processingで遊ぼう

甲南大学文化会KSWL技術部門
ごつちやん
twitter: @gotutiyan_kapi

今回の概要

- processingを使って、アートを描きます
- 作った作品は、#kc3_2018のハッシュタグで適当に投稿してもらえれば嬉しいです

- インストールの確認

- 今回使用するサンプルコードはgithubにツイートのURLであげます。gitを使える人はcolneすると楽です。

今回の流れ

- processingの仕組み、setup()とdraw()
- translate()とrotate()
- 媒介変数表示から見る数学的アート
- ノイズの有効活用
- 分割することでのアート
- 創作の時間

図形の描画

- ellipse(x,y,r1,r2)・・点(x,y)を中心に縦半径r1,横半径r2の楕円を描きます。r1=r2の時、円が描けます。
- rect(x,y,w,h)・・点(x,y)を左上の点として、横w、縦hの長さで四角を描きます。
- point(x,y)・・点（x,y）に点を打ちます。

- 原点の位置は左上
- y座標は下に行くほど大きくなる

システム変数

- システム変数はいわゆる予約語のこと。ピンク色になります
- width・・実行画面の横の長さ
- height・・実行画面の縦の長さ
- mouseX・・マウスのx座標
- mouseY・・マウスのy座標

ーーーーーーーーーーーーーーーーーーーーーーーーーーーーー

- (int)　key,mouseButton
- (bool) mousePressed,keyPressed,

translate()とrotate()（000.pde)

- 結構大事な原点に関する操作です。よく使います。
- translate(x,y)・・原点の位置をx,yに変える
- rotate(x)・・原点を中心にxラジアン回転する

```
1 size(500,500);
2 ellipse(0,0,100,100);
3 translate(width/2,height/2);
4 ellipse(0,0,100,100);
```

setup()とdraw()

- setup()・・実行した一番最初のフレームで実行
- draw()・・実行中毎フレーム実行

- ・frameRate(整数)・・フレームレートを設定
- size(横の長さ, 縦の長さ)・・実行画面のサイズを指定
- background(R,G,B)・・背景色を設定
- fill(R,G,B)・・塗りつぶす色を設定
- stroke(R,G,B)・図形の輪郭線を設定

drawの中で動かすということ (001.pde)

```
1  int x=0;
2  void setup(){
3    size(500,500);
4  }
5
6  void draw(){
7    background(0);
8    ellipse(x,height/2,100,100);
9    x++;
10 }
```

画面の中心周りに円を回転

- 円軌道は三角関数を使います。x座標をcos、y座標をsinにします。
- $x=\cos\theta$,　$y=\sin\theta$
- 三角関数を使うことを前提に、２つの方針を立てることができます。
- 方針１：画面の中心にtranslate()、そこからさらにtranslate()を繰り返して円を描く。
  - translate(width/2,height/2);→ellipse(x,y,10,10);
- 方針２：画面の中心にtranslate()、そこからx=cos(),y= sin()として円を描く。
  - translate(width/2+x, height/2+y);→ellipse(0,0,10,10);

方針１（002.pde)

8: 画面中心に原点を移動

9,10: xとyを更新

11: (x,y)を中心に円を描く

12: ラジアンを増やす

```
1  float x,y,theta=0;
2  void setup(){
3    size(500,500);
4  }
5
6  void draw(){
7    background(0);
8    translate(width/2,height/2);
9    x=200*cos(theta);
10   y=200*sin(theta);
11   ellipse(x,y,20,20);
12   theta+=PI/128;
13 }
```

方針 2 (003.pde)

8: 画面中心に原点を移動

9,10: xとyを更新

11: (x,y)に原点を移動

12: (0,0)を中心に円を描く

13: ラジアンを増やす

どちらでも、好きな方で
実装してみましょう～

```
1  float x,y,theta=0;
2  void setup(){
3    size(500,500);
4  }
5
6  void draw(){
7    background(0);
8    translate(width/2,height/2);
9    x=200*cos(theta);
10   y=200*sin(theta);
11   translate(x,y);
12   ellipse(0,0,20,20);
13   theta+=PI/128;
14 }
```

