Laporan Sederhana Tugas Pemrograman Lanjut

Disusun Oleh :

Nama	: Sendy Prismana Nurferian
NRP	: 5024211012
Departemen	: Teknik Komputer
Kelas	: (A) Pemrograman Lanjut
Dosen	: Bapak Reza Fuad Rachmadi
Tugas	: Laporan Sederhana Program Operator Overloading Persegi Panjang

INSTITUT TEKNOLOGI SEPULUH NOPEMBER SURABAYA

TEKNIK KOMPUTER

2022

Nama Github : M4$ $3ÑĐÝ (SendyPrismanaNurferian)

Github : https://github.com/SendyPrismanaNurferian

Repo : https://github.com/SendyPrismanaNurferian/Prolan-Operator-Overloading-Rectangle

PEMBAHASAN PROGRAM

Tugas program “Persegi Panjang” ini adalah bentuk program yang digunakan untuk latihan *Operator Overloading*. *Operator Overloading* dalam pemrograman komputer, kelebihan beban operator, kadang-kadang disebut polimorfisme ad hoc operator, adalah kasus polimorfisme tertentu, di mana operator yang berbeda memiliki implementasi yang berbeda tergantung pada argumen mereka. Overloading operator umumnya didefinisikan oleh bahasa pemrograman, programmer, atau keduanya. Pada penjelasan singkatnya fitur overloading pada bahasa pemrograman C++ memungkinkan kita untuk mendefinisikan ulang fungsi atau kegunaan dari sebuah operator. Operator yang digunakan dalam overloading ini ada (+, -, =, dll) dalam bahasa C++ ini. Dalam tugas ini, Bapak Fuad meminya mahasiswa bisa menggunakan Operator Overloading untuk membuat program yang mampu menjumlahkan dan mengurangi luas dari 2 buah Persegi Panjang yang beririsan, dan juga memperbesar maupun memperkecil ukuran dari suatu Persegi Panjang tanpa mengubah titik tengahnya. Persegi Panjang kali ini sudah memiliki atribut yang digunakan, yaitu titik X minimal, titk X maksimal, titik Y minimal, dan titik Y maksimal. Atribut tadi bergunan untuk mempresentasikan titik maksimal dan minimal dalam bidang *cartersius* untuk sebuah Persegi Panjang.

Pada program ini saya memulainya dengan membuat class “*PersegiPanjang*” yang memiliki atribut anggota (*member atribute*) yaitu *xmin*, *xmax*, *ymax*, *ymin* seperti yang telah dijelaskan di paragraf awal tadi. Kemudian saya melakukan inisiasi *constructor* class persegi_panjang yang memiliki parameter berupa titik tengah Persegi Panjang dengan simbol (T_tengah_*) dengan maksud * ini adalah di isi atribut (x dan y), panjang Persegi Panjang dengan simbol (Panjang) dan lebar Persegi Panjang (Lebar). Lalu untuk tipe data yang saya gunakan untuk parameter tersebut menggunakan “*float*”. Tujuan saya agar nilai dari parameter tersebut adalah berupa angka desimal, dan program bisa tetap membaca sampai bilangan dibelakang koma. Dan dalam program saya beri “system (clear)” agar membersihkan terminal atau konsol yang digunakan setiap program berjalan.

PersegiPanjang.hpp

Prolan-Operator-Overloading-Rectangle > include > PersegiPanjang.hpp > ...

```hpp
#ifndef __PERSEGIPANJANG_HPP__
#define __PERSEGIPANJANG_HPP__
#include <iostream>

class PersegiPanjang
{
    private :
        float xmin,xmax,ymin,ymax;

    public :
        PersegiPanjang();
        PersegiPanjang(float Ttengah_x, float Ttengah_y, float pjg, float lbr);
        PersegiPanjang operator+(PersegiPanjang const &);
        PersegiPanjang operator-(PersegiPanjang const &);
        bool operator==(PersegiPanjang const &) const;
        float operator[](int huha);
        void operator ++();
        void operator --();
        void operator --(int);
        void operator ++(int);
        void printresult();

};

#endif
```

Gambar 1. Class di file hpp berupa persegi_panjang dan parameter maupun atributnya

