LAPORAN OPERATOR OVERLOADING PEMROGRAMAN LANJUT

Disusun Oleh

Kenanya Keandra Adriel Prasetyo 5024211004

Dosen Pengampu

Reza Fuad Rachmadi 198504032012121000

INSTITUT TEKNOLOGI SEPULUH NOPEMBER SURABAYA

TEKNIK KOMPUTER

2022

Desain Pembuatan Program

Program dapat dilihat pada link https://github.com/KenanyaKAP/operator-overload. Program dibuat dengan menggunakan bahasa C++. Terdapat tiga file yang berbeda, dalam folder scr terdapat main.cpp dan square.cpp. Sedangkan dalam folder include terdapat satu file yaitu square.hpp. Semua file tersebut dicompile dengan menggunakan makefile dan menghasilkan file executable main.exe. File square.hpp dan square.cpp digunakan untuk membuat class PersegiPanjang. Dalam square.hpp berisi deklarasi class PersegiPanjang. Sedangkan dalam square.cpp berisikan definisi dari class PersegiPanjang.

Class PersegiPanjang adalah kelas yang akan membuat persegi panjang. Kelas ini memiliki atribut private yaitu xmin, xmax, ymin, dan ymax. Dimana xmin adalah koordinat x sisi kiri persegi panjang, xmax adalah koordinat x sisi kanan persegi panjang, ymin adalah koordinat y sisi bawah persegi panjang, dan ymax adalah koordinat y sisi atas persegi panjang. Keempat atribut dibuat dengan tipe data float, karena dalam pengolahan operasi persegi panjang, memungkinkan terjadinya sisi memiliki nilai desimal. Kelas juga memiliki public function yaitu constructor, dan operator overloading. Operator overloading dari kelas ini adalah penambahan(+), pengurangan(-), increment(++), decrement(--), index access([]) dan comparison equal to(==).

Constructor class PersegiPanjang memiliki input berupa titik tengah dari persegi panjang, dan panjang serta lebar dari persegi panjang yang akan dibuat. Maka dari itu constructor class PersegiPanjang memiliki empat parameter dengan tipe data integer yaitu xCenter, yCenter, width_x, dan height_y. xCenter adalah koordinat x titik tengah persegi panjang, yCenter adalah koordinat y titik tengah persegi panjang, width_x adalah panjang persegi panjang dalam sumbu x, dan height_y adalah lebar persegi panjang dalam sumbu y. Maka dari itu, logika untuk mendapatkan xmin adalah xCenter dikurangi dengan setengah dari panjang yaitu width_x/2. Sedangkan untuk xmax adalah xCenter ditambah dengan width_x/2. Begitu juga untuk mencari ymin dan ymax.

Operator overloading penambahan (+) akan menghasilkan persegi panjang dengan luasan gabungan dari dua persegi panjang, dan dengan syarat kedua persegi panjang harus beririsan. Operator + memiliki return berupa class PersegiPanjang dan dengan parameter berupa const PersegiPanjang &rhs. Digunakan reference (&) agar parameter function ini

akan langsung menjadi alias dari parameter dan tidak membuat variable baru, sedangkan const digunakan untuk memproteksi agar parameter rhs tidak berubah saat function dijalankan. Operator + juga dibuat menjadi const agar tidak mengubah variable yang melakukan operator, yaitu pada sisi kiri (lhs/left hand side). Sebagai contoh pp1 + pp2, maka dari itu pp1 akan melakukan operator + dengan pp2 sebagai parameternya (rhs/right hand side). Dalam definisinya, akan dilakukan pengecekan terlebih dahulu apakah kedua persegi panjang tersebut beririsan atau tidak dengan menggunakan operator ==, yaitu "if (*this == rhs)". Apabila kedua persegi panjang beririsan, maka komparasi akan mengeluarkan nilai true (1). Dan bila true, maka logika untuk mendapatkan xmin adalah nilai xmin terkecil dari kedua persegi panjang, yaitu "min(this->xmin, rhs.xmin)". Sedangkan untuk mendapatkan xmax adalah nilai xmax terbesar dari kedua persegi panjang, yaitu dengan "max(this->xmax, rhs.xmax)". Begitu juga untuk mencari ymin dan ymax. Kemudian akan dibuat variable PersegiPanjang baru dan mengubah semua atributnya, lalu variable tersebut akan di return.