円の数を増やす

- 円の数を増やすことで、より軌跡を分かりやすくします。円ごとに足すラジアンを変えるようにしましょう。

- 円の数を増やすには、「クラス」が使うとすごく楽ですが、今回は配列でやることにします。

- さて、次からは「リサージュ曲線」を扱います。

媒介変数表示から見る数学的アート

- 先ほど：

```
x=cos(θ);
y=sin(θ);
```

- リサージュ曲線:

```
x=sin(3*θ);
y=sin(4*θ);
```

媒介変数表示から見る数学的アート (004.pde)

$$x=\sin(3*\theta);$$
$$y=\sin(4*\theta);$$

全ての変数名を配列に

11,12: x,yに代入するものをリサージュ曲線の媒介変数表示に合わせる。

14: θを更新する。(i+1)は動かない点をなくすため。

```
1  float x[]=new float[50],y[]=new float[50];
2  float theta[]=new float[50];
3  void setup(){
4    size(500,500);
5  }
6
7  void draw(){
8    background(0);
9    translate(width/2,height/2);
10   for(int i=0;i<50;i++){
11     x[i]=200*sin(3*theta[i]);
12     y[i]=200*sin(4*theta[i]);
13     ellipse(x[i],y[i],20,20);
14     theta[i]+=(i+1)*PI/128/100;
15   }
16 }
```

媒介変数表示から見る数学的アート

- 他にも色々な曲線があります。実装が簡単なものを挙げます。
- アステロイド曲線

$$x=\cos^3(\theta);$$
$$y=\sin^3(\theta);$$

- インボリュート曲線

$$x=\cos(\theta)+\theta\sin(\theta);$$
$$y=\sin(\theta)-\theta\cos(\theta);$$

ノイズを使ったアート

- ノイズの登場です。ノイズはランダムに値を生成しますが、引数にとる値によって「ランダムさ」をある程度制御できます。

- の前に、まずはprocessingにおけるランダム：

```
random(最小値, 最大値+1);
```

- float型の値を返します。第２引数自体の値は範囲に含まれないので、0~100が欲しければrandom(0,101)とします。
- （最小値が0であれば、省略できてrandom(101)ともできます）

