

**LAPORAN HASIL PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN II**



**NAMA : THOMAS NURHUDA
NIM : 193010503002
KELAS : A
MODUL : III (POLIMORFISME)**

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

**LAPORAN HASIL PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN II**



Nama : Thomas Nurhuda
NIM : 193010503002
Kelas : A
Modul : III (Polimorfisme)

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

Penilai
Asisten Praktikum

Diana

BAB I

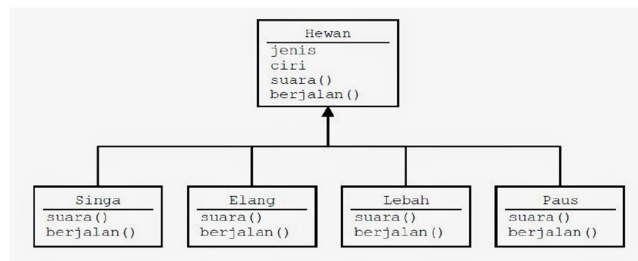
TUJUAN DAN LANDASAN TEORI

A. Tujuan

Setelah menyelesaikan modul ini, mahasiswa diharapkan mampu membuat polimorfisme.

B. Landasan Teori

Polimorfisme memiliki arti “banyak bentuk”, melakukan hal yang sama untuk berbagai data yang berbeda, mengirimkan pesan yang sama ke berbagai objek yang berbeda karena tiap objek memberi respons dengan cara yang berbeda. Berikut ini merupakan contoh polimorfisme.



Gambar 1.1. Ilustrasi Polimorfisme

Polimorfisme memiliki syarat-syarat sebagai berikut:

1. Ada hirarki pewarisan.
2. Kelas dalam hirarki pewarisan harus memiliki fungsi virtual (virtual method) dengan signature yang sama.
3. Menggunakan pointer atau rujukan ke kelas induk. Pointer digunakan untuk memanggil fungsi virtual.

Polimorfisme dapat diimplementasikan dengan menggunakan dasar function overriding (melakukan redefinisi suatu fungsi di kelas anak, fungsi yang di-override memiliki signature sama, signature sama : tipe balik, nama fungsi, parameter sama) dan pewarisan.

Suatu kelas disebut abstrak apabila memiliki minimal satu fungsi abstrak. Fungsi abstrak merupakan fungsi yang tidak memiliki definisi (hanya deklarasi fungsi)/menggunakan fungsi virtual (pure virtual).

`virtual balikan namaFungsi (parameter) = 0;`

Dalam pemrograman berorientasi objek, dalam bahasa seperti C++, dan Object Pascal, fungsi virtual atau metode virtual adalah fungsi atau metode yang dapat diwariskan dan dapat ditimpa yang memfasilitasi pengiriman dinamis. Konsep ini merupakan bagian penting dari bagian (runtime) polimorfisme pemrograman berorientasi objek (OOP). Singkatnya, fungsi virtual mendefinisikan fungsi target untuk dieksekusi, tetapi target mungkin tidak diketahui pada waktu kompilasi.

Tidak seperti fungsi non-virtual, ketika fungsi virtual ditimpa, versi yang paling diturunkan digunakan di semua tingkatan hirarki kelas, bukan hanya pada tingkat di mana ia dibuat. Oleh karena itu jika salah satu metode kelas dasar memanggil metode virtual, versi yang ditentukan dalam kelas turunan akan digunakan alih-alih versi yang ditentukan dalam kelas dasar.

Ini berbeda dengan fungsi non-virtual, yang masih dapat ditimpa dalam kelas turunan, tetapi versi "baru" hanya akan digunakan oleh kelas turunan dan di bawahnya, tetapi tidak akan mengubah fungsionalitas kelas dasar sama sekali. Sedangkan Fungsi virtual murni atau metode virtual murni adalah fungsi virtual yang harus diterapkan oleh kelas turunan jika kelas turunannya tidak abstrak.

Ketika metode virtual murni ada, kelasnya "abstrak" dan tidak bisa dipakai sendiri. Sebagai gantinya, kelas turunan yang mengimplementasikan metode virtual-virtual harus digunakan. A pure-virtual sama sekali tidak didefinisikan di kelas dasar, jadi kelas turunan harus mendefinisikannya, atau kelas turunannya juga abstrak, dan tidak bisa dipakai. Hanya kelas yang tidak memiliki metode abstrak yang dapat dipakai.

