GRAFIKA KOMPUTER

Translate, Rotate, Scale

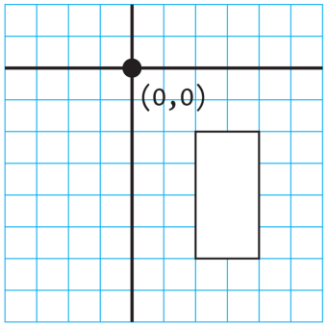
I. TUJUAN

- Mahasiswa mampu mengaplikasikan transformasi koordinat menggunakan fungsi translate(), Scale(), dan Rotation()

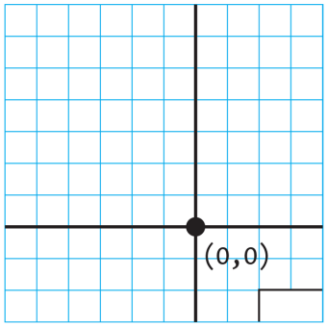
II. DASAR TEORI DAN PERCOBAAN

Teknik lain untuk penentuan posisi dan memindahkan objek gambar di layar adalah dengan mengubah sistem koordinat layar. Sebagai contoh, Anda dapat memindahkan bentuk 50 piksel ke kanan, atau Anda dapat memindahkan lokasi koordinat (0,0) 50 piksel ke kanan — hasil visualnya sama.

```
translate(40, 20);
rect(20, 20, 20, 40);
```



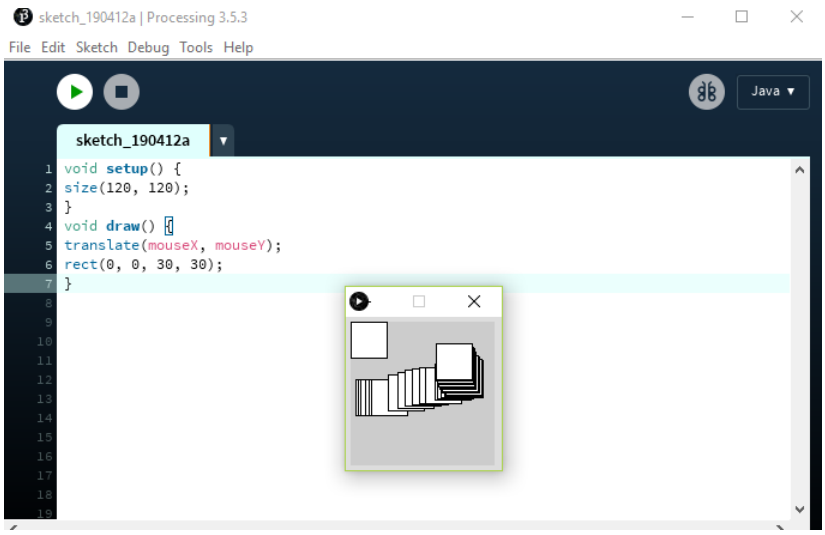

```
translate(60, 70);
rect(20, 20, 20, 40);
```



Dengan memodifikasi default sistem koordinat, kita dapat membuat transformasi berbeda termasuk translate, rotasi, dan scaling. Translate() merupakan fungsi yang paling mudah digunakan untuk melakukan transformasi. fungsi ini dapat menggeser sistem koordinat ke kiri, kanan, atas, dan bawah.