Setelah selesai membuat class maka saya selanjutnya melanjutkan atribute yang sudah dibuat oleh Bapak Fuad untuk dimasukkan ke dalam program saya dengan method sebagai berikut:

```cpp
PersegiPanjang::PersegiPanjang(){}
//Cara mencari dan menentukan xmin,xmax dan ymin,ymax
PersegiPanjang::PersegiPanjang(float Ttengah_x,float Ttengah_y,float pjg,float lbr)
{
    this->xmin = Ttengah_x - (pjg/2);
    this->xmax = Ttengah_x + (pjg/2);
    this->ymin = Ttengah_y - (lbr/2);
    this->ymax = Ttengah_y + (lbr/2);
}
```

Gambar 2. Pemberian Atribute yang ada di dalam program ini

Setelah selesai saya lanjutkan untuk membuat operator overloading dari operator (+). Ketika operator (+) ini digunakan maka program akan menjumlahkan luas dari kedua buah Persegi Panjang. Akan tetapi, program bisa menjalankan jika kedua buah Persegi Panjang tadi saling beririsan satu sama lain. Maka dari itu agar program bisa berjalan maka dalam algortima nya dibuatlah *method bool* operator "==". Operator overloading "==" ini berfungsi untuk

memeriksa apakah kedua buah Persegi Panjang ini saling beririsan atau tidak. Berikut code saya:

```
//Logika untuk mengetahui apakah kedua persegi panjang bersifat beririsan atau tidak
bool PersegiPanjang::operator==(PersegiPanjang const &yoi)const
{
    if (this->xmax > yoi.xmin && this->xmin < yoi.xmax && this->ymax > yoi.ymin && this->ymin < yoi.ymax)
    return true;
    else
    return false;
}
```

Gambar 3. Logika dari method Operator Overloading "=="

Dari code di gambar 2 terdapat ***const***. *Const* disini digunakan untuk memastikan bahwa fungsi yang telah dibuat tidak bisa merubah nilai apapun itu. Karena disini pengujian dilakukan terhadap variable – variable yang digunakan, sementara variable – variable yang digunakan tidak perlu mengubah nilai apapun dari variable – variable tersebut. Setelah selesai di method tadi, saya lanjutkan pada operator overloading (+) dengan menggunakan method seperti berikut:

```
//Operator (+) berfungsi untuk menambah luasan persegi panjang dengan menggabungkan kedua luasan persegi panjang tadi
PersegiPanjang PersegiPanjang::operator+(PersegiPanjang const &yoi)
{
    PersegiPanjang temp(0,0,0,0);

    if (*this == yoi)//Berfungsi penunjuk nilai dari variable persegi dan syarat yang dipenuhi (terdapat di line command no 16)
    {
        // Untuk mengetahui update dari x
        temp.xmin = min(this-> xmin,yoi.xmin);
        temp.xmax = max(this-> xmax,yoi.xmax);
        // Untuk mengetahui update dari y
        temp.ymin = min(this-> ymin,yoi.ymin);
        temp.ymax = max(this-> ymax,yoi.ymax);

    }else cout << "Persegi Panjang 1 dam 2 Tidak Beririsan, Persegi Panjang tidak bisa lanjut ke pertambahan";
        return temp;
}
```

Gambar 4. Method di Operator Overloading (+)

Dari gambar diatas nilai atau besaran dalam atribut xmin dari Persegi Panjang yang baru (gabungan) merupakan xmin terkecil dari kedua buah Persegi Panjang. Sedangkan nilai atau besaran dalam atribut xmax dari Persegi Panjang yang baru (gabungan) merupakan xmax terbesar dari kedua buah Persegi Panjang. Untuk nilai ymin dan ymax juga menerapkan logika yang sama, yaitu ymin merupakan nilai atau besaran dalam atribut ymin dari Persegi Panjang baru (gabungan) merupakan ymin terkecil dari kedua buah Persegi Panjang. Lalu untuk ymax merupakan nilai atau besaran dalam atribut ymax dari Persegi Panjang baru (gabungan) merupakan ymax terbesar dari kedua Persegi Panjang tadi.

Tugas selanjutnya saya membuat operator Overloading dari operator (-). Pada dasarnya logika yang saya gunakan sama namun berkebalikan atau lebih overall ke lawan arah dari logika operator (+) diatas. Untuk method dari operator (-) adalah sebagai berikut:

```
//Operator (-) berfungsi untuk mengambil irisan dari dua persegi panjang tadi
PersegiPanjang PersegiPanjang::operator-(PersegiPanjang const &yoi)
{
    PersegiPanjang temp(0,0,0,0);

    if (*this == yoi)//Berfungsi penunjuk nilai dari variable persegi dan syarat yang dipenuhi (terdapat di line command no 16)
    {
        // Untuk mengetahui update dari x
        temp.xmin = min(this-> xmin,yoi.xmin);
        temp.xmax = max(this-> xmax,yoi.xmax);
        // Untuk mengetahui update dari y
        temp.ymin = min(this-> ymin,yoi.ymin);
        temp.ymax = max(this-> ymax,yoi.ymax);

    }else cout << "Persegi Panjang 1 dan 2 Tidak Beririsan, Persegi Panjang tidak bisa lanjut ke pengurangan";
        return temp;
}
```