BAB II

PEMBAHASAN

1. Program Polimorfisme Class Buah

Perintah yang diberikan kali ini adalah membuat sebuah program Polimorfisme Class Buah, Dalam membuat program ini harus berdasarkan konsep atau sifat-sifat Pemrograman Berorientasi Objek (PBO). Pendeklarasian program yang akan dibuat adalah sebagai berikut.

```
PoliBuah_A.cpp
#ifndef Buah_h
#define Buah_h
using namespace std;

class Buah{
    public:
        void info();
        virtual void nBuah()=0;
        virtual void rBuah()=0;

    protected:
        char nama[20];
        char rasa [20];
};

class Manggis:public Buah{
    public:
        Manggis(char *nm, char *rs);
        void nBuah();
        void rBuah();
};

class Nanas:public Buah{
```

```

        public:
        Nanas(char *nm, char *rs);
        void nBuah();
        void rBuah();
    };
#endif

```

PoliBuah_B.cpp

```

#include "PoliBuah_A.cpp"
#include <iostream>
#include <string.h>
#include <conio.h>
using namespace std;

void Buah::info(){
    cout<<"Informasi Buah"<<endl;
}

Manggis::Manggis(char *nm, char *rs){
    strcpy(nama, nm);
    strcpy(rasa, rs);
}

void Manggis::nBuah(){
    cout<<"Nama : "<<nama<<endl;
}

void Manggis::rBuah(){
    cout<<"Rasa : "<<rasa<<endl;
}

Nanas::Nanas(char *nm, char *rs){
    strcpy(nama, nm);
    strcpy(rasa, rs);
}

```

```

}
void Nanas::nBuah(){
    cout<<"Nama : "<<nama<<endl;
}
void Nanas::rBuah(){
    cout<<"Rasa : "<<rasa<<endl;
}

```

PoliBuah_C.cpp

```

#include"PoliBuah_A.cpp"
#include<iostream>
#include<string.h>
#include<conio.h>
using namespace std;

int main(){
    Buah *obj_buah;
    Manggis mgs("Manggis","Manis");
    Nanas nns("Nanas","Asam");

    cout<<"Polimorfisme Buah"<<endl;
    cout<<"====="<<endl;
    cout<<endl;

    //menunjuk kelas manggis
    obj_buah = &mgs;
    obj_buah->info();
    obj_buah->nBuah();
    obj_buah->rBuah();
    cout<<endl;
}

```

```

        //menunjuk kelas nanas
        obj_buah = &nns;
        obj_buah->info();
        obj_buah->nBuah();
        obj_buah->rBuah();

        _getche();
        return 0;
    }

```

Pada pendeklarasian program diatas terdapat 3 buah program didalamnya yang saling berkaitan satu dengan lainnya. Oleh karena itu, untuk menghubungkan ketiga program tersebut kita dapat menggunakan Project agar program dapat berjalan sebagaimana mestinya.

Pada program PoliBuah_A.cpp terdapat preprocessor yang digunakan untuk memberikan suatu perintah pada compiler. Contohnya seperti #define yang digunakan untuk melaksanakan substitusi makro dari satu lembar teks ke lembar teks yang lain melalui suatu file dimana teks tersebut digunakan. #ifndef dan #endif merupakan instruksi logika umum, dimana jika suatu ekspresi adalah benar, maka kode yang berada ditengah-tengah kedua instruksi tersebut akan tersusun. Program ini merupakan program yang digunakan untuk mendeklarasikan Kelas Dasar Buah (Kelas Abstrak) dan sebagai berikut.

```

class Buah{
    public:
        void info();
        virtual void nBuah()=0;
        virtual void rBuah()=0;

    protected:

```



```
char nama[20];  
char rasa [20];  
};
```

Suatu kelas bisa dikatakan sebagai kelas abstrak jika didalamnya memiliki setidaknya satu buah fungsi virtual murni. Pada kelas ini nantinya tidak diperbolehkan menciptakan sebuah objek, yang diperbolehkan hanyalah menggunakan pointer. Pada akses public kelas buah terdapat fungsi biasa “void info();” yang digunakan untuk menampilkan informasi buah pada output program dan fungsi virtual murni yang diawali dengan kede blok “virtual void” dan diakhiri “=0;”. Dengan menjadikan virtual murni, kita memberitahu compiler bahwa fungsi virtual tersebut tidak memiliki body fungsi. Kemudian terdapat akses protected yang memiliki member dengan variabel nama dan rasa yang masing-masing bertipe data char dengan nilai maksimum 20.