Percobaan 1: Translating Location

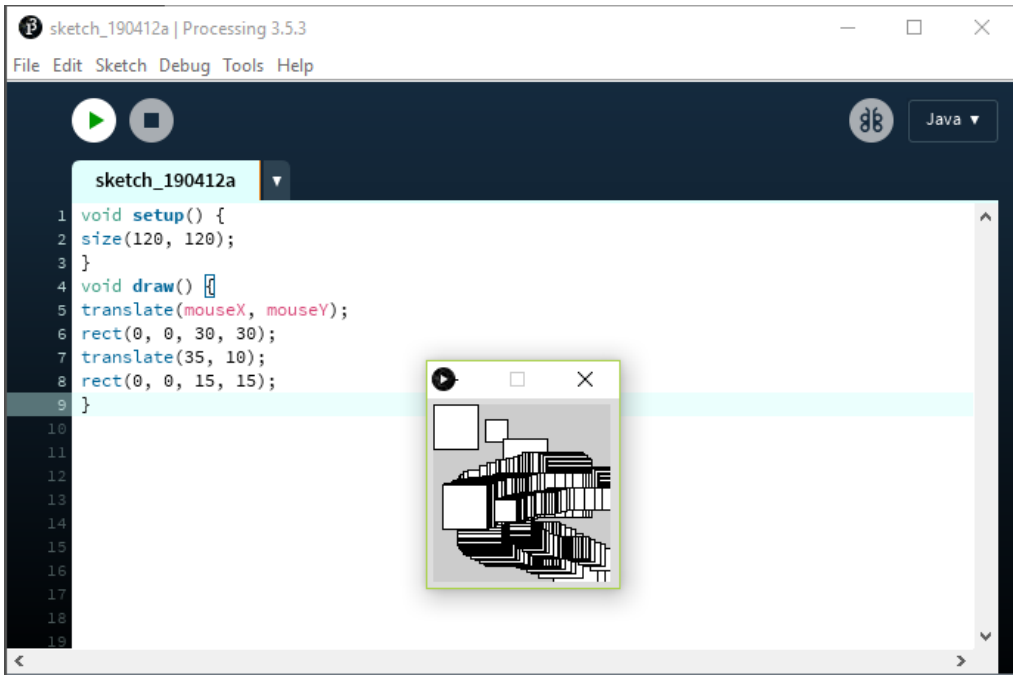
cobalah program di bawah ini dan analisa , perhatikan bahwa persegi panjang digambar pada koordinat (0,0), tetapi bergerak di sekitar layar, karena terpengaruh dengan translate():



Fungsi translate() mengatur koordinat (0,0) layar ke lokasi mouse (mouseX dan mouseY). Setiap kali blok draw() berulang, rect() ditarik pada origin baru, yang berasal dari lokasi mouse saat ini.

Percobaan 2: Multiple Translation

Cobalah program di bawah ini dan analisa. Setelah transformasi dibuat, dan diterapkan ke semua fungsi menggambar yang ada. Perhatikan apa yang terjadi ketika fungsi translate() kedua ditambahkan untuk mengontrol gambar persegi panjang kedua:

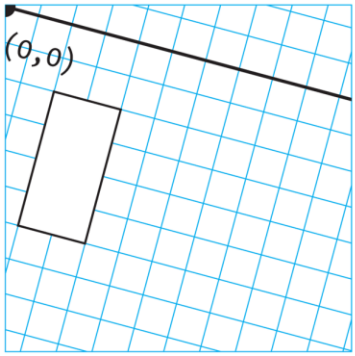


Nilai untuk fungsi translate() akan ditambahkan secara bersamaan. Sehingga kotak yang lebih kecil diterjemahkan ke jumlah mouseX + 35 dan mouseY + 10. Koordinat x dan y untuk kedua persegi panjang adalah (0,0), tetapi fungsi translate () memindahkannya ke posisi lain pada layar. Namun, meskipun transformasi terakumulasi di dalam blok draw(), semua akan kembali reset setiap kali draw () dimulai lagi pada atas.

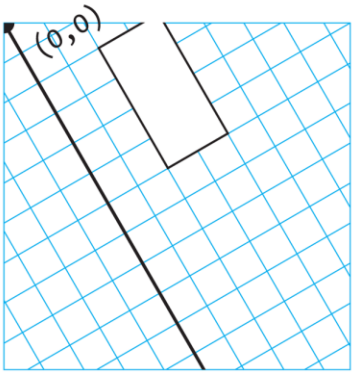
Fungsi rotate() untuk memutar sistem koordinat. Ada satu parameter, yang merupakan sudut (dalam radian) untuk memutar. Yang selalu berputar relatif terhadap (0,0), atau dikenal sebagai berputar di sekitar titik asal. Gambar 3-2 dalam Contoh 3-7 pada halaman