ノイズを使ったアート

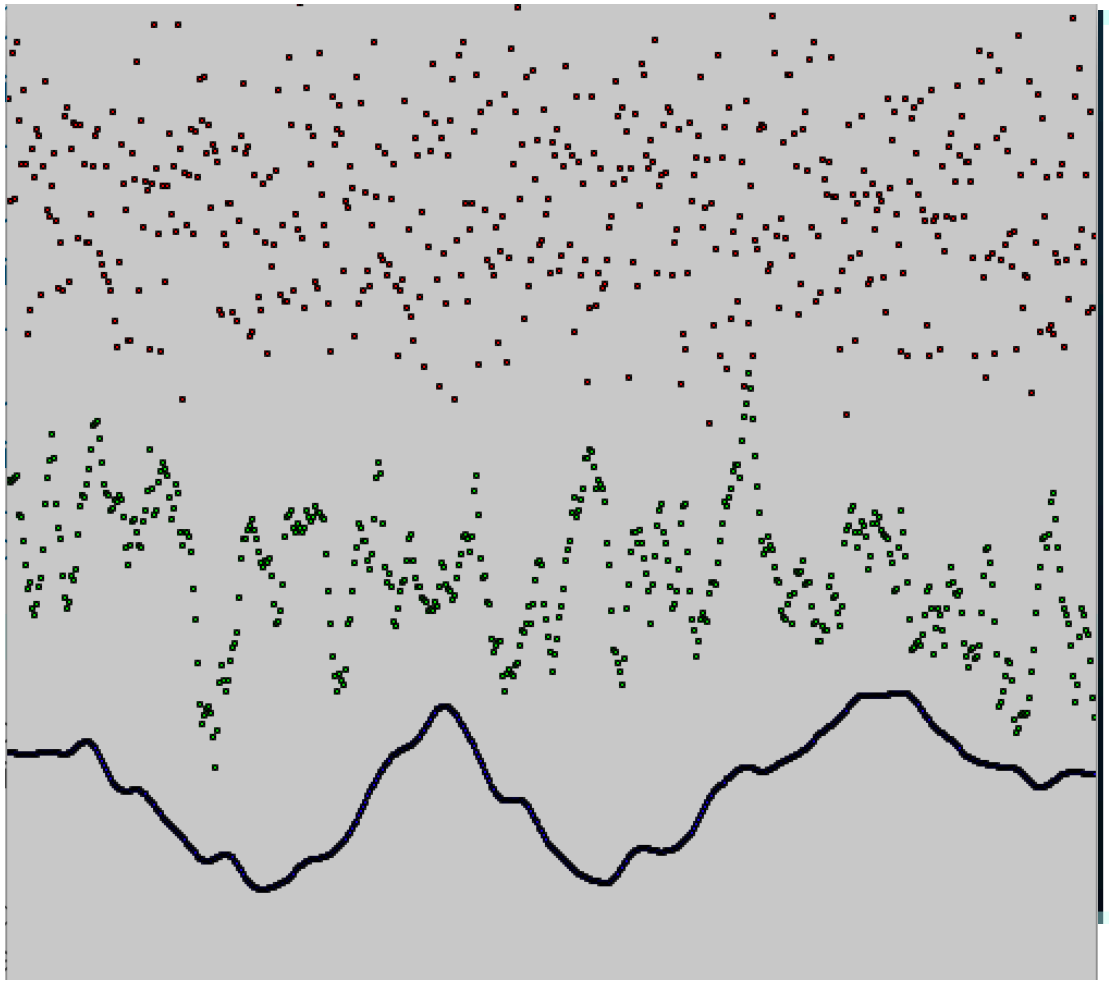
- 次にノイズです。

```
noise(シード値);
```

- noise()は0~1のfloat型の値を返します。
- シード値を与えて使用しますが、どんな値が返るかは分かりません。しかし、あるシード値を基準として、それに近いシード値を与えれば近い値が返り、離れたシード値を与えれば離れた値が返ります。

- 例：noise(100)=0.475のとき、
  noise(100.01)=0.479, noise(101)=0.621

ノイズを使ったアート (005.pde)



```
size(500,500);
stroke(0.1);
for(int i=0;i<width;i++){
  pushMatrix();
  translate(0,height/4-150);
  fill(255,0,0);//red
  ellipse(i,250*noise(100+i),2,2);

  translate(0,height/4+50);
  fill(0,255,0);//green
  ellipse(i,250*noise(100+i*0.1),2,2);

  translate(0,height/4);
  fill(0,0,255);//blue
  ellipse(i,250*noise(100+i*0.01),2,2);
  popMatrix();
}
```

ノイズを使ったアート

- ノイズは「自然物」を表現するのによく使われます。この微妙にブレる感じが、人の自然な手書きの様子を表現したり、炎や雲などの自然物を表現するのに役立っています。

- ノイズの次元・・連続的にさせたい方向がいくつあるか
  - 手書き風の線、ランダムなグラデーション：１次元
  - 雲のある青空：２次元
  - ３Dゲームで洞窟をランダムで生み出す：３次元

休憩

雲がある青空　(006a.pde)

- 雲がある青空

２次元ノイズを使用

12,13: ノイズの値を色のR,G,Bに利用

8,10: ノイズのシード値を元に戻す（重要）

```
1  float noiseX=0,noiseY=0;
2  void setup(){
3    size(500,500);
4  }
5
6  void draw(){
7    background(0);
8    noiseY=0;
9    for(int i=0;i<height;i++){
10     noiseX=0;
11     for(int j=0;j<width;j++){
12       float c=noise(noiseX,noiseY);
13       stroke(255*c,130+125*c,255);
14       point(i,j);
15       noiseX+=0.01;
16     }
17     noiseY+=0.01;
18   }
19 }
```