Gambar 5. Method di Operator Overloading (-)

Setelah saya menyelesaikan method di operator overloading (-) saya mencoba membuat method di operator overloading lainnya yaitu (++) dan (--). Saya terlebih dahulu membuat operator (++) dan dapat dilihat method yang saya gunakan sebagai berikut:

```
//Operator (++) berfungsi untuk mengubah luasan persegi panjang dengan mengkali 2x luasan awalnya (diperbesar)
void PersegiPanjang::operator++()
{
    float pjg = 0,lbr = 0,Ttengah_x = 0,Ttengah_y = 0;
  //Persegi panjang yang diminta dengan memasukkan panjang dan lebar
    pjg = (this->xmax - this->xmin);
    lbr = (this->ymin - this->ymax);
    pjg = abs(pjg);
    lbr = abs(lbr);
  //Cara menentukkan titik-titik sumbu baru persegi tadi
    this->xmin = Ttengah_x - (pjg/2);
    this->xmax = Ttengah_x + (pjg/2);
    this->ymin = Ttengah_y - (lbr/2);
    this->ymax = Ttengah_y + (lbr/2);
  //Titik Tengah Yang dimasukkan dari Persegi panjang tadi
    Ttengah_x = pjg/2 + this->xmin;
    Ttengah_y = lbr/2 + this->ymin;
  //Lalu luasan persegi panjang menjadi 2 kali luasannya
    pjg*=2;
    lbr*=2;

}
```

Gambar 6. Method di Operator Overloading (++)

Fungsi dari Operator (++) dalam program ini adalah operator ini akan mengubah luasan dari Persegi Panjang untuk dilipat gandakan atau bahasa yang mudah dimengerti adalah luas dari Persegi Panjang akan dikalikan dua kali lipatnya. Dengan titik pusat perbesaran adalah titik tengahnya. Lantas perbesaran yang terjadi adalah disebabkan oleh panjang dan lebar dari Persegi Panjang yang tentunya sudah dikalikan dua kali lipatnya. Lalu dari operator maka bisa diketahui jika Persegi Panjang yang baru merupakan bentuk perbesaran dari luasan awal Persegi Panjang yang lama.

Lalu tugas berikutnya saya membuat method operator overloading dari operator overloading (--). Berbeda dari operator yang dijelaskan sebelumnya operator ini merupakan kebalikan dari operator overloading (++) yang diperbesar pada konsepnya. Untuk operator ini luasan Persegi Panjang akan di perkecil 0,5 nya atau ½ dari ukuran semula. Dan pada operator (--) sebenarnya mempunyai method yang sama namun perbedaan di tanda saja. Tampilan method yang saya gunakan sebagai berikut:

```
//Operator (--) berfungsi untuk membuat luasan persegi panjang menjadi 0,5 atau setengah luasan awalnya (diperkecil)
void PersegiPanjang::operator--()
{
    float pjg = 0,lbr = 0,Ttengah_x = 0,Ttengah_y = 0;
  //Persegi panjang yang diminta dengan memasukkan panjang dan lebar
    pjg = (this->xmax - this->xmin);
    lbr = (this->ymin - this->ymax);
    pjg = abs(pjg);
    lbr = abs(lbr);
  //Cara menentukkan titik-titik sumbu baru persegi tadi
    this->xmin = Ttengah_x - (pjg/2);
    this->xmax = Ttengah_x + (pjg/2);
    this->ymin = Ttengah_y - (lbr/2);
    this->ymax = Ttengah_y + (lbr/2);
  //Titik Tengah Yang dimasukkan dari Persegi panjang tadi
    Ttengah_x = pjg/2 + this->xmin;
    Ttengah_y = lbr/2 + this->ymin;
  //Lalu luasan persegi panjang menjadi setengah atau 0,5 kali luasannya
    pjg/=2;
    lbr/=2;
}
```

Gambar 7. Method di Operator Overloading (--)