Operator overloading pengurangan (-) memiliki return, dan parameter yang sama dengan operator penambahan (+). Hanya saja operator pengurangan akan menghasilkan persegi panjang yang beririsan antara kedua persegi panjang. Operator ini juga memiliki syarat yang sama dengan operator +, yaitu kedua persegi panjang harus beririsan. Untuk mendapatkan persegi panjang irisan dari kedua persegi panjang, maka logika untuk mencari xmin adalah nilai xmin terbesar dari kedua persegi panjang, yaitu "max(this->xmin, rhs.xmin)". Sedangkan untuk xmax adalah nilai xmax terkecil dari kedua persegi panjang, yaitu "min(this->xmax, rhs.xmax)". Begitu juga untuk mencari ymin dan ymax. Dengan begitu akan didapatkan persegi panjang irisan lalu di return.

Operator overloading increment (++) akan menambah luasan persegi panjang menjadi dua kali luasan awalnya. Operator ++ memiliki return berupa class PersegiPanjang, dan tidak memiliki parameter sama sekali. Logika dari operator ++ sebagai contoh adalah misalkan suatu persegi panjang memiliki panjang "X" dan lebar "Y". Saat dikenai operator ++ maka panjang persegi panjang tersebut akan menjadi "2X" dan lebarnya menjadi "2Y", dengan titik tengah persegi panjang tidak berubah. Maka dari diperlukan dua variable, width (panjang) yaitu "this->xmax - this->xmin", dan height (lebar) yaitu "this->ymax - this->ymin". Dan untuk mencari xmin yang baru hanya perlu menguranginya dengan setengah dari panjang, yaitu "this->xmin -= width/2", dan untuk

xmax adalah menambahnya dengan setengah dari panjang, yaitu “this->xmax += width/2”. Begitu juga untuk mencari ymin dan ymax. Dengan begini panjang dan lebar persegi panjang akan menjadi dua kali lipat dari awalnya. Dan hal terakhir yang perlu dilakukan adalah mengembalikan hasil operator, yaitu dengan “return *this”.

Operator overloading decrement (--) akan mengurangi luasan persegi panjang menjadi setengah kali dari luasan awalnya. Operator -- memiliki return berupa class PersegiPanjang, dan tidak memiliki parameter sama sekali. Logika dari operator -- sebagai contoh adalah misalkan suatu persegi panjang memiliki panjang “X” dan lebar “Y”. Saat dikenai operator -- maka panjang persegi panjang tersebut akan menjadi “½X” dan lebarnya menjadi “½Y”, dengan titik tengah persegi panjang tidak berubah. Maka dari diperlukan dua variable, width (panjang) yaitu “this->xmax - this->xmin”, dan height (lebar) yaitu “this->ymax - this->ymin”. Dan untuk mencari xmin yang baru hanya perlu menambahnya dengan seperempat dari panjang, yaitu “this->xmin += width/4”, dan untuk xmax adalah menguranginya dengan seperempat dari panjang, yaitu “this->xmax -= width/4”. Begitu juga untuk mencari ymin dan ymax. Dengan begini panjang dan lebar persegi panjang akan menjadi setengah dari awalnya. Dan hal terakhir yang perlu dilakukan adalah mengembalikan hasil operator, yaitu dengan “return *this”.

Operator overloading index access ([]) berfungsi untuk mengambil masing-masing atribut dari class PersegiPanjang. Maka dari itu operator ini memiliki return berupa float, dan parameter berupa const int index. Sehingga untuk memetakan keluaran, hanya perlu menggunakan switch pada index, bila “case 0: return xmin”, “case 1: return xmax”, “case 2: return ymin”, “case 3: return ymax”, dan “default: return 0” bila index bukanlah angka dari nol sampai tiga.