Selanjutnya adalah pendeklarasian Kelas Turunan yaitu kelas yang mewarisi akses public dari Kelas Dasar. Pendeklarasiannya adalah sebagai berikut.

```
class Manggis:public Buah{  
    public:  
    Manggis(char *nm, char *rs);  
    void nBuah();  
    void rBuah();  
};  
class Nanas:public Buah{  
    public:  
    Nanas(char *nm, char *rs);  
    void nBuah();  
    void rBuah();  
};
```

Didalam program tersebut terdapat 2 buah kelas turunan yang diberi nama Kelas Manggis dan Kelas Nanas. Kelas turunan juga memiliki hak akses, karena kelas diatas menurunkan hak akses public dari kelas induknya, maka member pada akses public kelas induk harus dimiliki oleh kelas turunan. Pada kelas turunan tersebut terdapat Konstruktor yang digunakan pada saat ingin memasukkan data didalam program sehingga data bisa terbaca oleh output program nantinya. Salah satu contoh konstruktor pada program diatas adalah kode “Manggis(char *nm, char *rs);”. Untuk mengakhiri program yang pertama ini digunakan preprocessor #endif.

Program PoliBuah_B.cpp digunakan untuk mendeklarasikan fungsi-fungsi dari kelas dasar dan kelas turunan yang akan dijalankan nantinya. Sebelum mendeklarasikan fungsi tersebut, seorang programmer harus membuat File Header terlebih dahulu, dimana nantinya File Header ini digunakan untuk memanggil library-library yang ada sehingga suatu fungsi dapat digunakan secara baik dan benar. #include digunakan untuk mendeklarasikan File Header didalam bahas pemrograman C++. File Header yang digunakan pada program ini adalah <iostream> yang digunakan untuk menampilkan perintah cin,cout dan endl. <conio.h> digunakan untuk menampilkan perintah getch dan clrscr. Dan <string.h> digunakan untuk menampilkan perintah strcpy. Serta terdapat kode (#include “PoliBuah_A.cpp”) yang digunakan untuk menghubungkan program ini ke program PoliBuah_A sebelumnya. Kode using namespace std; berfungsi untuk memanggil Class/Object/Fungsi yang terdapat didalam namespace tersebut. Pendeklarasian file header pada program ini adalah sebagai berikut.

```
#include "PoliBuah_A.cpp"  
#include <iostream>  
#include <string.h>  
#include <conio.h>  
using namespace std;
```

Selanjutnya terdapat fungsi sebagai berikut.

```
void Buah::info(){  
    cout<<"Informasi Buah"<<endl;  
}
```

Fungsi diatas merupakan pendeklarasian dari fungsi info pada kelas Buah. Fungsi ini digunakan untuk menampilkan kalimat Informasi Buah pada output program nantinya. Blok program berikutnya adalah pendeklarasian fungsi yang terdapat pada kelas Manggis.

```
Manggis::Manggis(char *nm, char *rs){  
    strcpy(nama, nm);  
    strcpy(rasa, rs);  
}
```

Fungsi diatas merupakan pernyataan Manggis = Manggis yang berfungsi untuk memberikan insialisasi terhadap variabel input Manggis ke variabel Manggis. Inisialisasi ini dilakukan agar nantinya kita bisa memasukkan data didalam program sehingga data bisa ditampilkan pada output program. “strcpy” digunakan untuk menyalin string asal ke variabel tujuan dengan syarat string tujuan harus mempunyai tipe data dan ukuran yang sama dengan string asal. “(nama, nm)” merupakan variabel dari pointer nm, dan “(rasa, rs)” merupakan variabel dari pointer rs. Variabel ini berfungsi sebagai alamat atau memori pemanggilan saat kita menginputkan data.

