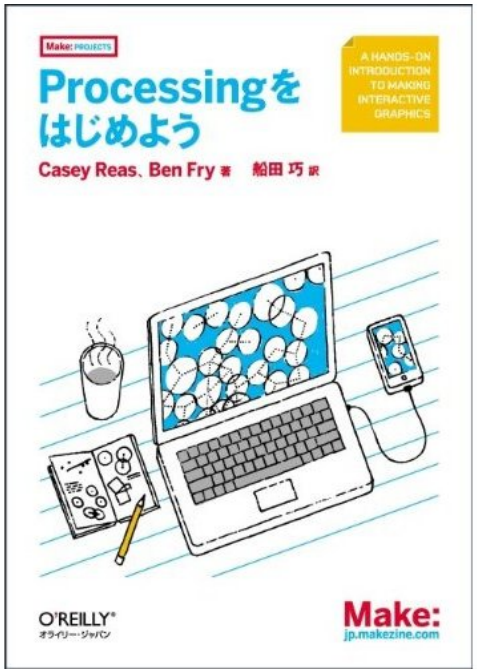
Practice #3

アニメーション（時間的な処理）

演習３Ｂ　ランダムドローイング



Processingをはじめよう
第二版
(Make: PROJECTS)

オンデマンド授業　6.19

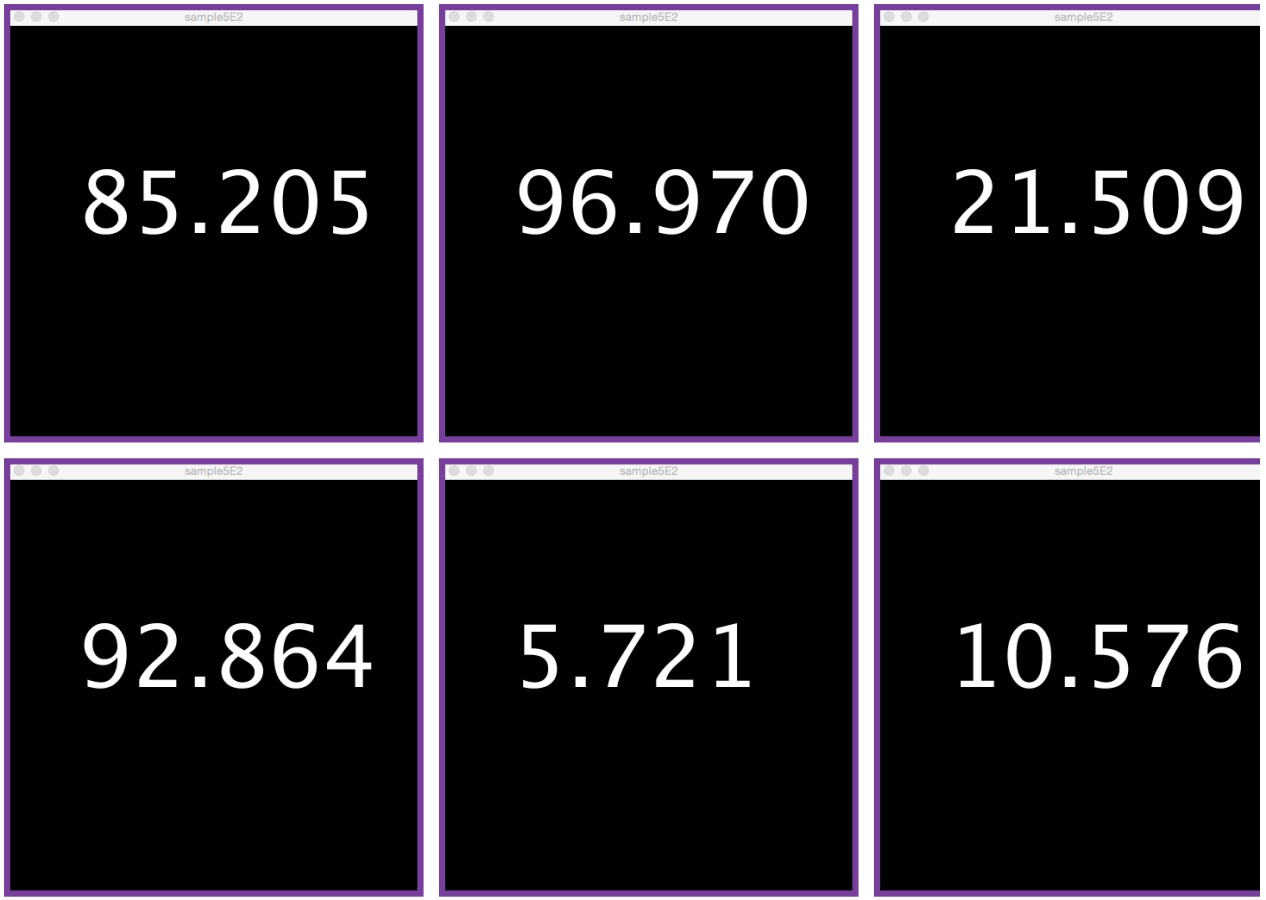
p.64 – 71

p.236 – 239

ランダム関数

sample3B_1.pde

```
1
2  void setup(){
3    size(480,480);
4    background(0);
5    fill(255);
6    textSize(100);
7  }
8
9  void draw(){
10 }
11
12 void mousePressed(){
13   background(0);
14   float r = random(100);
15   text(r,50,240);
16 }
```



float random(n);