流れる雲がある青空　(006b.pde)

1,8,10,19行目を追加する

noiseX,noiseYを元に戻す値(seed)を0.01ずつ増やしていく

色を変えると変わる雰囲気

13行目をstroke(255*c,0,0);に・・？

```
1  float noiseX=0,noiseY=0,seed=0;
2  void setup(){
3    size(200,200);
4  }
5
6  void draw(){
7    background(0);
8    noiseY=seed;
9    for(int i=0;i<height;i++){
10     noiseX=seed;
11     for(int j=0;j<width;j++){
12       float c=noise(noiseX,noiseY);
13       stroke(255*c,130+125*c,255);
14       point(i,j);
15       noiseX+=0.01;
16     }
17     noiseY+=0.01;
18   }
19   seed+=0.01;
20 }
```

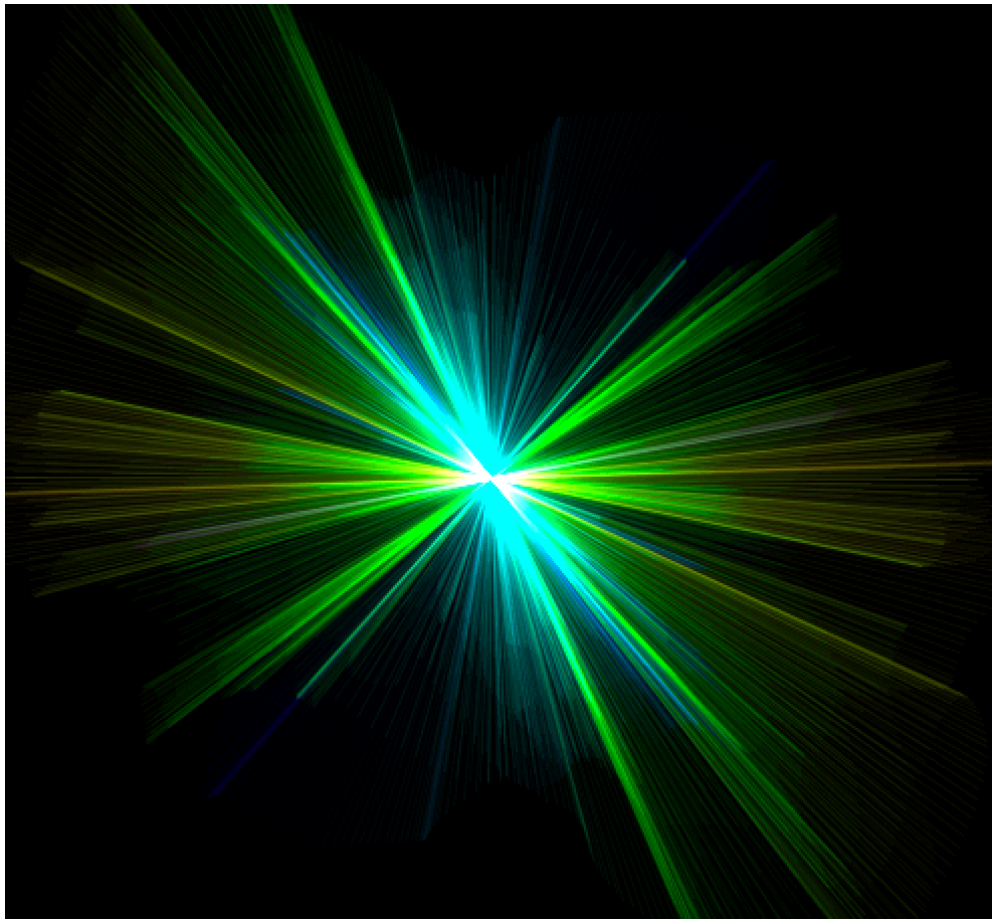
他のノイズ使用例　(007a.pde)