Selanjutnya terdapat fungsi sebagai berikut.

```
void Manggis::nBuah(){  
    cout<<"Nama : "<<nama<<endl;  
}
```

```
void Manggis::rBuah(){  
    cout<<"Rasa : "<<rasa<<endl;  
}
```

Fungsi diatas merupakan pendeklarasian kelas Manggis yang akan mengakses Fungsi virtual murni dari kelas dasar Buah. Kode blok (cout<<"Nama : "<<nama<<endl;) digunakan untuk menampilkan data variabel nama dari fungsi virtual murni pada output program. Kode blok (cout<<"Rasa : "<<rasa<<endl;) digunakan untuk menampilkan data variabel rasa dari fungsi virtual murni pada output program.

Blok program berikutnya adalah pendeklarasian fungsi yang terdapat pada kelas Nanas sebagai berikut.

```
Nanas::Nanas(char *nm, char *rs){  
    strcpy(nama, nm);  
    strcpy(rasa, rs);  
}
```

Fungsi diatas merupakan pernyataan Nanas = Nanas yang berfungsi untuk memberikan insialisasi terhadap variabel input Nanas ke variabel Nanas. Inisialisasi ini dilakukan agar nantinya kita bisa memasukkan data didalam program sehingga data bisa ditampilkan pada output program. "strcpy" digunakan untuk menyalin string asal ke variabel tujuan dengan syarat string tujuan harus mempunyai tipe data dan ukuran yang sama dengan string asal. "(nama, nm)" merupakan variabel dari pointer nm, dan "(rasa, rs)" merupakan variabel dari pointer rs. Variabel ini berfungsi sebagai alamat atau memori pemanggilan saat kita menginputkan data. Untuk menginsialisasikan data pada kelas Nanas dapat dilakukan pada blok program sebagai berikut.

```

void Nanas::nBuah(){
    cout<<"Nama : "<<nama<<endl;
}
void Nanas::rBuah(){
    cout<<"Rasa : "<<rasa<<endl;
}

```

Fungsi diatas merupakan pendeklarasian kelas Nanas yang akan mengakses Fungsi virtual murni dari kelas dasar Buah. Kode blok (cout<<"Nama : "<<nama<<endl;) digunakan untuk menampilkan data variabel nama dari fungsi virtual murni pada output program. Kode blok (cout<<"Rasa : "<<rasa<<endl;) digunakan untuk menampilkan data variabel rasa dari fungsi virtual murni pada output program.

Program berikutnya adalah PoliBuah_C.cpp, karena program ini terpisah dari program sebelumnya maka dalam pembuatannya juga diperlukan file header seperti program PoliBuah_B.cpp, deklarasinya adalah sebagai berikut.

```

#include"PoliBuah_A.cpp"
#include<iostream>
#include<string.h>
#include<conio.h>
using namespace std;

```

File Header ini digunakan untuk memanggil library-library yang ada sehingga suatu fungsi dapat digunakan secara baik dan benar. #include digunakan untuk mendeklarasikan File Header didalam bahas pemrograman C++. File Header yang digunakan pada program ini adalah <iostream> yang digunakan untuk menampilkan perintah cin,cout dan endl. <conio.h> digunakan untuk menampilkan perintah getch dan clrscr. Dan <string.h> digunakan untuk menampilkan perintah strcpy.

Kemudian juga terdapat kode (`#include "PoliBuah_A.cpp"`) yang digunakan untuk menghubungkan program ini ke program `PoliBuah_A` sebelumnya. Kode `using namespace std;` berfungsi untuk memanggil Class/Object/Fungsi yang terdapat didalam namespace tersebut.

