private, protected, dan public akan menjadi diri mereka sendiri pada derived class tersebut.
- **Protected Inheritance :** jika mendaftarkan base class pada derived class menggunakan access specifier protected maka akan membuat

member dari base class yang bersifat protected dan public menjadi bersifat protected pada derived class.

- **Private Inheritance :** jika mendaftarkan base class pada derived class menggunakan access specifier private maka akan membuat member dari base class yang bersifat protected dan public menjadi bersifat private pada derived class.

4. Multiple Inheritence atau Pewarisan

Dalam bahasa pemrograman C++ dimungkinkan untuk mendaftarkan banyak *base class* pada satu *derived class*. Cara penulisanya sama seperti kita mendaftarkan satu *base class* pada satu *derived class*, hanya daftar dari *base class* akan ditulis secara sejejar dan dipisahkan dengan tanda koma , .

Contoh penulisan.

```
1.class pegawai: public orang, private manusia{
2.//…
3.}
```

Dengan contoh di atas maka class dari pegawai akan mewarisi informasi dari dua base class yaitu orang dan manusia.

BAB II

PEMBAHASAN

1. Program pertama

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

//Kelas Dasar "Buah"
class Buah
{
private:
    char nama[15];
    char jenis[15];

public:
    //konsruktor
    Buah(char *nm, char *jns)
    {
        cout<<"//konstruktor kelas Buah dijalankan"<<endl;
        strcpy(nama, nm);
        strcpy(jenis, jns);
    }

    ~Buah()
    {
        cout<<"//destruktor kelas Buah dijalankan"<<endl;
        _getche();
    }

    void infoBuah()
    {
        cout<<endl;
        cout<<"Info salah satu jenis buah"<<endl;
        cout<<"Nama       : "<<nama<<endl;
        cout<<"Jenis      : "<<jenis<<endl;
    }
};

class Mangga : public Buah
{
private:
    char ciri_khusus[25];

public:
     //konstruktor
```

```
    Mangga(char *nm, char *jns, char *cr_k) : Buah(nm, jns)
    {
        cout<<"//konstruktor kelas Mangga dijalankan"<<endl;
        strcpy(ciri_khusus, cr_k);
    }

    ~Mangga()
    {
        cout<<"//destruktor kelas Mangga dijalankan"<<endl;
    }

    void infoMangga()
    {
        cout<<"Ciri Khusus : "<<ciri_khusus<<endl;
    }
};

int main()
{
   //menciptakan objek "mga" sekaligus melakukan inisialisasi
melalui Konstruktor dari Kelas Mangga dan Konstruktor dari
Kelas Buah
    Mangga mga("mangga", "mangga bangkok", "buahnya manis berbentuk
lonjong");

    cout<<"--------------------"<<endl;
    cout<<"INHERITANCE PADA C++"<<endl;
    cout<<"--------------------"<<endl;

    mga.infoBuah();
    mga.infoMangga();

    cout<<"\n--------------------\n"<<endl;
    return 0;
}
```

Code program 2.1

Output program

```
//konstruktor kelas Buah dijalankan
//konstruktor kelas Mangga dijalankan

-----------------------------------
INHERITANCE PADA C++
-----------------------------------

Info  salah satu jenis buah
```

```
Nama        : Mangga
Jenis       : mangga bangkok
Ciri Khusus : buahnya manis berbentuk lonjong
---------------------------------------

//destruktor kelas Mangga dijalankan
//destruktor kelas Buah dijalankan
```

Output program 2.1

Code diatas termasuk kelas buah yang diwariskan ke kelas mangga.

Perhatikan potongan program di bawah ini dan output program,

Mangga mga("mangga", "mangga bangkok", "buahnya manis berbentuk lonjong");

Ketika objek mga diciptakan secara otomatis konstruktor akan dijalankan. Lalu bagaimana urutannya? Seperti yang terlihat pada ouput program kontruktor dari kelas dasar (Buah) dijalankan terlebih dahulu, kemudian baru konstruktor dari kelas turunan (Mangga) dijalankan. Destruktor kelas turunan (Mangga) dijalankan terlebih dahulu, kemudian baru destruktor kelas dasar (Buah) dijalankan.

Perhatikan konstruktor Mangga(char *nm, char *jns, char *cr_k) : Buah(nm, jns). Pernyataan : : Buah(nm, jns) digunakan untuk melakukan inisialisasi terhadap variabel *nama* dan *jenis* pada kelas Buah. Penginisialisasian terjadi ketika objek mga diciptakan tepatnya ketika konstruktor pada kelas Buah dijalankan. Penginisialisasian dilakukan dengan memanfaatkan nilai yang ditangkap oleh variabel pada parameter fungsi, yaitu nm dalam hal ini “Mangga” dan variabel jns dalam hal ini “mangga bangkok”.

2. Program Kedua

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

//Kelas Dasar "Buah"
class Buah
{
private:
```

```
    char nama[15];
    char jenis[15];

public:
    //konsruktor
    Buah(char *nm, char *jns)
    {
        cout<<"//konstruktor kelas Buah dijalankan"<<endl;
        strcpy(nama, nm);
        strcpy(jenis, jns);
    }

    ~Buah()
    {
        cout<<"//destruktor kelas Buah dijalankan"<<endl;
        _getche();
    }

    void infoBuah()
    {
        cout<<endl;
        cout<<"Info salah satu jenis buah"<<endl;
        cout<<"Nama       : "<<nama<<endl;
        cout<<"Jenis      : "<<jenis<<endl;
    }
};

class Jambu : public Buah
{
private:
    char ciri_khusus[30];

public:
     //konstruktor
    Jambu(char *nm, char *jns, char *cr_k) : Buah(nm, jns)
    {
        cout<<"//konstruktor kelas Jambu dijalankan"<<endl;
        strcpy(ciri_khusus, cr_k);
    }

    ~Jambu()
    {
        cout<<"//destruktor kelas Jambu dijalankan"<<endl;
    }

    void infoJambu()
    {
        cout<<"Ciri Khusus : "<<ciri_khusus<<endl;
```

```
    }
};

int main()
{
  //menciptakan objek "jmb" sekaligus melakukan inisialisasi
melalui Konstruktor dari Kelas Jambu dan Konstruktor dari
Kelas Buah
    Jambu jmb("jambu", "jambu biji", "buahnya bulat dalamnya merah muda");

    cout<<"--------------------"<<endl;
    cout<<"INHERITANCE PADA C++"<<endl;
    cout<<"--------------------"<<endl;

    mga.infoBuah();
    mga.infoJambu();

    cout<<"\n--------------------\n"<<endl;
    return 0;
}
```