0からnまでの乱数を発生させる.

```
12 void mousePressed(){
13   background(0);
14   //float r = random(100);
15   int r = int(random(100));
16   text(r,50,240);
17 }
```

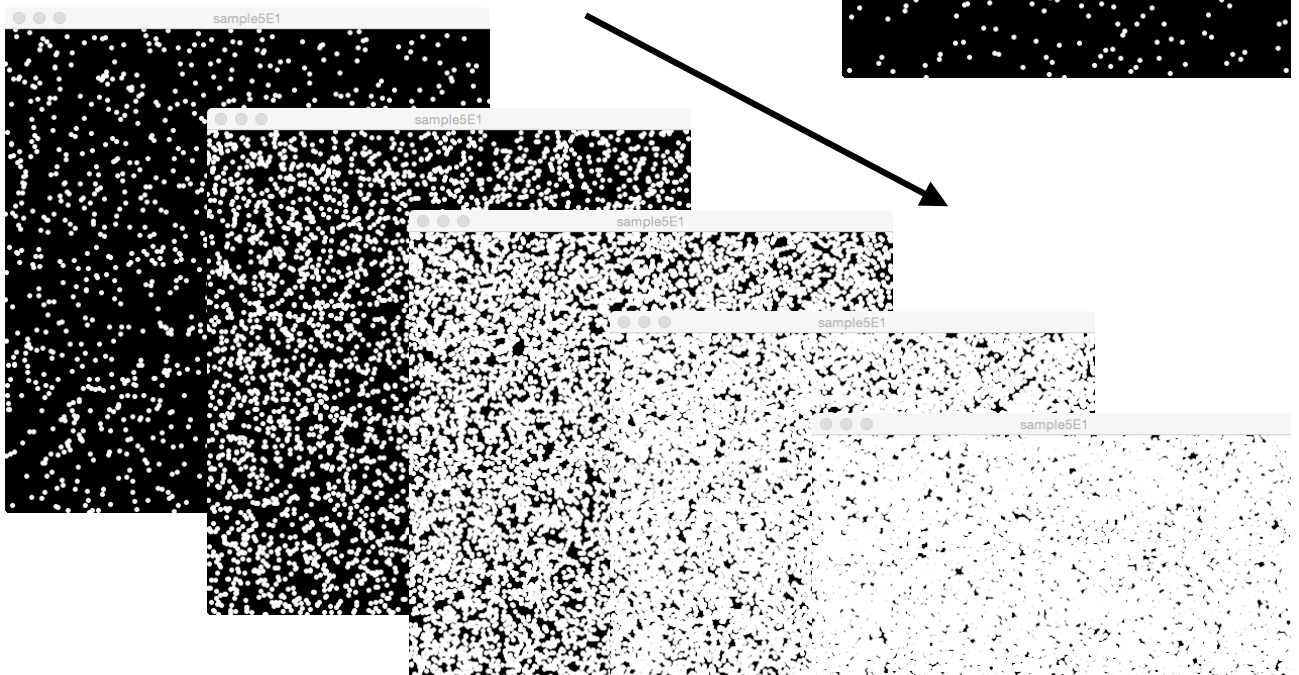
int int(f);

f を整数に変換する.

ランダム関数のビジュアライゼーション

sample3B_2.pde

```
1  int x = 0;
2
3  void setup(){
4    size(480,480);
5    background(0);
6    stroke(255);   strokeWeight(5);
7
8    frameRate(100);
9  }
10
11 void draw(){
12
13   point(x,random(height));
14
15   x++;
16
17   if(x==width){
18     x = 0;
19   }
20
21 }
```



void strokeWeight(n);

線の幅をnピクセルとする

void point(x, y);

（x, y）に点を打つ.