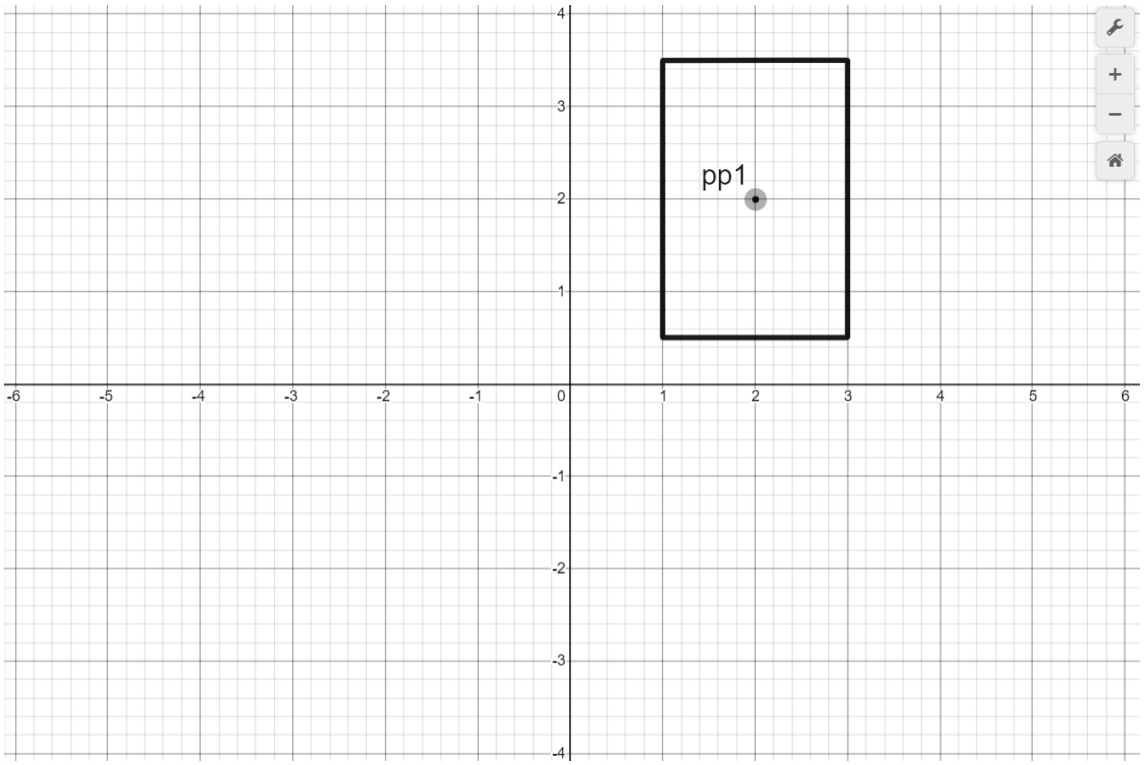
Dan yang terakhir adalah operator overloading comparison equal to (==) berfungsi untuk menentukan apakah kedua persegi panjang yang dikomparasi saling beririsan atau tidak. Operator == memiliki return berupa boolean, dan parameter berupa const PersegiPanjang &rhs. Operator == juga dibuat menjadi const agar tidak mengubah variable yang melakukan operator, yaitu pada sisi kiri (lhs/left hand side). Untuk mengetahui apakah persegi panjang 1 (pp1) dan persegi panjang 2 (pp2) beririsan, maka perlu dilakukan pengecekan sisi persegi panjang. Pada sumbu x perlu dilakukan pengecekan apakah xmin pp1 lebih kecil dari xmax pp2, dan apakah xmin pp2 lebih kecil

dari xmax pp1, yaitu “if (this->xmin < rhs.xmax && rhs.xmin < this->xmax)”. Apabila pp2 berada di kanan pp1, maka logika akan gagal pada “rhs.xmin < this->xmax)”, bila pp2 berada di kiri pp1, maka logika akan gagal pada “this->xmin < rhs.xmax”. Pengecekan dilakukan dengan tanda lebih kecil “<” karena bila menggunakan lebih kecil sama dengan “<=”, pengecekan akan tetap menghasilkan true bahkan jika kedua persegi panjang hanya berhimpitan (tidak beririsan). Pengecekan yang sama juga dilakukan untuk sumbu y. Bila kedua pengecekan menghasilkan true, maka function akan menghasilkan “return true”, bila salah satu saja dari pengecekan salah, maka function akan menghasilkan “return false”.