18 menunjukkan sudut radian nilai-nilai. Gambar 6-2 menunjukkan perbedaan antara berputar dengan angka positif dan negatif.

```
rotate(PI/12.0)
rect(20, 20, 20, 40);
```

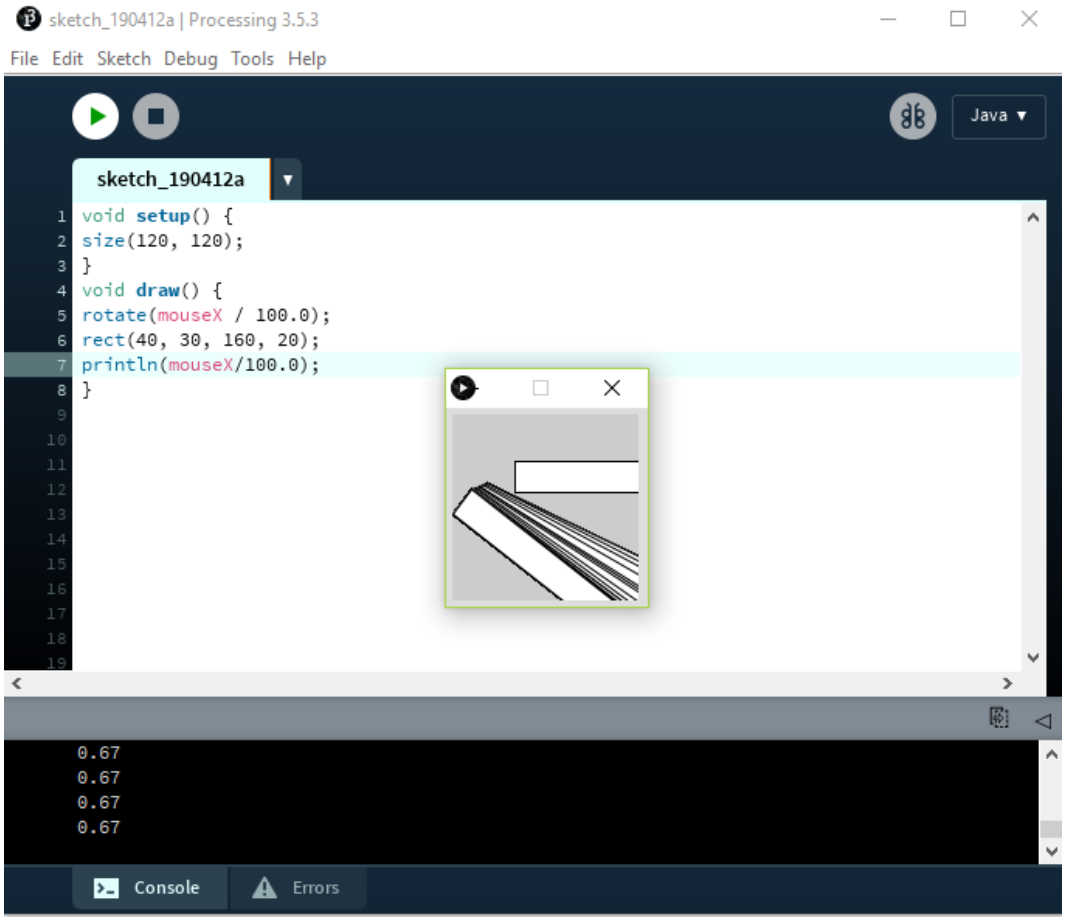



```
rotate(-PI/3);
rect(20, 20, 20, 40);
```