中心部を発光させたいので、blendMode(ADD);
HSBで色を決定したいので、colorMode(HSB);

画面中心に原点を移動
x,y座標をnoise()で算出
色もnoise()を使って算出
line(x,y,-x,-y)で、原点対称な２点間に線を引く

stroke()の第４引数に10前後の値を入れることで
線がぼやけて良い感じに

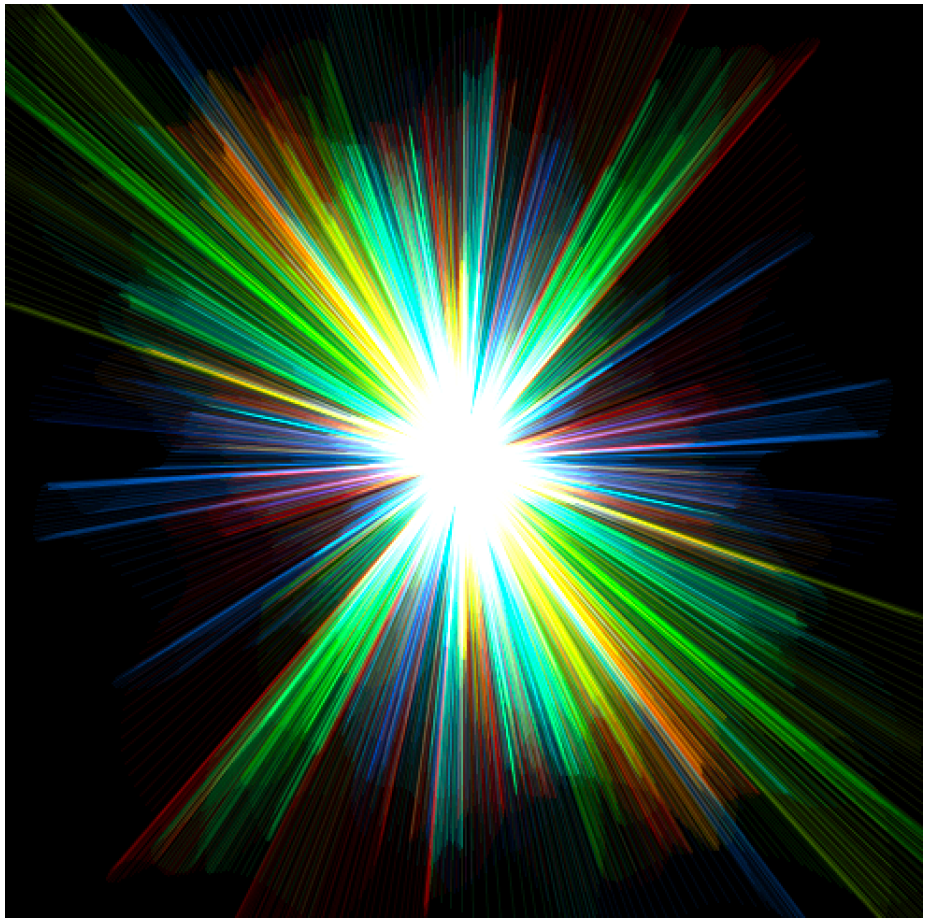
map(a,b,c,d,e)・・値aを範囲[b-c]から範囲[d-e]に移す



他のノイズ使用例　(007b.pde)

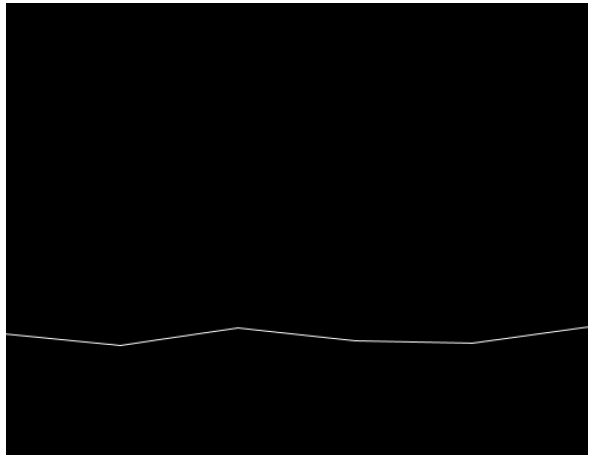
先ほどの例では赤系統の色が現れにくい
→noise()が1や0など「範囲の端」の値を取りにくいから。