Code program 2.2

Output program

```
//konstruktor kelas Buah dijalankan
//konstruktor kelas Jambu dijalankan

-----------------------------------
INHERITANCE PADA C++
-----------------------------------

Info  salah satu jenis buah
Nama          : jambu
Jenis         : jambu biji
Ciri Khusus : buahnya bulat dalamnya merah muda
---------------------------------------

//destruktor kelas Jambu dijalankan
//destruktor kelas Buah dijalankan
```

Output program 2.2

Code diatas termasuk kelas buah yang diwariskan ke kelas mangga.

Perhatikan potongan program di bawah ini dan output program, Jambu jmb("jambu", "jambu biji", "buahnya bulat dalamnya merah muda");

Ketika objek jmb diciptakan secara otomatis konstruktor akan dijalankan. Lalu bagaimana urutannya seperti yang terlihat pada ouput program kontruktor dari kelas dasar (Buah) dijalankan terlebih dahulu, kemudian baru konstruktor dari kelas turunan (Jambu) dijalankan. Destruktor kelas turunan (Jambu) dijalankan terlebih dahulu, kemudian baru destruktor kelas dasar (Buah) dijalankan.

Perhatikan konstruktor Jambu(char *nm, char *jns, char *cr_k) : Buah(nm, jns). Pernyataan : : Buah(nm, jns) digunakan untuk melakukan inisialisasi terhadap variabel *nama* dan *jenis* pada kelas Buah. Penginisialisasian terjadi ketika objek jmb diciptakan tepatnya ketika konstruktor pada kelas Buah dijalankan. Penginisialisasian dilakukan dengan memanfaatkan nilai yang ditangkap oleh variabel pada parameter fungsi, yaitu nm dalam hal ini “Jambu” dan variabel jns dalam hal ini “jambu biji”.

BAB III
KESIMPULAN

Inheritance atau Pewarisan adalah salah satu konsep pada object oriented programming, yang mengadopsi konsep pewarisan yang di miliki oleh object pada dunia nyata. Konsep Pewarisan ini memiliki model relasi “adalah”, dimana sebuah class akan dimungkinkan untuk mengambil isi dari class lain sebagai isi dari class tersebut. Konsep pewarisan akan mempermudah kita dalam membuat dan perawatan sebuah program, dengan menghilangkan “penggunaan kode berulang” saat pembuatan banyak class yang serupa. Contoh kasus yang tepat untuk penggunaan konsep pewarisan adalah ketika kita sedang membuat class macam-macam orang, dimana pertama-tama anda akan membuat sebuah class orang sebagai base class yang memiliki attribute dan method layaknya sebagai manusia pada dunia nyata, lalu base class tersebut isinya dapat di warisi ke derived classes, dimana derived merupakan macam-macam golongan dari aktivitas manusia, misalnya class pekerja, class pelajar, class pengusaha dan lain sebagainya. Dalam kasus tersebut, class dari macam-macam orang (pelajar, pengusaha, pekerja) harus mewarisi isi dari class orang, karena beberapa class tersebut memang membutuhkan identitas (attribute) dan sifat (method) dasar menjadikanya layaknya orang, dan hal itu dimiliki oleh class orang. Dengan hal ini kita tidak perlu mengetikan kode berulang kali yang mirip seperti isi dari class orang ke beberapa class yang membutuhkan isi tersebut.

DAFTAR PUSTAKA

Nblognlife. 2017, Inheritance dasar pewarisan (Tugas pewarisan C++) http://www.nblognlife.com/2017/04/c-inheritance-dasar-pewarisan.html (diakses pada : Selasa, 14 April 2020 pada pukul 18:00).

Belajarcpp. Inheritance tutorial (Materi pewarisan atau inheritance) https://www.belajarcpp.com/tutorial/cpp/inheritance/ (diakses pada : Selasa, 14 April 2020 pada pukul 18:00).

LAMPIRAN

```
//konstruktor kelas Buah dijalankan
//konstruktor kelas Mangga dijalankan

-----------------------------------
INHERITANCE PADA C++
-----------------------------------

Info  salah satu jenis buah
Nama         : Mangga
Jenis        : mangga bangkok
Ciri Khusus : buahnya manis berbentuk lonjong
----------------------------------------

//destruktor kelas Mangga dijalankan
//destruktor kelas Buah dijalankan
```

Output program 2.1

```
//konstruktor kelas Buah dijalankan
//konstruktor kelas Jambu dijalankan

-----------------------------------
INHERITANCE PADA C++
-----------------------------------

Info  salah satu jenis buah
Nama         : jambu
Jenis        : jambu biji
Ciri Khusus : buahnya bulat dalamnya merah muda
----------------------------------------

//destruktor kelas Jambu dijalankan
//destruktor kelas Buah dijalankan
```

Output program 2.2