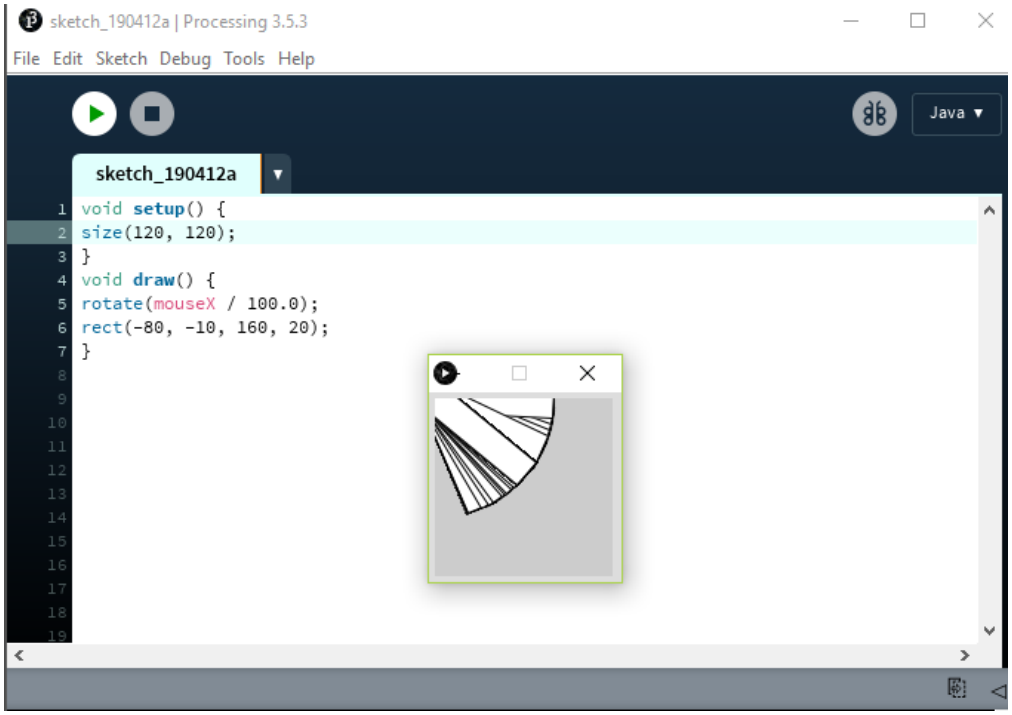
Percobaan 3: Corner Rotation

Cobalah program di bawah ini dan analisa. Untuk memutar gambar, pertama-tama tentukan sudut rotasi dengan rotate(), lalu gambar bentuknya. Dalam sketsa ini, jumlah yang akan diputar (mouseX / 100.0) akan berada di antara 0 dan 1.2 untuk menentukan sudut rotasi karena mouseX akan berada di antara 0 dan 120, lebar Tampilan Window ditentukan dengan fungsi size (). Catatan Anda harus membagi dengan 100.0 bukan 100, sebagaimana angka bekerja di Processing.



Percobaan 4: Center Rotation

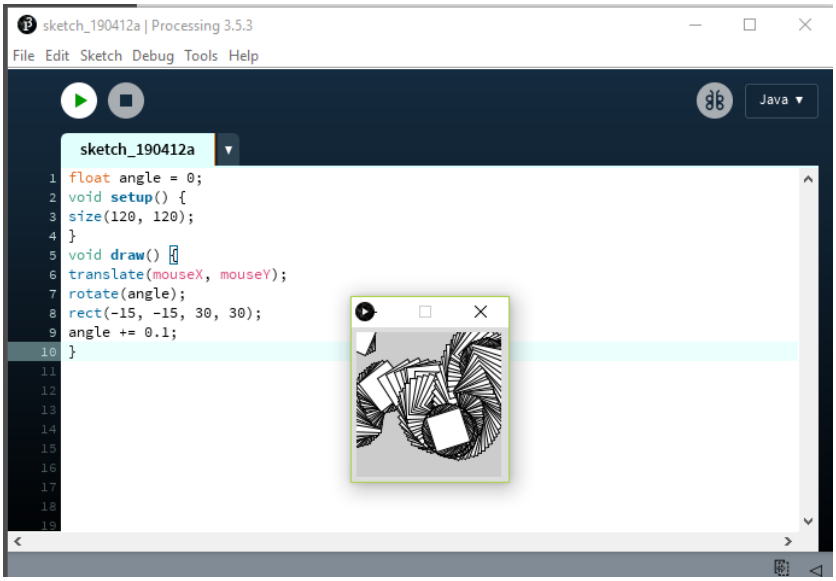
Untuk memutar gambar di sekitar pusatnya sendiri, maka ia harus digambar dengan koordinat (0,0) di tengah. Dalam contoh ini, karena lebar bentuknya 160 dan tinggi 20 seperti yang didefinisikan dalam rect(), maka digambarkan pada koordinat (–80, –10) untuk menempatkan (0,0) di pusat gambar.