ランダムウォーク

sample3B_3.pde

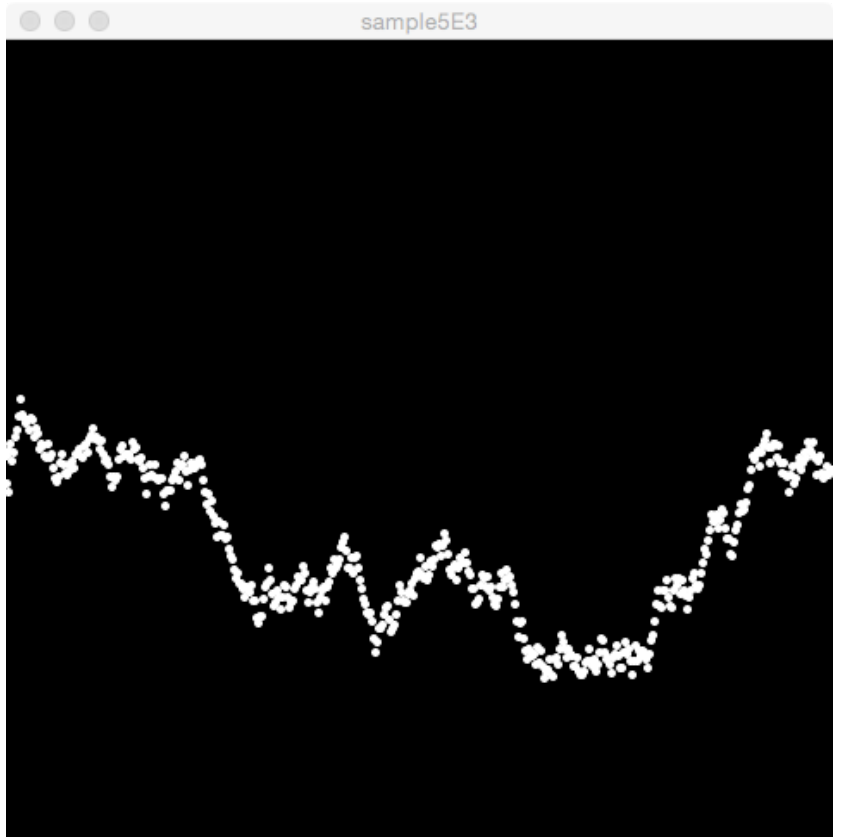
```
1  int x; float y;
2
3  void setup(){
4    size(480,480);  background(0);
5    stroke(255);  strokeWeight(5);
6    x = 0; y = 0.5 * height;
7
8    frameRate(100);
9  }
10
11 void draw(){
12   float n = 10;
13   y = y + random(2*n) - n;
14   y = constrain(y, 0, height);
15
16   point(x,y);
17   x++;
18
19   if(x==width){
20     x = 0;
21   }
22 }
```

以下の計算により，現在のY座標を中心に，最大移動距離（n）でランダムウォークできます．

```
y = y + random(2*n) - n;
```

```
-n< random(2*n) - n; <n
```

```
(y-n)< y + random(2*n) - n; <(y+n)
```




```
float constrain(a,min,max);
```

aをminとmaxの間に収める．
minより小さい場合はminに，
maxより大きい場合はmax，
それ以外はそのままaを返す．

二次元ランダムウォーク

sample3B_4.pde

```
1  float px; float py;
2  float x; float y;
3
4  void setup(){
5    size(480,480);  background(0);
6    stroke(255);  strokeWeight(1);
7
8    px = 0.5 * width; py = 0.5 * height;
9    x = 0.5 * width; y = 0.5 * height;
10
11   frameRate(100);
12 }
```

(px, py)を1フレーム前のポイント, (x, y)を現フレームのポイントとする.