Pada program ini berisi fungsi `main` atau `Int main()` yang merupakan fungsi utama dalam program. Fungsi ini akan dieksekusi pertama kali saat program dijalankan. Oleh karena itu, kita harus menuliskan logika program di dalam fungsi ini. Maksud dari kata `Int` didepan `main` adalah tipe data yang akan dikembalikan. Maka didalam fungsi `main()`, wajib kita sertakan kode `return 0` yang berarti fungsi `main` akan mengembalikan nilai 0 setelah dieksekusi. Pada program C++ terdapat Blok kode yang merupakan kumpulan statement atau ekspresi yang dibungkus dengan kurung kurawal atau `{...}`. Kode blok yang terdapat didalam fungsi ini adalah sebagai berikut.

```
Buah *obj_buah;  
Manggis mgs("Manggis","Manis");  
Nanas nns("Nanas","Asam");
```

Karena Kelas Dasar Buah merupakan kelas abstrak maka pembentukan objek tidak boleh dilakukan. Oleh karena itu, maka dibuat suatu pointer dengan nama `obj_buah`. Pada kelas `Manggis` terdapat objek `mgs` yang berisi data “Manggis” sebagai nilai untuk variabel nama dan data “Manis” sebagai nilai untuk variabel rasa pada kelas `Manggis`. Sedangkan pada kelas `Nanas` terdapat objek `nns` yang berisi data “Nanas” sebagai nilai untuk variabel nama dan data “Asam” sebagai nilai untuk variabel rasa pada kelas `Nanas`.

Kode program berikutnya adalah perintah yang digunakan untuk menunjuk objek `mgs` pada kelas `Manggis` sebagai berikut.

```
obj_buah = &mgs;  
obj_buah->info();
```

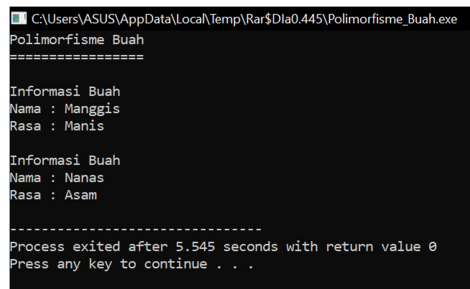
```
obj_buah->nBuah();  
obj_buah->rBuah();
```

Kode “obj_buah = &mgs;” berarti pointer obj_buah diambil alih oleh objek mgs sehingga nantinya fungsi yang dijalankan berasal dari kelas Manggis. Data yang akan ditampilkan nantinya adalah Informasi Buah, nama Buah, serta rasa Buah. Kode program berikutnya adalah perintah yang digunakan untuk menunjuk objek nns pada kelas Nanas sebagai berikut.

```
obj_buah = &nns;  
obj_buah->info();  
obj_buah->nBuah();  
obj_buah->rBuah();
```

Kode “obj_buah = &nns;” berarti pointer obj_buah diambil alih oleh objek nns sehingga nantinya fungsi yang dijalankan berasal dari kelas Nanas. Data yang akan ditampilkan nantinya adalah Informasi Buah, nama Buah, serta rasa Buah.

Pada bagian akhir program terdapat kode _getche(); yang merupakan kepanjangan dari Get Character And Echo yang digunakan untuk menghentikan suatu proses yang berjalan. Dan kode return 0; yang digunakan untuk mengakhiri sebuah program. Ketika program berhasil dijalankan, maka output yang akan ditampilkan adalah sebagai berikut.



```
C:\Users\ASUS\AppData\Local\Temp\Rar$Dla0.445\Polimorfisme_Buah.exe  
Polimorfisme Buah  
=====  
Informasi Buah  
Nama : Manggis  
Rasa : Manis  
  
Informasi Buah  
Nama : Nanas  
Rasa : Asam  
  
-----  
Process exited after 5.545 seconds with return value 0  
Press any key to continue . . .
```

Gambar 2.1. Output Program Polimorfisme Buah

BAB III

KESIMPULAN

Dalam membuat sebuah program polimorfisme menggunakan prinsip pemrograman berorientasi objek (PBO) sebaiknya menggunakan project agar beberapa program yang telah dibuat bisa berkaitan satu dengan yang lainnya. Selain itu pada kelas Dasar program harus terdapat fungsi virtual yang menandakan suatu program tersebut merupakan sebuah polimorfisme. Jika didalam suatu kelas dasar terdapat fungsi virtual murni, maka kelas tersebut dinamakan kelas abstrak. Didalam kelas abstrak tidak dapat dibuat sebuah objek. Oleh karena itu, untuk memudahkan programmer maka dibuatlah suatu pointer. Fungsi virtual dari kelas Dasar inilah yang nanti nya di turunkan pada kelas Turunan.