Pembahasan Hasil Program

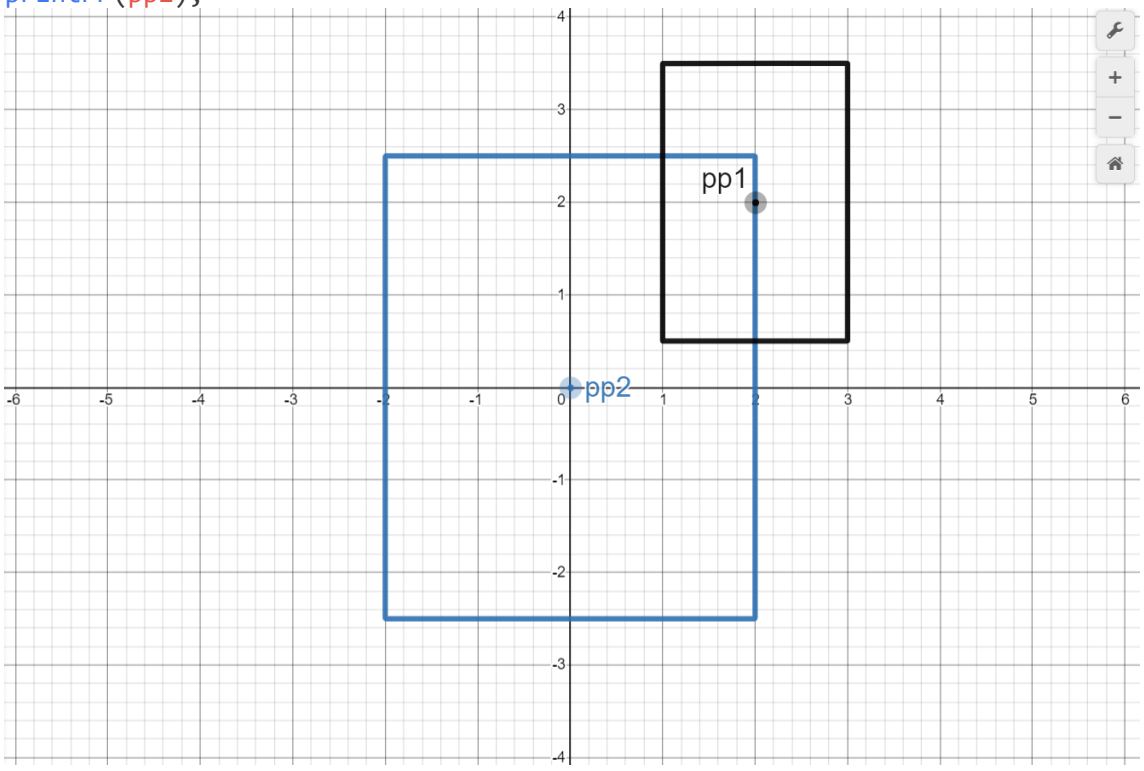
Pada main.cpp, terdapat function printPP dengan return void dan parameter const PersegiPanjang &pp. Function ini digunakan untuk mencetak di terminal atribut yang dimiliki persegi panjang, dan panjang serta lebar persegi panjang yang diinputkan. Dengan runtutan program main sebagai berikut:

```
PersegiPanjang pp1(2, 2, 2, 3);
printPP(pp1);
```




```
[terminal]:
Membuat persegi panjang 1:
x min     : 1
x max     : 3
y min     : 0.5
y max     : 3.5
panjang x : 2
lebar  y  : 3
```

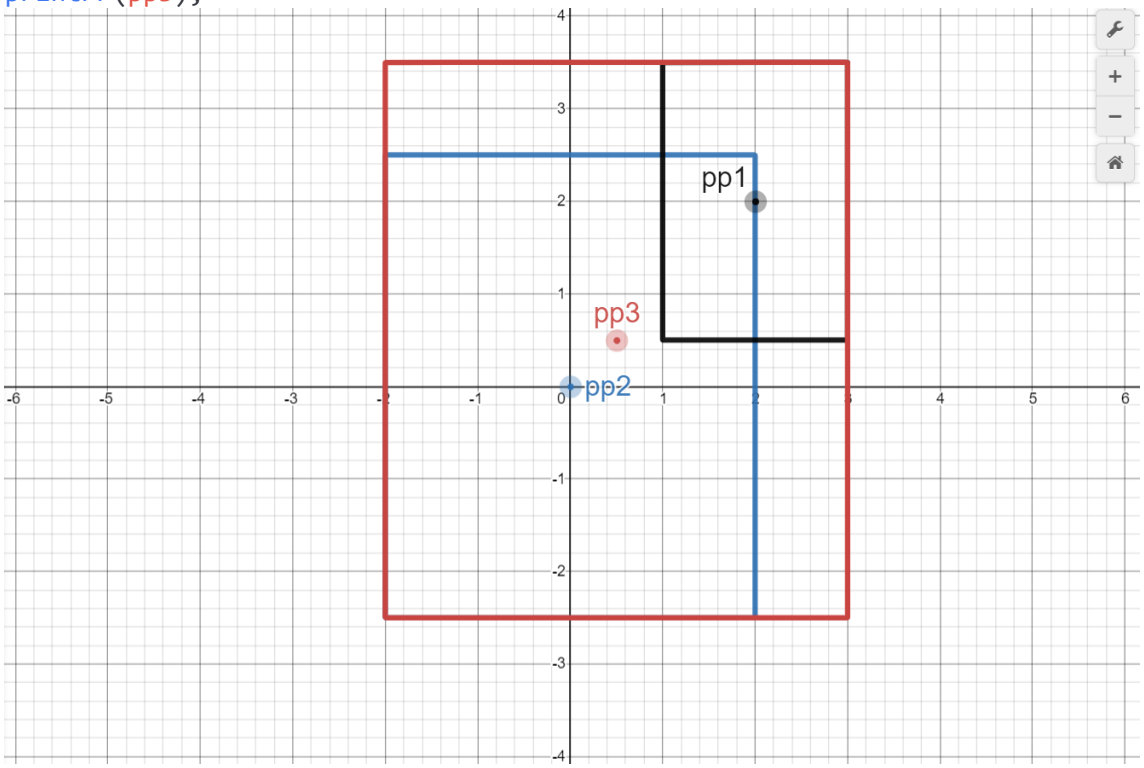
```
PersegiPanjang pp2(0, 0, 4, 5);
printPP(pp2);
```




```
[terminal]:
Membuat persegi panjang 2:
x min     : -2
x max     : 2
y min     : -2.5
y max     : 2.5
panjang x : 4
lebar  y  : 5
```