Selanjutnya tugas saya yang terakhir adalah menentukan method di operator overloading ([]). Operator ini memiliki fungsi dapat mengembalikan nilai berupa *member atribut* dari Persegi Panjang yang dipilih sesuai dengan *index* yang dimasukkan pada program. Berikut merupakan method yang saya gunakan seperti berikut ini:

```
float PersegiPanjang::operator[](int huha)
{
    switch(huha)
    {
        case 1:
            return this->xmin;
            break;

        case 2:
            return this->xmax;
            break;

        case 3:
            return this->ymin;
            break;

        case 4:
            return this->ymax;
            break;
    }
        return 0;
}
```

Gambar 8. Method di Operator Overloading ([])

Penjelasan sederhananya void PersegiPanjang akan di kerjakan oleh 2 operator (++) dan (--) namun, dari kedua operator tadi hanya dipilih salah satu saja yang akan dilanjutkan

dan dicetak oleh program karena nanti akan masuk ke switch case di operator ([]) yang dimasukkan ke int dan akan break bila user mengakhiri program ini. Agar lebih jelas di gambar 9 tersebut menjelaskan tentang *post-increment* dan *pre-increment* (i++ dan ++i). Dari *statement* diatas maka disimpulkan jika ada (int) nya maka akan di overload yang (i++). Namun sebaliknya untuk yang tidak ada (int) nya maka akan di overload yang (++i) agar program berlanjut.

Lalu setiap fungsi akan bisa berjalan dengan menggunakan "void enter()". Disini void enter berfungsi sebagai lanjutan dari setiap fungsi dalam program untuk dijalankan oleh program. Namun, pada dasarnya walaupun tidak ada void enter tersebut program tetap bisa berlanjut dan jalan. Hanya saja saya disini ingin menambahkan hal tersebut agar user bisa berinteraktif selayaknya program pada umumnya yang selalu membantu user agar user bisa menggunakan program dengan baik dan berjalan lancar. Catatan untuk Void enter ini saya gunakan di file main.cpp dan juga PersegiPanjang.cpp dan PersegiPanjang.hpp. Namun dalam hpp saya hanya deklarasi di namespace enter dan void enter. Berikut beberapa screenshoot dari algoritma yang menggunakan void enter di program saya:

```
void enter()
{
    cout <<"Perhatian!!! Anda boleh menggunakan coding ini dijadikan sebagai referensi, namun tidak boleh keras menco
    cout <<"Harap tekan 'ENTER' untuk start dan melanjutkan program ini";
    cin.ignore();
    system("cls||clear");
}
```

Gambar 10. Void enter sebagai awalan program saya

Untuk cout yang awal itu kalimat yang saya buat terlalu panjang sehingga tidak bisa maksimal saya Screenshoot dan juga line yang terlalu panjang membuat tampilannya juga terpotong.

```
enter();
PersegiPanjang pp3;//Persegi Panjang pp1(5,6,7,8) Persegi Panjang pp2(8,7,4,4)
//Melihat nilai awal yang sudah diinput dari Persegi Panjang 1 dan 2
cout << "Nilai awal Persegi Panjang 1 : "<< endl;
pp1.printresult();
cout << "Nilai awal Persegi Panjang 2 : "<< endl;
pp2.printresult();
```

Gambar 11. Void enter sebagai penampil nilai awal dari Persegi Panjang 1 dan 2

```
enter();
//Program berjalan ke Operator (+)
pp3 = pp1 + pp2;
if (pp1==pp2)
{
cout << "Nilai Hasil Penjumlahan dari Persegi Panjang 1 dan 2 : "<< endl;
pp3.printresult();
}
```

Gambar 12. Void enter berjalan ke operator (+)

```
enter();
//Program berjalan ke Operator (-)
pp3 = pp1 - pp2;
if (pp1==pp2)
{
cout << "Nilai Hasil Pengurangan dari Persegi Panjang 1 dan 2 : "<<endl;
pp3.printresult();
}
```

Gambar 13. Void enter berjalan ke operator (-)

```
enter();
//Program berjalan ke Operator (++)
cout << "Nilai awal Persegi Panjang 1 : "<< endl;
pp1.printresult();
++pp1;
cout << "Nilai Persegi Panjang 1 setelah dilakukannya Operator (++) : "<< endl;
pp1.printresult();

cout << "Nilai awal Persegi Panjang 2 : "<< endl;
pp2.printresult();
++pp2;
cout << "Nilai Persegi Panjang 2 setelah dilakukannya Operator (++) : "<< endl;
pp2.printresult();
```