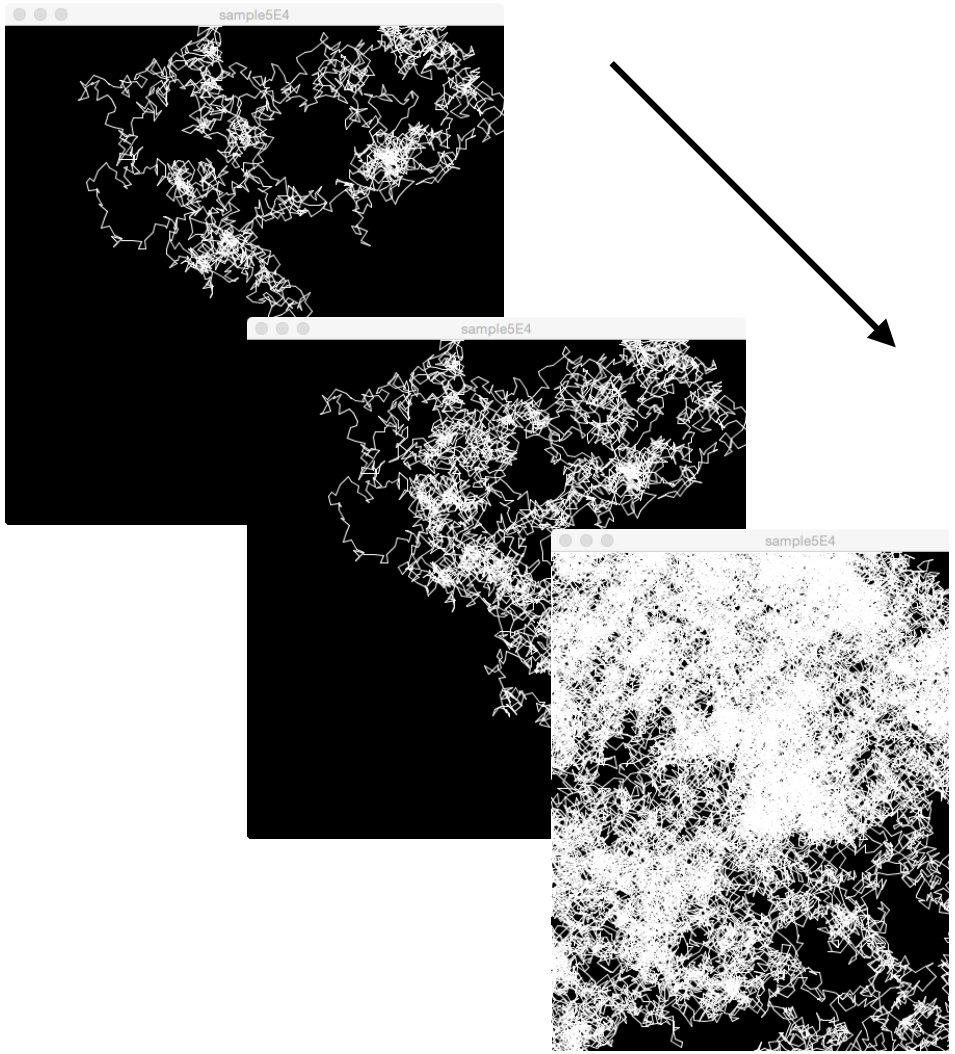
(px, py), (x, y) の初期点をウィンドウの中点とする.

```
14 void draw(){
15
16   float n = 10;
17   x = x + random(2*n) - n;
18   x = constrain(x, 0, width);
19   y = y + random(2*n) - n;
20   y = constrain(y, 0, height);
21
22   line(px,py,x,y);
23   px = x; py = y;
24
25 }
```

x座標のランダムウォーク

y座標のランダムウォーク

(px,py)と(x,y)に線を引き, (px,py)を更新する.



最大移動距離（n）を変更することで, ランダムウォークの細やかさを変更することができます.

二次元ランダムウォーク インサイド サークル

sample3B_5.pde

```
1  float px; float py;
2  float x; float y;
3  float cx; float cy;
4
5  void setup(){
6    size(480,480);  background(0);
7    stroke(255);  strokeWeight(1);
8
9    cx = 0.5 * width; cy = 0.5 * height;
10   px = cx; py = cy;   x = cx; y = cy;
11
12   frameRate(1000);
13 }
```