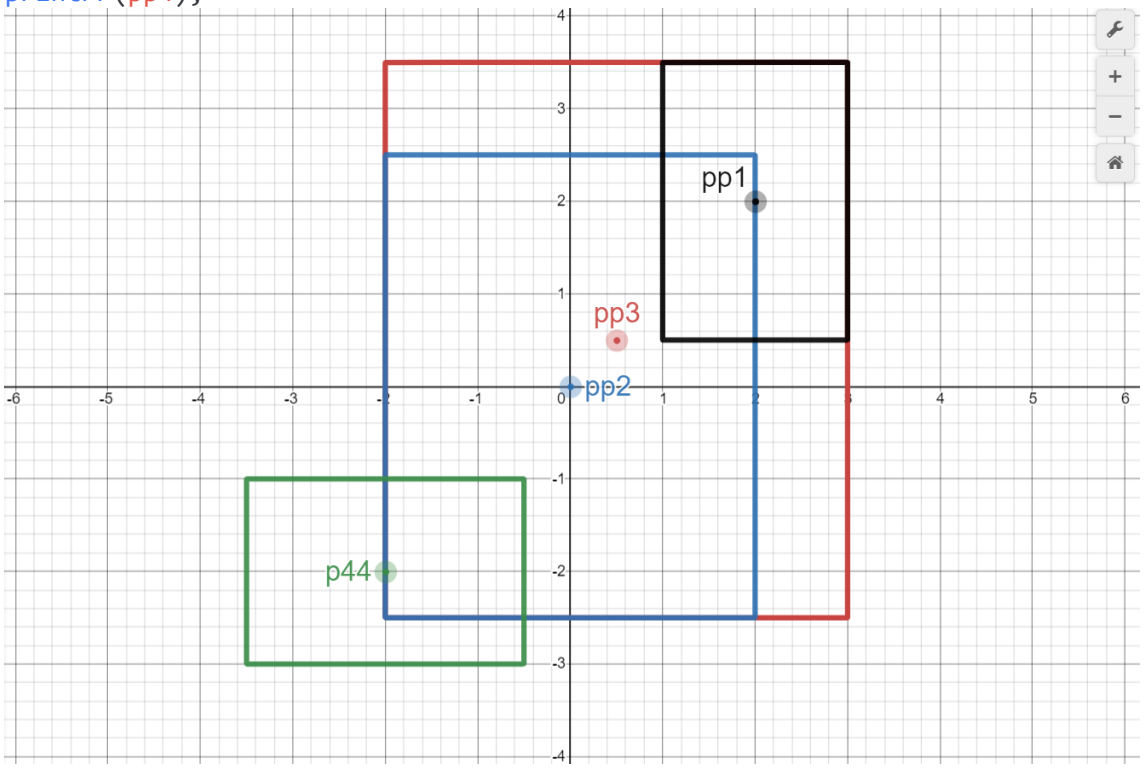
```
cout << ((pp1 == pp2) ? "Ya" : "Tidak") << endl;

[terminal]:
Apakah pp1 dan pp2 beririsan:
Ya
```

```
PersegiPanjang pp3 = pp1 + pp2;
printPP(pp3);
```