Gambar 14. Void enter berjalan ke operator (++)

```
enter();
//Program berjalan ke Operator (--)
cout << "Nilai awal Persegi Panjang 1 : "<< endl;
pp1.printresult();
--pp1;
cout << "Nilai Persegi Panjang 1 setelah dilakukannya Operator (--) : "<< endl;
pp1.printresult();

cout << "Nilai awal Persegi Panjang 2 : "<< endl;
pp2.printresult();
--pp2;
cout << "Nilai Persegi Panjang 2 setelah dilakukannya Operator (--) : "<< endl;
pp2.printresult();
```

Gambar 15. Void enter berjalan ke operator (--)

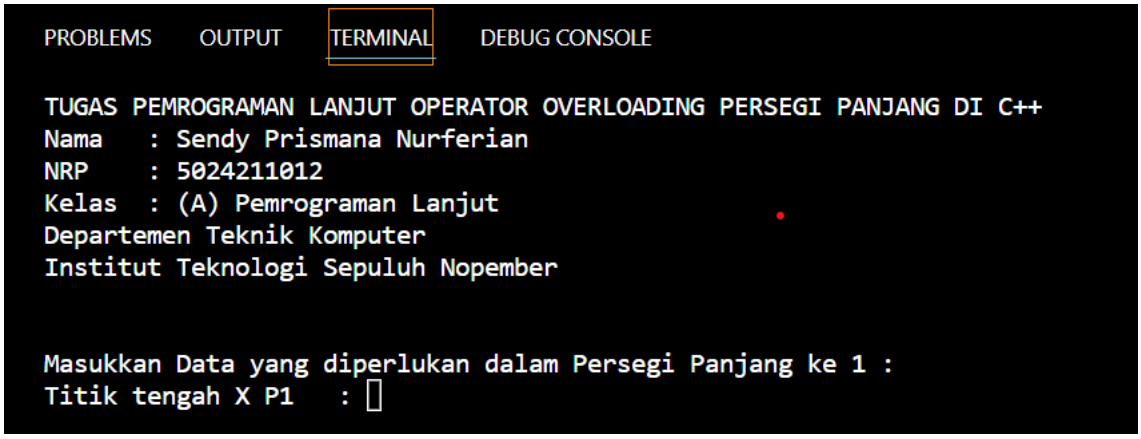
```
enter();
//Program berjalan ke Operator == berfungsi untuk membuktikan bahwa Persegi Panjang 1 dan 2 beririsan
if (pp1==pp2)
    cout << "Kedua Persegi Panjang saling beririsan"<< endl;
else cout << "Kedua Persegi Panjang tidak saling beririsan"<< endl;
```

Gambar 16. Void enter berjalan ke operator (==)

```
enter();
cout << "Program sudah selesai dijalankan, Terima Kasih";
cin.ignore();
system("cls||clear");
```

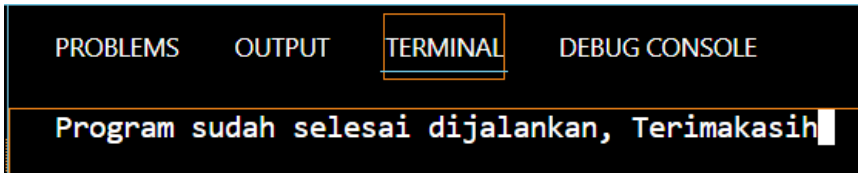
Gambar 17. Void enter mengakhiri program

Untuk menjalankan program maka kita mengkompilasi dan build dituliskan MakeFile. Caranya MakeFile ini dijalankan dengan mengetik command pada Terminal “make” dan proses ini akan mengambil semua *source code* yang dikompilasi dan menjadi *object* di folder build. Kemudian file-file *object* tadi akan di buat menjadi .exe(executable) file di folder utama. Setelah selesai maka muncul PersegiPanjang.exe nah ini program yang bisa kita jalankan dengan ketik run atau .\PersegiPanjang.exe di ketik pada Terminal maka program akan berjalan. Berikut hasilnya:



Gambar 18. Tampilan awal program saat dijalankan

Setelah selesai maka seperti inilah tampilan program nya:



Gambar 19. Tampilan akhir program saat dijalankan

Sekian penjelasan dari saya tentang tugas program Operator Overloading Persegi Panjang yang sudah saya buat. Mungkin dalam pembuatan laporan ini terdapat kekurangan dan kesalahan kata maupun ada kesalahan yang mungkin belum disadari oleh penulis. Saya berharap setelah dibuatnya laporan maupun source code nantinya bisa bermafaat bagi diri saya pribadi maupun pembaca lainnya.