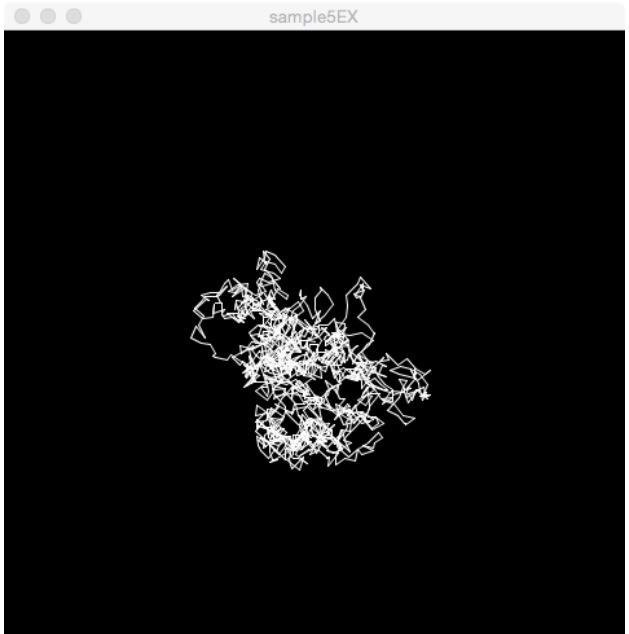
(cx, cy)はウィンドウの中点

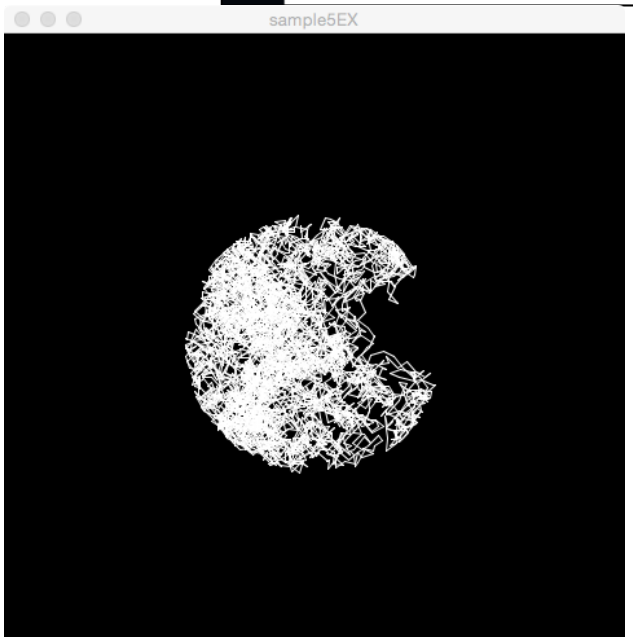
出発点が中点！！

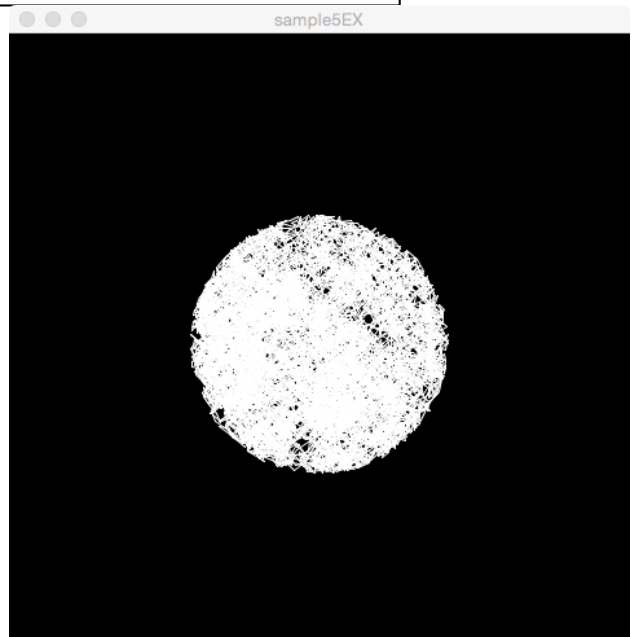
```
15 void draw(){
16
17   float n = 10;
18
19   x = x + random(2*n) - n;
20   y = y + random(2*n) - n;
21
22   if(dist(x,y,cx,cy)>100.){
23     x = px;   y = py;
24   }
25
26   line(px,py,x,y);
27   px = x; py = y;
28
29 }
```

（各軸）最大距離 n のランダムウォーク

(x, y)と(cx,cy)の距離が100を超えた場合，移動をキャンセル

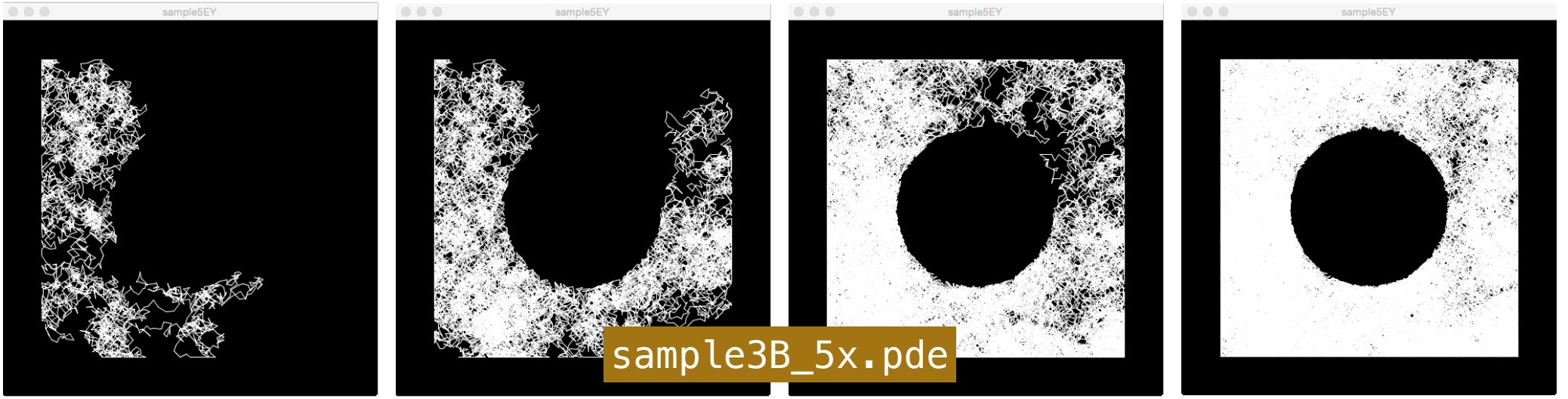






練習

以下のような区間内でランダムウォークをする描画をプログラムしてください．（後半は、課題学習の提出物とします。）



マウスイベント

sample3B_5のdraw()関数の中身を, mousePressed関数の中に移すことによって, マウスのクリックによって, ランダムウォークの描画を呼び出すことができます.