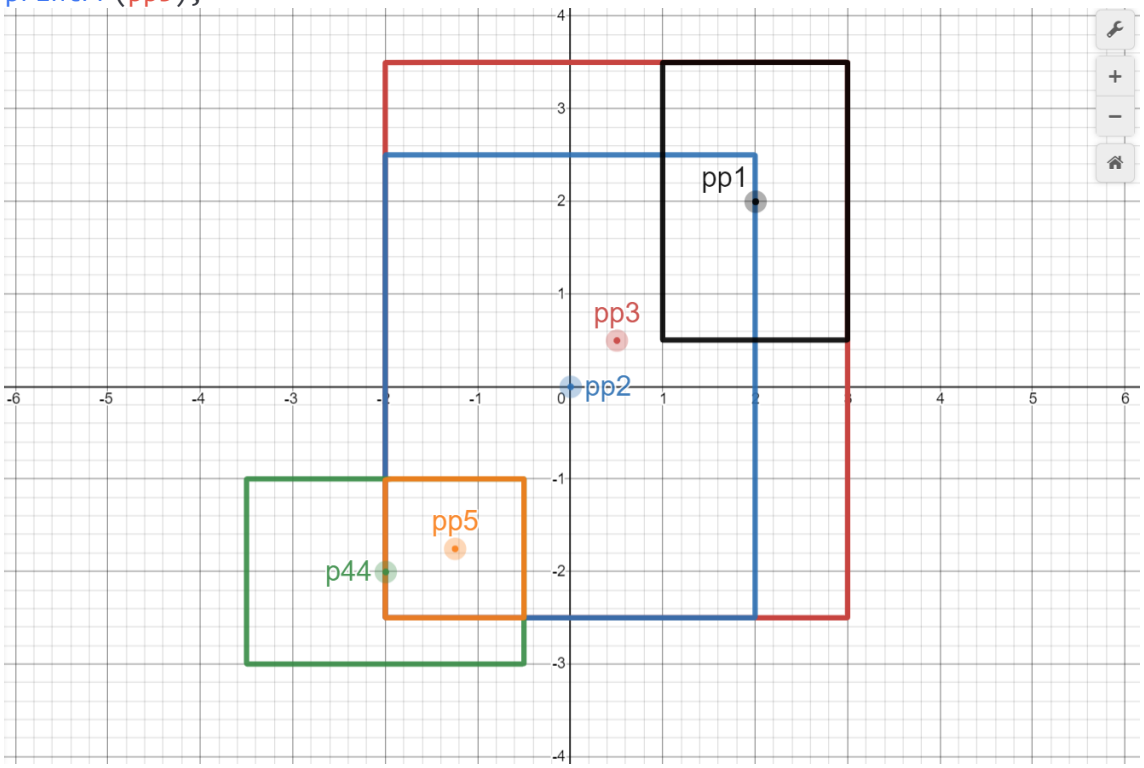

```
[terminal]:
Menambah pp1 dan pp2: (pp3)
x min     : -2
x max     : 3
y min     : -2.5
y max     : 3.5
panjang x : 5
lebar  y  : 6
```

```
PersegiPanjang pp4(-2, -2, 3, 2);
printPP(pp4);
```