Sepasang contoh sebelumnya menunjukkan cara memutar pada koordinat (0,0), tetapi bagaimana dengan kemungkinan lain? kita bisa gunakan fungsi translate () dan rotate () untuk kontrol lebih lanjut. Ketika mereka digabungkan, urutan penulisannya akan memengaruhi hasil. Jika sistem koordinat pertama kali dipindahkan dan kemudian diputar, akan berbeda hasilnya dengan yang dari pertama memutar sistem koordinat, lalu memindahkannya.

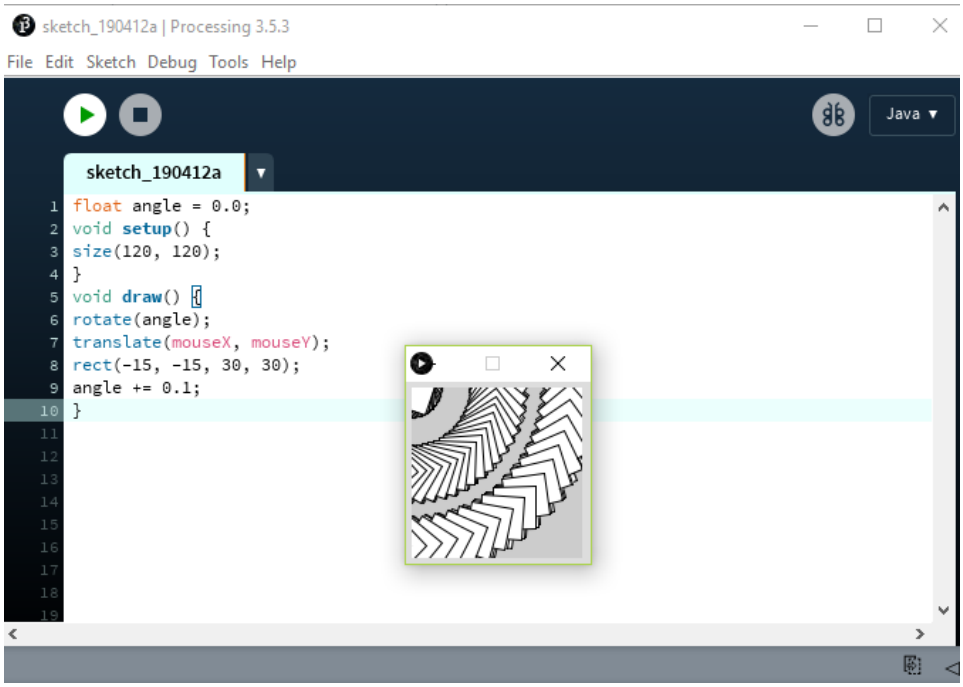
Percobaan 5: Translation then Rotation

Cobalah program di bawah ini dan analisa. Untuk memutar bentuk di sekitar titik tengahnya dan berada jauh dari titik asalnya, pertama-tama gunakan translate() untuk pindah ke lokasi sesuai keinginan, lalu panggil rotate(), dan gambar bentuk dengan pusatnya di koordinat (0,0):