sample3B_6.pde

```
1  float px; float py;
2  float x; float y;
3  float cx; float cy;
4
5  void setup(){
6    size(480,480);  background(0);
7    stroke(255);  strokeWeight(1);
8
9    cx = 0.5 * width; cy = 0.5 * height;
10   px = cx; py = cy;   x = cx; y = cy;
11
12 }
13
14 void draw(){
15 }
```

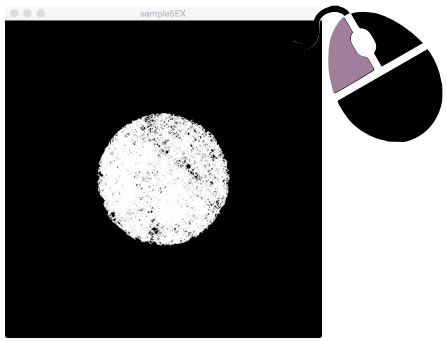
```
17 void mousePressed(){
18   for(int i=0;i<1000;i++){
19
20     float n = 10;
21     x = x + random(2*n) - n;
22     y = y + random(2*n) - n;
23
24     if(dist(x,y,cx,cy)>100.){
25       x = px;  y = py;
26     }
27     line(px,py,x,y);
28     px = x; py = y;
29
30   }
31 }
32
33 void mouseReleased(){
34   background(0);
35 }
```

