LAPORAN HASIL PRAKTIKUM
ALGORITMA DAN PEMROGRAMAN II

Nama : **OKKI ANDARESTA**
NIM : **DBC 118 068**
Kelas : **A**
Modul : **III (POLIMORFISME)**

JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNIK
UNIVERSITAS PALANGKA RAYA
2020

LAPORAN HASIL PRAKTIKUM

ALGORITMA DAN PEMROGRAMAN II

Nama : Okki Andaresta

NIM : DBC 118 068

Kelas : A

Modul : POLIMORFISME

Komposisi	MAX	Nilai
BAB I Tujuan dan Landasan Teori	10	8
BAB II Pembahasan	60	49
BAB III Kesimpulan	20	13
Daftar Pustaka	5	5
Lampiran	5	5
Jumlah	100	

Penilai

Asisten Praktikum

Diana

BAB I
TUJUAN DAN LANDASAN TEORI

1.1 TUJUAN

Setelah menyelesaikan modul ini, mahasiswa diharapkan mampu membuat Polimorfisme.

1.2 DASAR TEORI

Polimorfisme berasal dari bahasa Yunani yang berarti banyak bentuk. Dalam PBO, konsep ini memungkinkan digunakannya suatu interface yang sama untuk memerintah objek agar melakukan aksi atau tindakan yang mungkin secara prinsip sama namun secara proses berbeda.

Polimorfisme merupakan kemampuan suatu method untuk bekerja dengan lebih dari satu tipe argumen. Pada bahasa lain (khususnya C++), konsep ini sering disebut dengan method overloading. Pada dasarnya, Python tidak menangani hal ini secara khusus. Hal ini disebabkan karena Python merupakan suatu bahasa pemrograman yang bersifat dynamic typing yaitu tidak memerlukan deklarasi tipe.

Polimorfsme adalah suatu object dapat memiliki berbagai bentuk, sebagai object dari class sendiri atau object dari superclassnya.

Keuntungan pemograman dengan menggunakan Polimorfisme, yaitu :

1. Dapat menggunakan kelas-kelas yang kita buat (sebagai super kelas) dan membuat kelas kelas baru berdasar superkelas tersebut dengan karakteristik yang lebih khusus dari behaviour umum yang dimiliki superkelas.
2. Dapat membuat super kelas yang hanya mendefinisikan behaviuor namun tidak memberikan implementasidari metode-metode yang ada.Hal ini berguna jika ingin membuat template kelas, kelas ini disebut kelas abstrak karena behaviournya masih abstrak dan belum diimplementasikan. Subkelas-subkelas dari kelas ini yang disebut kelas konkret, mengimplementasikan behaviuor abstrak tersebut sesuai dengan kebutuhan masing-masing.
3. Menghindari duplikasi object, yang dapat menciptakan class baru dari class yang sudah ada, sehingga tidak perlu menuliskan code dari nol ataupun

mengulangnya, namun tetap bisa menambahkan attribute dan atau method unik dari class itu sendiri. Dalam konsep yang lebih umum sering kali polimorfisme disebut dalam istilah satu interface banyak aksi.

Polimorfisme dapat berarti banyak bentuk, maksudnya yaitu dapat menimpa (override), suatu method, yang berasal dari parent class (super class) dimana object tersebut diturunkan, sehingga memiliki kelakuan yang berbeda.

Pada dasarnya ada 2 tipe polimorfisme, yaitu :

1. Static atau trivialmerupakan, function overloading (penggunaan kembali nama fungi yang sama tapi dengan argumen yang berbeda) yang terbagi dalam 3 signature yaitu :

a. Jenis Array
b. Letak Array
c. Type Array

Contoh function overloading :

Void tambah (int a, int b);

Void tambah (float d, float c);

2. Dynamic atau trueMerupakan function overriding (sebuah fungsi dalam class turunan yang memiliki nama, return type argumen function yang sama dengan fungsi dalam class induk). Menggunakan virtual method.

Pure virtual method (tanpa function body)

Contoh : virtual void jalan() = 0;

Squasi virtual method (ada function body)

Contoh : virtual void info() {;}

BAB II
PEMBAHASAN

2.1 Program pertama

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

using namespace std;

//Kelas Dasar Domba
class Domba{
protected:
    char nama[20];
public:
    //fungsi virtual murni
    virtual void info_tanduk() = 0;
};

//kelas Argali turunan dari kelas Domba
class Argali : public Domba{
public:
    Argali(char* nm) {
        strcpy(nama, nm);
    }
    void info_tanduk(){
        cout<<"Domba "<<nama<<"  memiliki tanduk";
    }
};

//kelas Domestik turunan dari kelas Domba
```

```
class Domestik : public Domba{
public:
    Domestik(char* nm){
        strcpy(nama, nm);
    }
    void info_tanduk(){
        cout<<"Domba "<<nama<<" tidak memiliki tanduk";
    }
};
int main(){
    //deklarasi objek
    Domba *obj_domba;
    Argali agl("Argali");
    Domestik dtk("Domestik");

    cout<<"POLIMORFISME "<<endl;
    cout<<"--------------"<<endl;

