

1 D4 - TEKKOM B

PRAKTIKUM 8 POLYMORPHISM



Nama	:	Septian Bagus Jumanoro
Kelas	:	1 – D4 Teknik Komputer B
NRP	:	3221600039
Dosen	:	Dr Bima Sena Bayu Dewantara S.ST, MT.
Mata Kuliah	:	Praktikum Pemrograman Dasar 2
Hari/Tgl. Praktikum	:	Rabu, 11 Mei 2022



1. Percobaan 1 –Polymorphism

```
#include <iostream>

using namespace std;

class Shape{
protected:
    int width, height;

public:
    Shape(int a=0, int b=0){
        width = a;
        height = b;
    }

    int area(){
        cout << "Parent class area: " << endl;
        return 0;
    }
};

class Rectangle: public Shape{
public:
    Rectangle(int a=0, int b=0): Shape(a,b){}

    int area(){
        cout << "Rectangle class area: " << endl;
        return(width * height);
    }
};

class Triangle: public Shape{
public:
    Triangle(int a=0, int b=0): Shape(a,b){}

    int area(){
        cout << "Triangle class area: " << endl;
        return(width * height / 2);
    }
};

int main(){
    system("cls");
    Shape *shape;
    Rectangle rec(10,7);
    Triangle tri(10,5);

    //store the address of Rectangle
```

```

    shape = &rec;

    //call rectangle area
    shape -> area();

    //store the address of Triangle
    shape = &tri;

    //call triangle area
    shape -> area();
    return 0;
}

```

Output

```

... Code + - [ ] [X] < X
Parent class area:
Parent class area:
PS D:\Kuliah\SMT 2\PD 2\Prak8> 

```

Kesimpulan

Berdasarkan program tersebut, polymorphism merupakan sebuah fitur yang biasa digunakan untuk memberikan kemampuan pada fungsi dan anggota class, untuk memberikan seakan fungsi tersebut memiliki berbagai bentuk yang sesuai berdasarkan class tersebut. Ketika menggunakan polymorphism dan mencoba memanggil sebuah anggota fungsi, maka program akan merespon anggota fungsi yang sesuai berdasarkan yang ada pada object dan class yang digunakan, tetapi karena tidak menggunakan fitur Polymorphism, maka fungsi area() tidak dapat diwakilkan dengan fungsi yang dimiliki oleh tiap class.

2. Percobaan 2 – Virtual Function

```

#include <iostream>

using namespace std;

class Shape{
protected:
    int width, height;

public:
    Shape(int a=0, int b=0){
        width = a;

```

```

        height = b;
    }

    virtual int area(){
        cout << "Parent class area: " << endl;
        return 0;
    }
};

class Rectangle: public Shape{
public:
    Rectangle(int a=0, int b=0): Shape(a,b){}

    int area(){
        cout << "Rectangle class area: " << endl;
        return(width * height);
    }
};

class Triangle: public Shape{
public:
    Triangle(int a=0, int b=0): Shape(a,b){}

    int area(){
        cout << "Triangle class area: " << endl;
        return(width * height / 2);
    }
};

int main(){
    system("cls");
    Shape *shape;
    Rectangle rec(10,7);
    Triangle tri(10,5);

    //store the address of Rectangle
    shape = &rec;

    //call rectangle area
    shape -> area();

    //store the address of Triangle
    shape = &tri;

    //call triangle area
    shape -> area();
    return 0;
}

```

Output

```
... Code + v [ ] [ ] < X
Rectangle class area:
Triangle class area:
PS D:\Kuliah\SMT 2\PD 2\Prak8> [ ]
```

Kesimpulan

Berdasarkan program tersebut, fungsi yang digunakan hampir sama dengan nomor 1 hanya berbeda pada fungsi yang pertama yaitu shape yang mendapat tambahan virtual int area(). Untuk Virtual fungsi merupakan sebuah anggota fungsi yang didirikan di dalam base class dan overridden oleh derived class. Ketika menunjuk suatu object yang dibuat dari derived class menggunakan sebuah object pointer yang dibuat menggunakan base class, kita dapat memanggil sebuah fungsi yang dibuat sebagai virtual function. Setiap pemanggilan fungsi, akan mengeksekusi tiap fungsi berdasarkan yang dimiliki oleh class tersebut.

3. Percobaan 3 – Pure Virtual Function

```
#include <iostream>

using namespace std;

class Shape{
protected:
int width, height;

public:
Shape(int a=0, int b=0){
width = a;
height = b;
}

virtual int area() = 0;
};

class Rectangle: public Shape{
public:
Rectangle(int a=0, int b=0): Shape(a,b){}

int area(){
cout << "Rectangle class area: " << endl;
return(width * height);
}
};
```

```

class Triangle: public Shape{
public:
    Triangle(int a=0, int b=0): Shape(a,b){}

    int area(){
        cout << "Triangle class area: " << endl;
        return(width * height / 2);
    }
};

int main(){
    system("cls");
    Shape *shape;
    Rectangle rec(10,7);
    Triangle tri(10,5);

    //store the address of Rectangle
    shape = &rec;

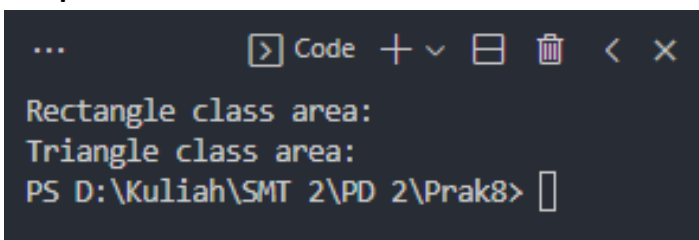
    //call rectangle area
    shape -> area();

    //store the address of Triangle
    shape = &tri;

    //call triangle area
    shape -> area();
    return 0;
}

```

Output



```

... Code + - [ ] [X] < X
Rectangle class area:
Triangle class area:
PS D:\Kuliah\SMT 2\PD 2\Prak8>

```

Kesimpulan

Berdasarkan program tersebut sebuah fungsi virtual merupakan fungsi dalam base class yang telah dideklarasikan menggunakan kata kunci 'Virtual'. Jika ingin membuat kemampuan Polymorphism pada class-class lalu beranggapan bahwa virtual function yang berada di base class sebenarnya tidak membutuhkan definisi, maka dari itu solusinya dapat menggunakan pure virtual function.