ランダムウォーク

x 1000

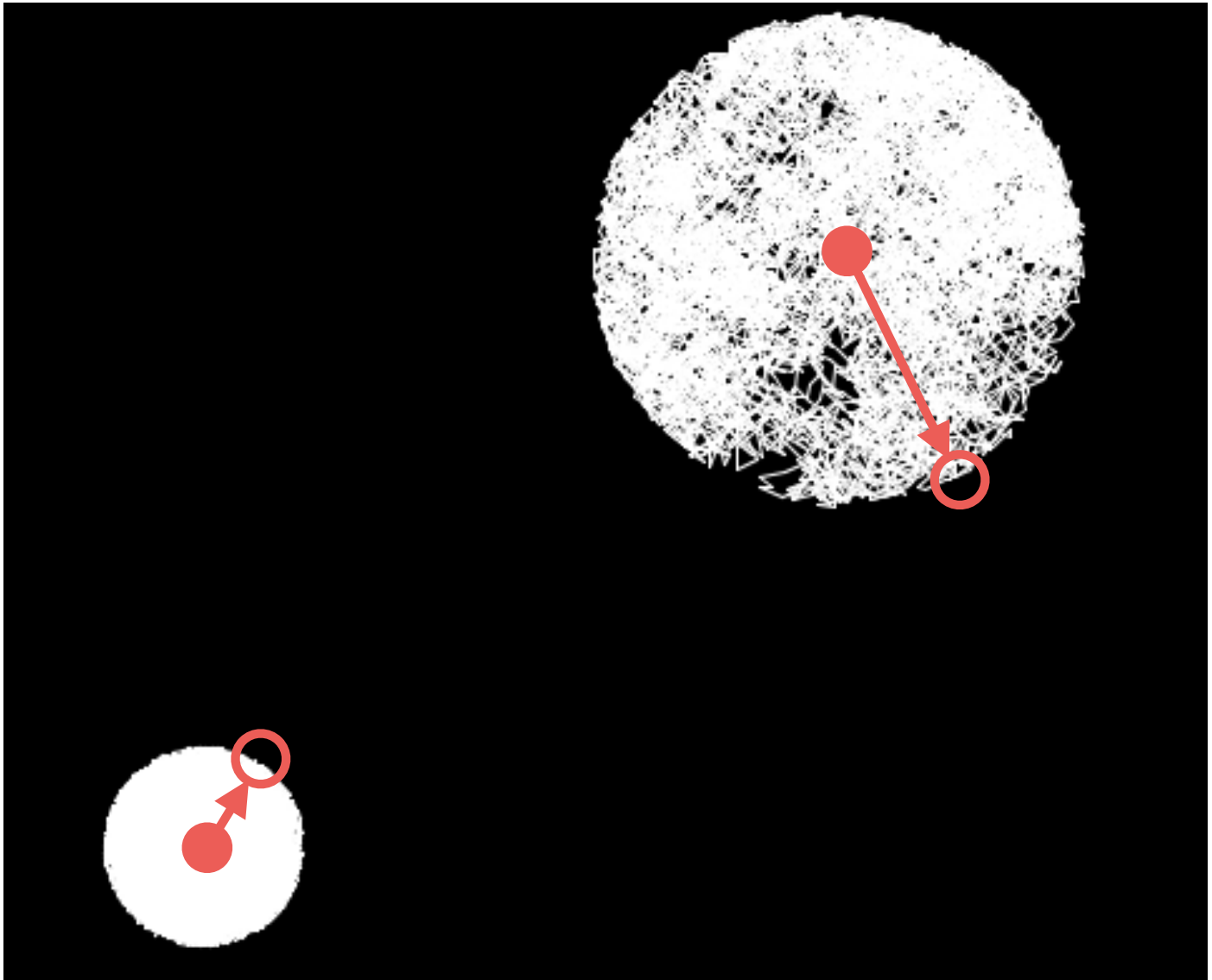
マウスを押した時の処理

マウスを離したときの処理：画面をクリアする

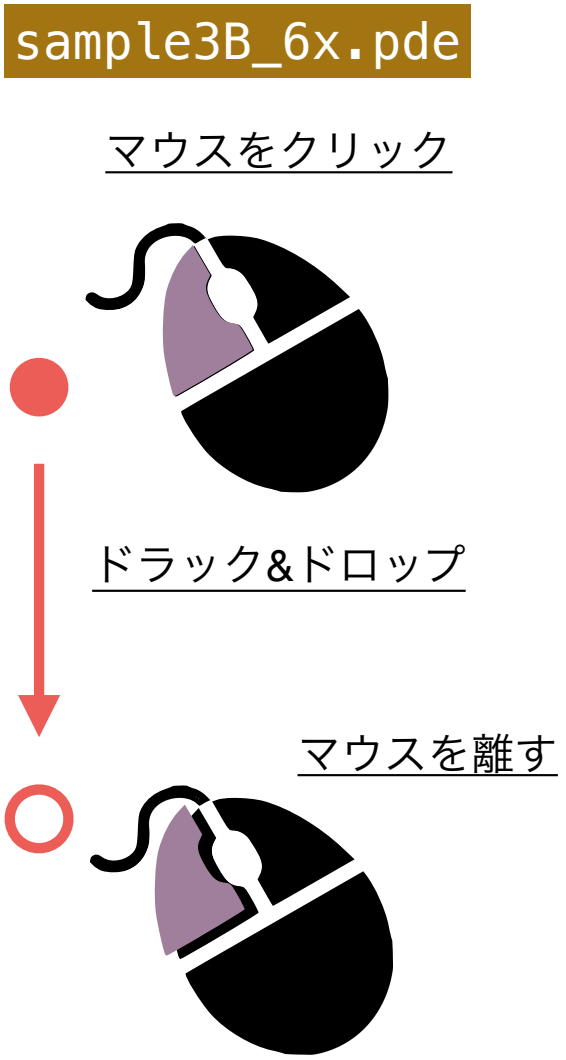


練習

sample3B_6を修正し, 下のように, マウスをクリックした位置を中心とし, その中心から, マウスの離した位置までの距離を半径とする円の内部にランダムウォークで描画を完成させてください.



sample3B_6x.pde



ランダムカラー

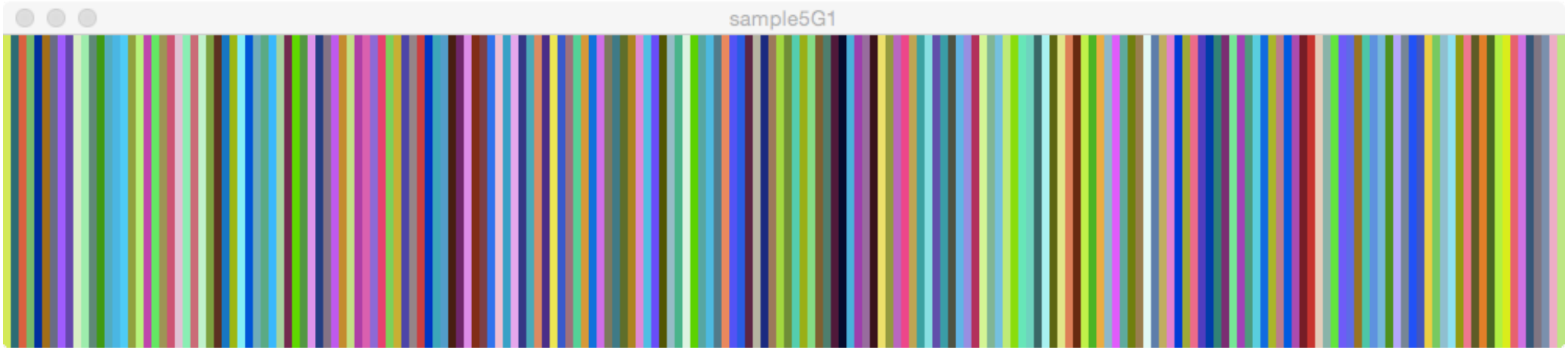
sample3B_7.pde

```
int x = 0;

void setup(){
  size(1000,200);  background(0);
  noStroke();
  frameRate(100);
}
```

```
void draw(){

  float red = random(255);
  float green = random(255);
  float blue = random(255);

  color col = color(red,green,blue);

  fill(col);
  rect(x,0,5,height);
  x = (x+5) % width;

}
```

RGB値のそれぞれをランダムに決定しています.