DAFTAR PUSTAKA

- Aditya Rizki. 2011. *Tutorial Pemrograman Berorientasi Obyek dengan C++*.
<https://adityarizki.net/tutorial-pemrograman-berorientasi-obyek-dengan-c-polimorfisme-studi-kasus/> (Diakses pada 21 April 2020, pukul 23.15 WIB)
- Ahmad muahrdian. 2019. *Fungsi Int Main*. <https://www.petanikode.com/c-sintak/>. (Diakses pada 21 April 2020, pukul 20.15 WIB)
- Dosen Teknik Informatika. 2020. *Modul Praktikum Struktur Data*. Print Pdf.
- Rachmat Santoso. 2017. *Polimorfisme Pada C++*.
<http://www.nblognlife.com/2017/07/polimorfisme-pada-c.html>
(Diakses pada 21 April 2020, pukul 21.00 WIB)
- Yanuardi. 2018. *Macam-Macam Perintah atau Header pada C++*.
<https://www.forumkomputer.com/2018/10/perintah-atau-header-pada-c.html> (Diakses pada 20 April 2020, pukul 14.20 WIB)

LAMPIRAN

```
1 #ifndef Buah_h
2 #define Buah_h
3 using namespace std;
4
5 class Buah {
6 public:
7     void info();
8     virtual void nBuah()=0;
9     virtual void rBuah()=0;
10
11 protected:
12     char nama[20];
13     char rasa[20];
14 };
15
16 class Manggis:public Buah{
17 public:
18     Manggis(char *nm, char *rs);
19     void nBuah();
20     void rBuah();
21 };
22
23 class Nanas:public Buah{
24 public:
25     Nanas(char *nm, char *rs);
26     void nBuah();
27     void rBuah();
28 };
29
30 #endif
```

Gambar 1. Program PoliBuah_A.cpp

```
1 #include "PoliBuah_A.cpp"
2 #include <iostream>
3 #include <string.h>
4 #include <conio.h>
5 using namespace std;
6
7 void Buah::info(){
8     cout<<"Informasi Buah"<<endl;
9 }
10 Manggis::Manggis(char *nm, char *rs){
11     strcpy(nama, nm);
12     strcpy(rasa, rs);
13 }
14
15 void Manggis::nBuah(){
16     cout<<"Nama : "<<nama<<endl;
17 }
18 void Manggis::rBuah(){
19     cout<<"Rasa : "<<rasa<<endl;
20 }
21 Nanas::Nanas(char *nm, char *rs){
22     strcpy(nama, nm);
23     strcpy(rasa, rs);
24 }
25
26 void Nanas::nBuah(){
27     cout<<"Nama : "<<nama<<endl;
28 }
29 void Nanas::rBuah(){
30     cout<<"Rasa : "<<rasa<<endl;
31 }
```

Gambar 2. Program PoliBuah_B.cpp

```

1  #include "PoliBuah_A.cpp"
2  #include <iostream>
3  #include <string.h>
4  #include <conio.h>
5  using namespace std;
6
7  int main() {
8      Buah *obj_buah;
9      Manggis mgs("Manggis", "Manis");
10     Nanas nns("Nanas", "Asam");
11
12     cout << "Polimorfisme Buah" << endl;
13     cout << "=====" << endl;
14     cout << endl;
15
16     // menunjuk kelas manggis
17     obj_buah = &mgs;
18     obj_buah->info();
19     obj_buah->nBuah();
20     obj_buah->rBuah();
21     cout << endl;
22
23     // menunjuk kelas nanas
24     obj_buah = &nns;
25     obj_buah->info();
26     obj_buah->nBuah();
27     obj_buah->rBuah();
28
29     _getche();
30     return 0;
31 }

```

Gambar 3. Program PoliBuah_C.cpp

```

C:\Users\ASUS\AppData\Local\Temp\Rar$DIa0.445\Polimorfisme_Buah.exe
Polimorfisme Buah
=====

Informasi Buah
Nama : Manggis
Rasa : Manis

Informasi Buah
Nama : Nanas
Rasa : Asam

-----
Process exited after 5.545 seconds with return value 0
Press any key to continue . . .

```

Gambar 4. Output Program Polimorfisme Buah