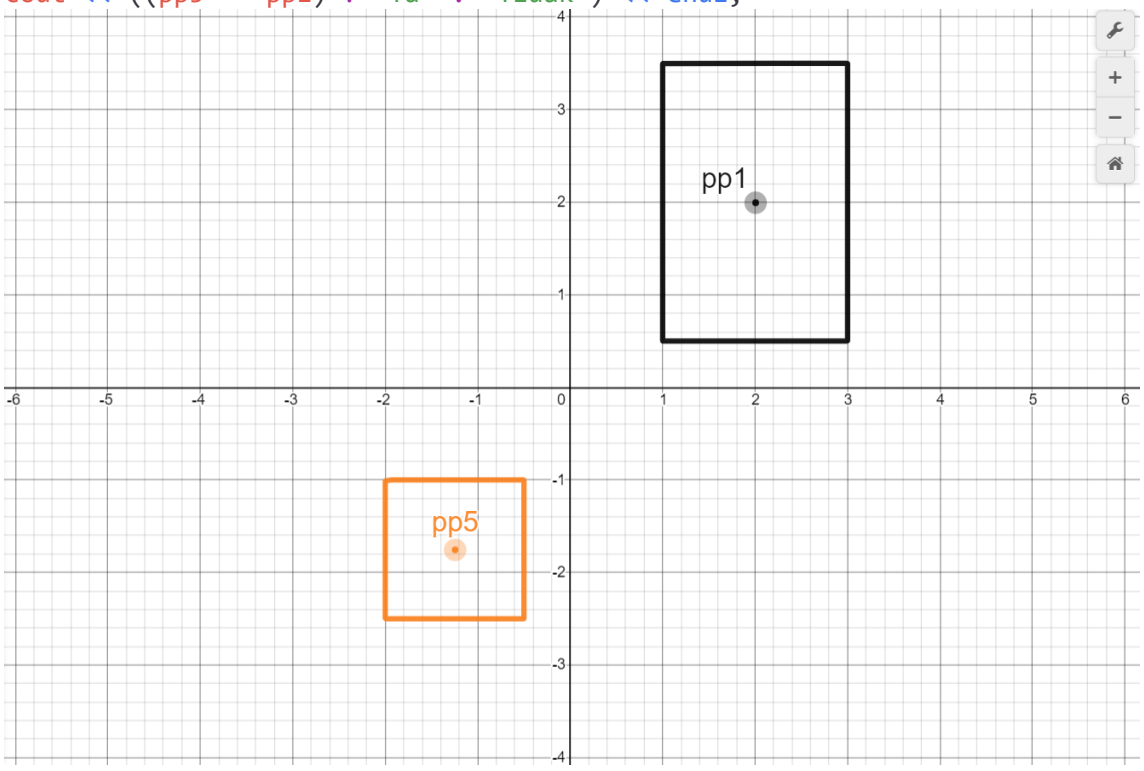

```
[terminal]:
Membuat persegi panjang 4:
x min     : -3.5
x max     : -0.5
y min     : -3
y max     : -1
panjang x : 3
lebar  y  : 2
```

```
PersegiPanjang pp5 = pp2 - pp4;
printPP(pp5);
```



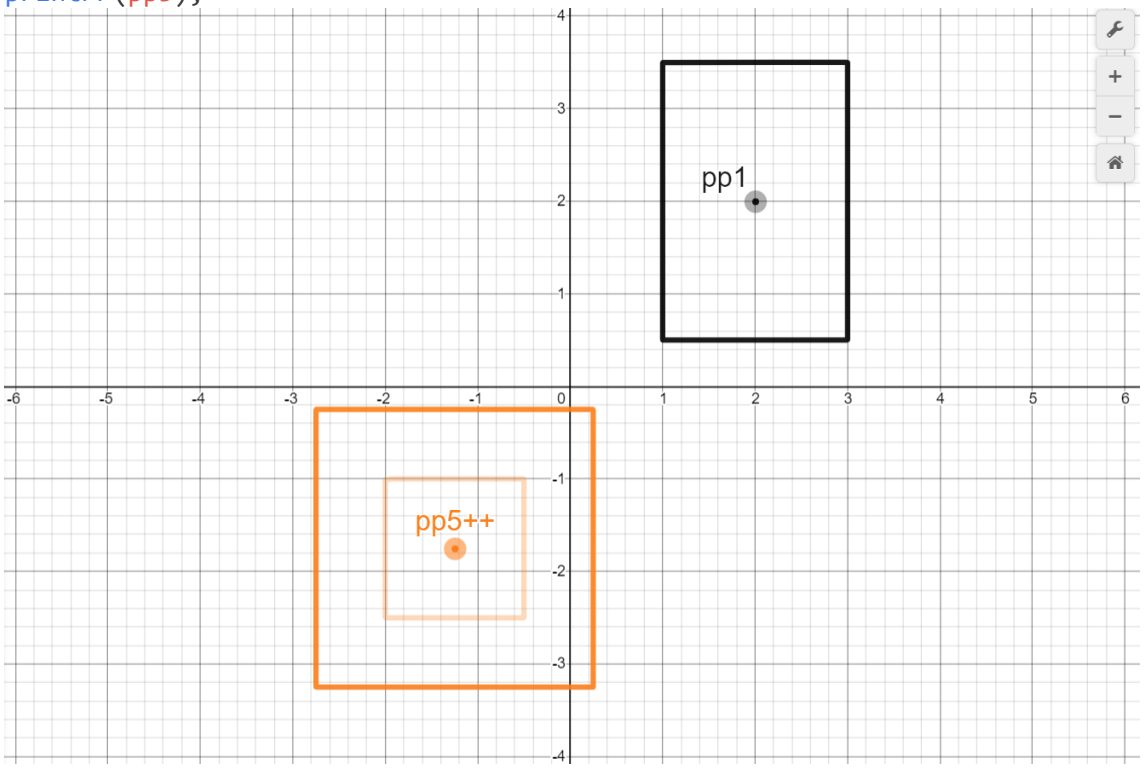

```
[terminal]:
Mengurangi pp2 dan pp4: (pp5)
x min     : -2
x max     : -0.5
y min     : -2.5
y max     : -1
panjang x : 1.5
lebar  y  : 1.5
```

```
cout << ((pp5 == pp1) ? "Ya" : "Tidak") << endl;
```