幅5x高さheightの長方形を,
5pxずつ右にずらしていきます.

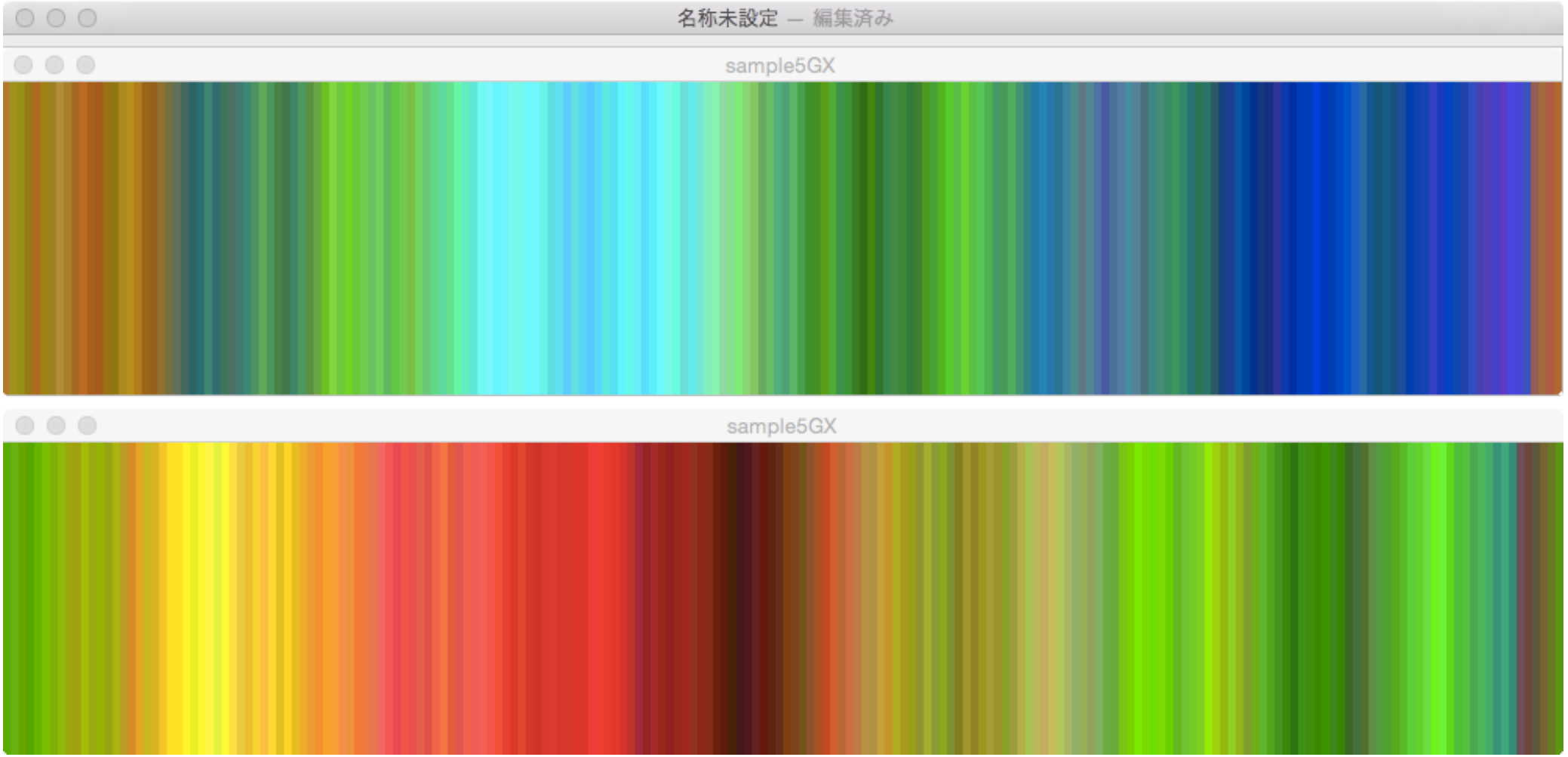


練習１（色のランダムウォーク）

sample3B_7を修正し, 四角形の色が, 徐々に変わるようなパターンに変更してください. この際, RGB値それぞれを, 0-255の区間でランダムウォークさせてください.

`sample3B_7x.pde` `work3B2.pde`

今週の課題２とします。



練習２（色のランダムウォーク）

sample3B_6xを修正して, 線の色もランダムウォークするようにしてください.

`sample3B_7y.pde`