Percobaan 6: Rotation, Then Translation

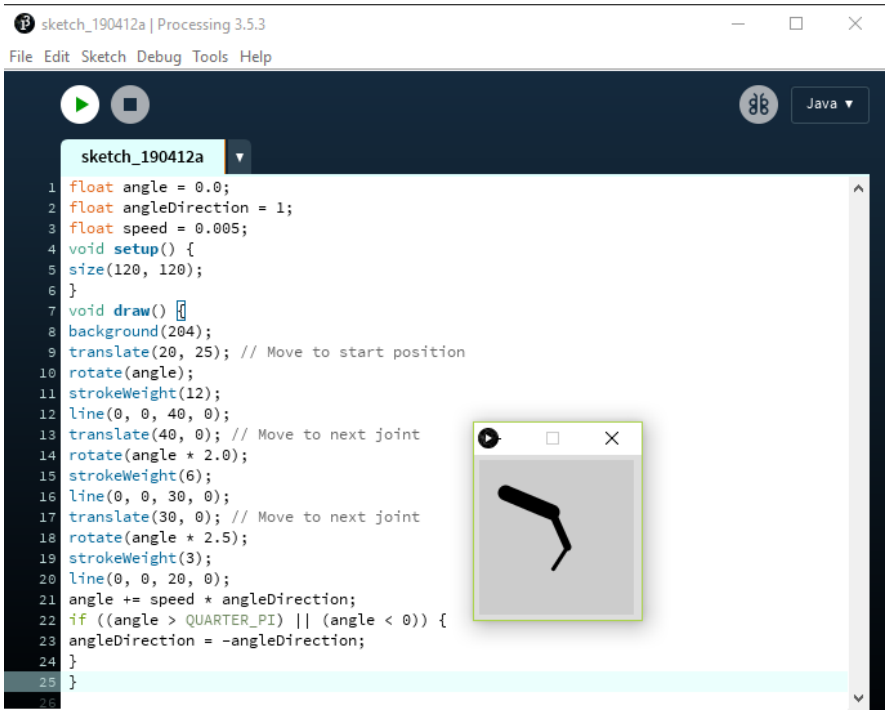
Cobalah program di bawha ini dan analisa. Bentuk yang sekarang berputar di sudut kiri atas Display Window, dengan jarak dari sudut yang ditentukan oleh translate():



Pilihan lain adalah menggunakan rectMode (), ellipsemode (), imageMode (), dan shapeMode (), yang membuatnya lebih mudah untuk menggambar bentuk dari pusatnya.

Percobaan 7: An Articulating Arm

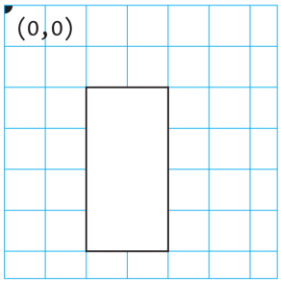
Cobalah program di bawah ini dan analisa. Dalam contoh ini, dikumpulkan beberapa fungsi translate() dan rotate() untuk membuat lengan yang terhubung dan bisa ditekuk. Setiap translate() selanjutnya menggerakkan posisi garis, dan setiap fungsi rotate() menambah rotasi setiap garis agar lengan bisa menekuk lebih banyak :



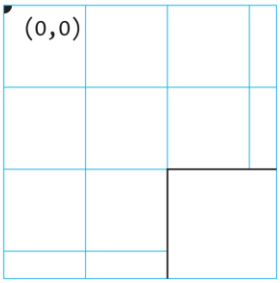
Nilai variabel Angel meningkat mulai 0 sampai dengan QUARTER_PI (seperempat nilai PI) lalu menurun sampai dengan kurang dari 0, dan terus berulang. Nilai dari variable angelDirection selalu 1 atau -1 untuk membuat nilai dari sudut meningkat dan menurun.

Fungsi scale () digunakan untuk meregangkan koordinat di layar. Karena koordinat bisa berkembang atau berkontraksi sesuai dengan perubahan skala, semua yang digambar di Jendela Display pun dimensinya bisa bertambah atau berkuran. Misalnya bisa menggunakan skala (1,5) untuk membuat semua objek menjadi 150% dari ukuran aslinya, atau skala (3) untuk membuat semua objek tiga kali lebih besar. Menggunakan skala (1) tidak akan berpengaruh, karena semuanya akan tetap ukurannya 100% sesuai aslinya. Sebaliknya untuk membuat sesuatu setengah ukurannya, gunakan skala (0,5).

```
scale(1.5);
rect(20, 20, 20, 40);
```




```
scale(3);
rect(20, 20, 20, 40);
```