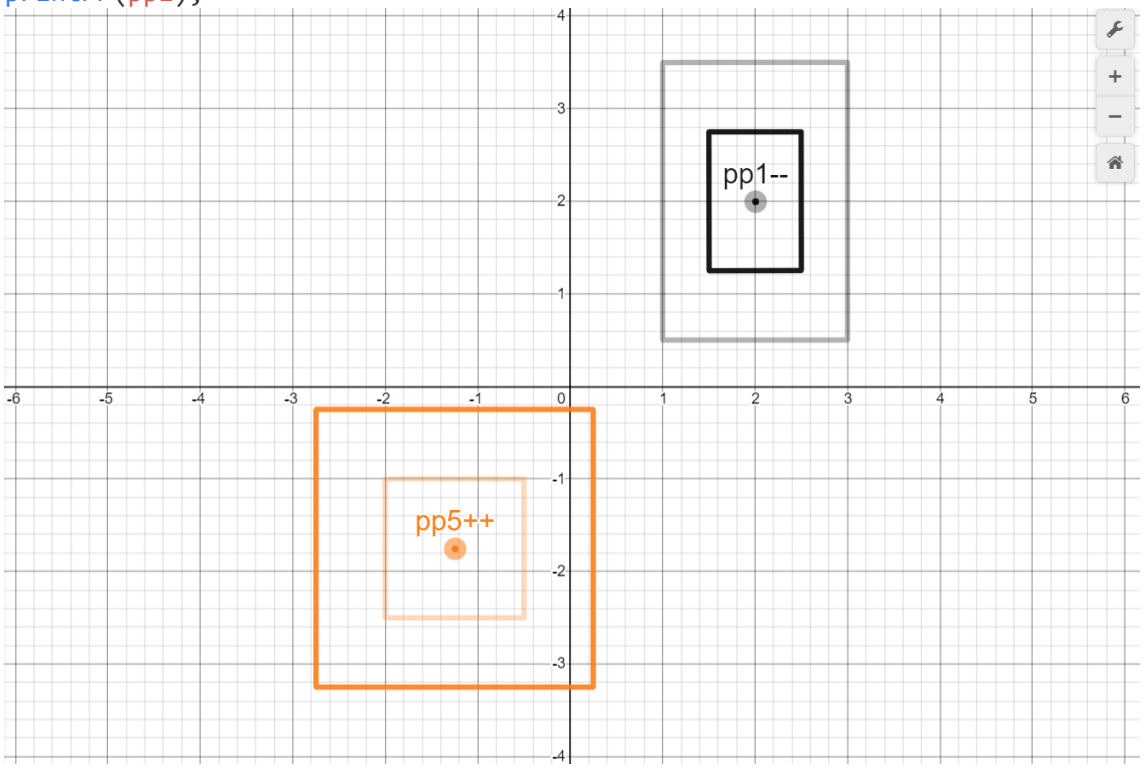

```
[terminal]:
Apakah pp5 dan pp1 beririsan:
Tidak
```

```
++pp5;
printPP(pp5);
```




```
[terminal]:
Melakukan operator ++ pada pp5:
x min     : -2.75
x max     : 0.25
y min     : -3.25
y max     : -0.25
panjang x : 3
lebar  y  : 3
```

```
--pp1;
printPP(pp1);
```




```
[terminal]:
Melakukan operator -- pada pp1:
x min     : 1.5
x max     : 2.5
y min     : 1.25
y max     : 2.75
panjang x : 1
lebar  y  : 1.5
```