    //menunjuk ke objek dari kelas Argali
    obj_domba = &agl;
    obj_domba->info_tanduk();

    cout<<endl;

    //menunjuk ke objek dari kelas Domestik
    obj_domba = &dtk;
    obj_domba->info_tanduk();

    getche();
    return 0;
}
```

Penjelasan

```
//Kelas Dasar Domba
class Domba{
protected:
    char nama[20];
public:
    //fungsi virtual murni
    virtual void info_tanduk() = 0;
};

//kelas Argali turunan dari kelas Domba
class Argali : public Domba{
public:
    Argali(char* nm) {
        strcpy(nama, nm);
    }
    void info_tanduk(){
        cout<<"Domba "<<nama<<"  memiliki tanduk";
    }
};
//kelas Domestik turunan dari kelas Domba
class Domestik : public Domba
public:
    Domestik(char* nm){
        strcpy(nama, nm);
    }
    void info_tanduk(){
        cout<<"Domba "<<nama<<" tidak memiliki tanduk";
    }
};
```

Code diatas menggunakan fungsi virtual murni yang dideklarasikan menggunakan kata virtual dalam fungsi kelas dasar. Ketika kelas dasar Domba fungsi virtual didefinisikan kembali dalam kelas turunan, ia memberitahu compiler tidak statis terkait dengan fungsi. fungsi virtual menjadi fungsi virtual menambahkan = 0 dan menghilangkan body fungsi, pada kode diatas, yaitu virtual void info tanduk() = 0; terdapat 2 kelas turunan dari domba yaitu kelas Argali dan Domestik, yang berlaku fungsi virtual pada murni adalah dengan kedua kelas turunannya.

```
int main(){
    //deklarasi objek
    Domba *obj_domba;
    Argali agl("Argali");
    Domestik dtk("Domestik");

    cout<<"POLIMORFISME "<<endl;
    cout<<"--------------"<<endl;

    //menunjuk ke objek dari kelas Argali
    obj_domba = &agl;
    obj_domba->info_tanduk();

    cout<<endl;

    //menunjuk ke objek dari kelas Domestik
    obj_domba = &dtk;
    obj_domba->info_tanduk();

    getche();
    return 0;
}
```

Pada fungsi main(), terdapat deklarasi pointer bertipe kelas dasar,yaitu kelas domba yang digunakan sebagai penunjuk objek kelas Argali dan juga kelas Domestik. Penunjuk objek kelas Argali,yaitu Argali agl("Argali") digunakan untuk mengakses member kelas Argali dan penunjuk objek kelas Domestik,yaitu Domestik dtk("Domestik") juga digunakan untuk mengakses member kelas Domestik. Akses member info bisa pada kedua kelas tersebut masih menggunakan objek dari kelas keduannya sendiri. mekanisme ini membuat member info_tanduk() pada kedua kelas tersbut dapat dikenali oleh pointer bertipe kelas dasar. Hal ini dimungkinkan karena adanya virtual function yang membuat pointer lebih memilih objek yang ditunjuk daripada kelas dasar yang diwakili.

Berikut adalah output program yang didapatkan:

Gambar 2.1 program pertama

BAB III
KESIMPULAN

Konstruksi polimorfisme dalam pemrograman berorientasi obyek memungkinkan untuk mengadakan ikatan dinamis(juga disebut ikatan tunda, atau ikatan akhir). Apabila fungsi-fungsi dari suatu kelas dasar didefinisikan ulang atau ditindih pada kelas turunan, maka obyek-obyek yang dihasilkan hirarki kelas berupa obyek polimorfik. Polimorfik artinya mempunyai banyak bentuk atau punya kemampuan untuk mendefinisi banyak bentuk. Apabila terdapat member khusus yang apabila dideklarasikan di kelas induk, dapat dideklarasikan lagi di kelas anak-anaknya, dengan kegunaan fungsi yang berbeda pada masing-masing kelas. Member ini disebut sebagai virtual member yang hanya dideklarasikan sebagai virtual pada kelas induk.

DAFTAR PUSTAKA

Dosen Teknik Informatika. Algoritma dan Pemprograman I. 2020. *Modul Praktikum Algoritma dan Pemprograman II*. Universitas Palangka Raya. Fakultas Teknik. Jurusan Teknik Informatika.

Andri, Kristanto. 2009. *Algoritma & Pemprograman dengan C++*. Yogyakarta: Graha Ilmu.

Budi, Raharjo.2010. *Pemprograman C++*. Bandung: Informatika.

Evaristus, Didik. 2017. *Polimorfisme Pada C++* https://sis.binus.ac.id/2017/08/30/polimorphism/ (diakses pada : Selasa, 21 April 2020 pada pukul: 23.10)

LAMPIRAN

Gambar 2.1 program pertama