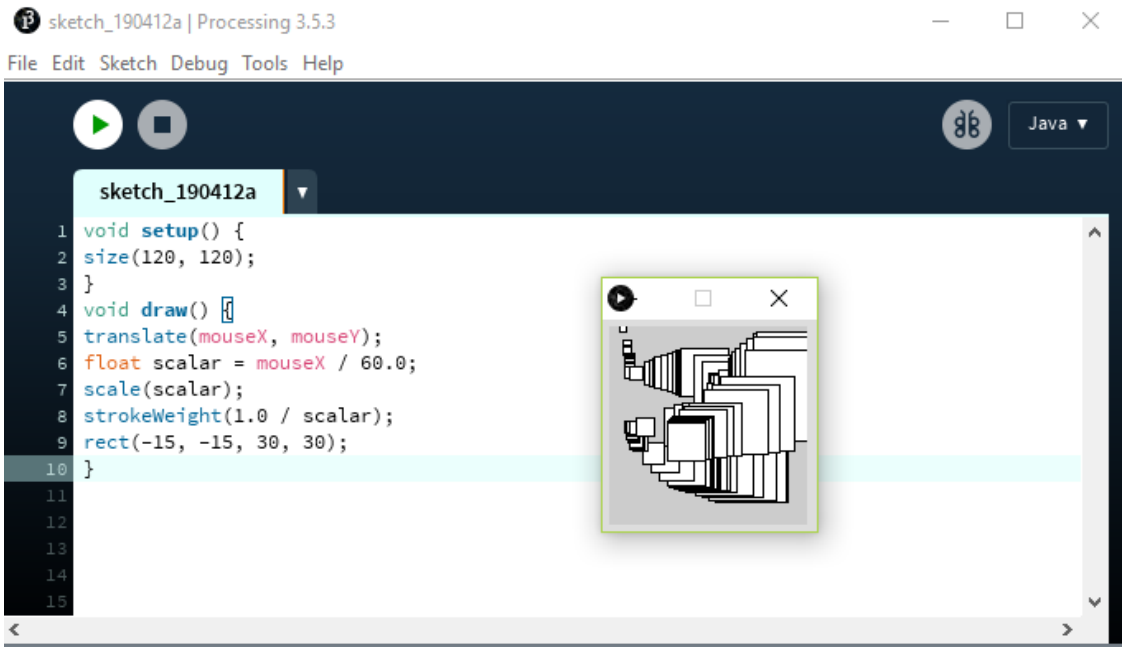
Percobaan 8: Scaling

Cobalah program di bawah ini dan analisa. Seperti fungsi rotate(), fungsi scale() digunakan untuk mentransformasikan objek dari bentuk originnya. Oleh karena itu, seperti halnya dengan rotate (), untuk merubah skala suatu bentuk dari pusatnya, maka perlu dilakukan

translate koordinat lokasinya, memasukkan nilai skalanya dan menggambar objek di koordinat (0,0):

Percobaan 9: Keeping Strokes Consistent

Cobalah program dibawah ini dan analisa hasilnya, melihat bagaimana fungsi scale() mempengaruhi strokeWeight.



Untuk mempertahankan efek dari transformasi suatu bentuk bisa digunakan fungsi pushMatrix()dan popMatrix() functions. Saat fungsi pushMatrix() dijalankan maka koordinat terkahir akan disimpan dan dikembalikan pada system saat fungsi popMatrix() dijalankan.

Percobaan 10: Isolating Transformations

Cobalah program di bawah ini dan analisa. Dalam contoh ini, persegi panjang yang lebih kecil selalu menarik posisi yang sama karena translate (mouseX, mouseY) dibatalkan oleh fungsi popMatrix ():

LATIHAN

1. **Gambar dua buah kotak. Gunakan funsgi translate() untuk merubah posisi dari kotak.**
2. **Modifikasi latihan 1 menggunakan pushMatrix() dan popMatrix() untuk memindahkan atau menggerakkan salah satu kotak saja.**
3. **Gunakan rotate() untuk merubah orientasi bentuk.**
4. **Gunakan scale() dengan "loop for" untuk merubah skala gambar berkali-kali lipat.**
5. **Kombinasikan translate() dan rotate() untuk memutar gambar pada titip pusatnya.**