範囲を極端に大きくして、255での剰余をとることで全ての色が出やすく。



分割することでのアート　(008.pde)

- １本の線を考える
- 線に中間点をいくつか設定し、その点をnoise()や三角関数を使って揺らす。これは線を分割するという発想。



```
float seed=0;
void setup(){
  size(500,500);
}

void draw(){
  background(0);
  stroke(255);
  for(int i=0;i<5;i++){
    line(100*i,height/2+100*noise(seed+i*10),
            100*(i+1),height/2+100*noise(seed+(i+1)*10));
  }
  seed+=0.01;
}
```

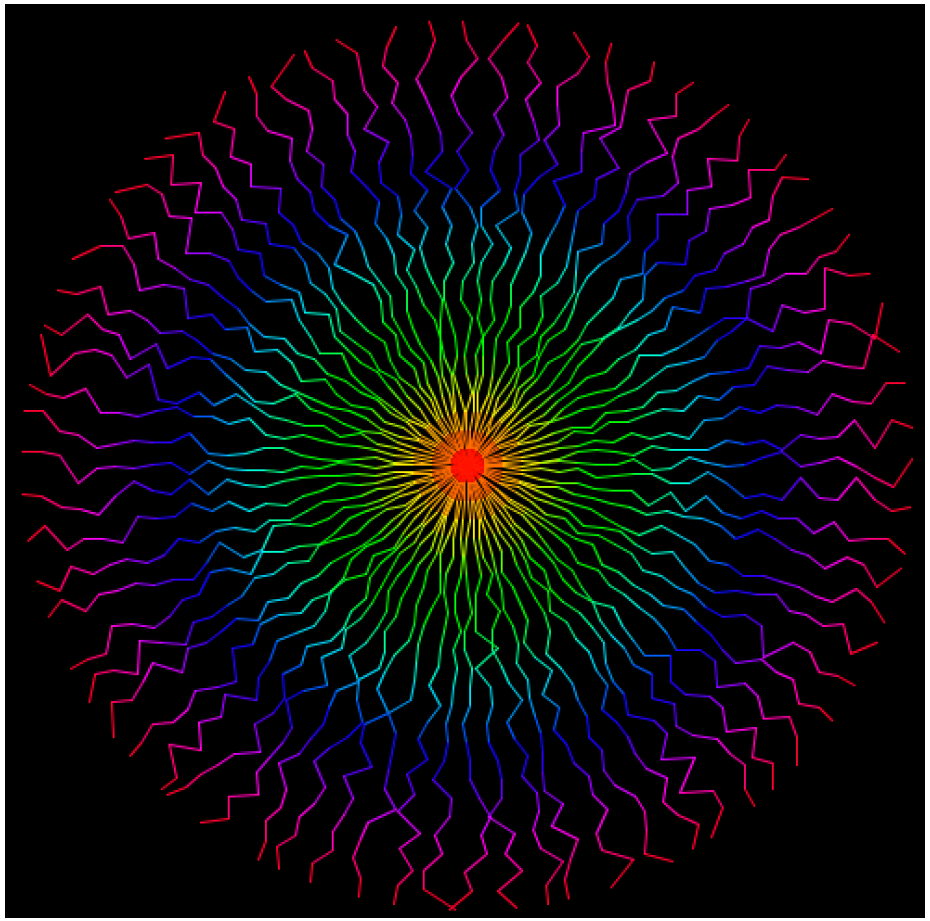
分割することでのアート　(009.pde)

008.pdeで描いた線を、放射線状に並べて、色をグラデーションするようにした。

009.pdeでは、線を何本配置するかのint n、分割する点を幾つ設定するかのint rを作った。

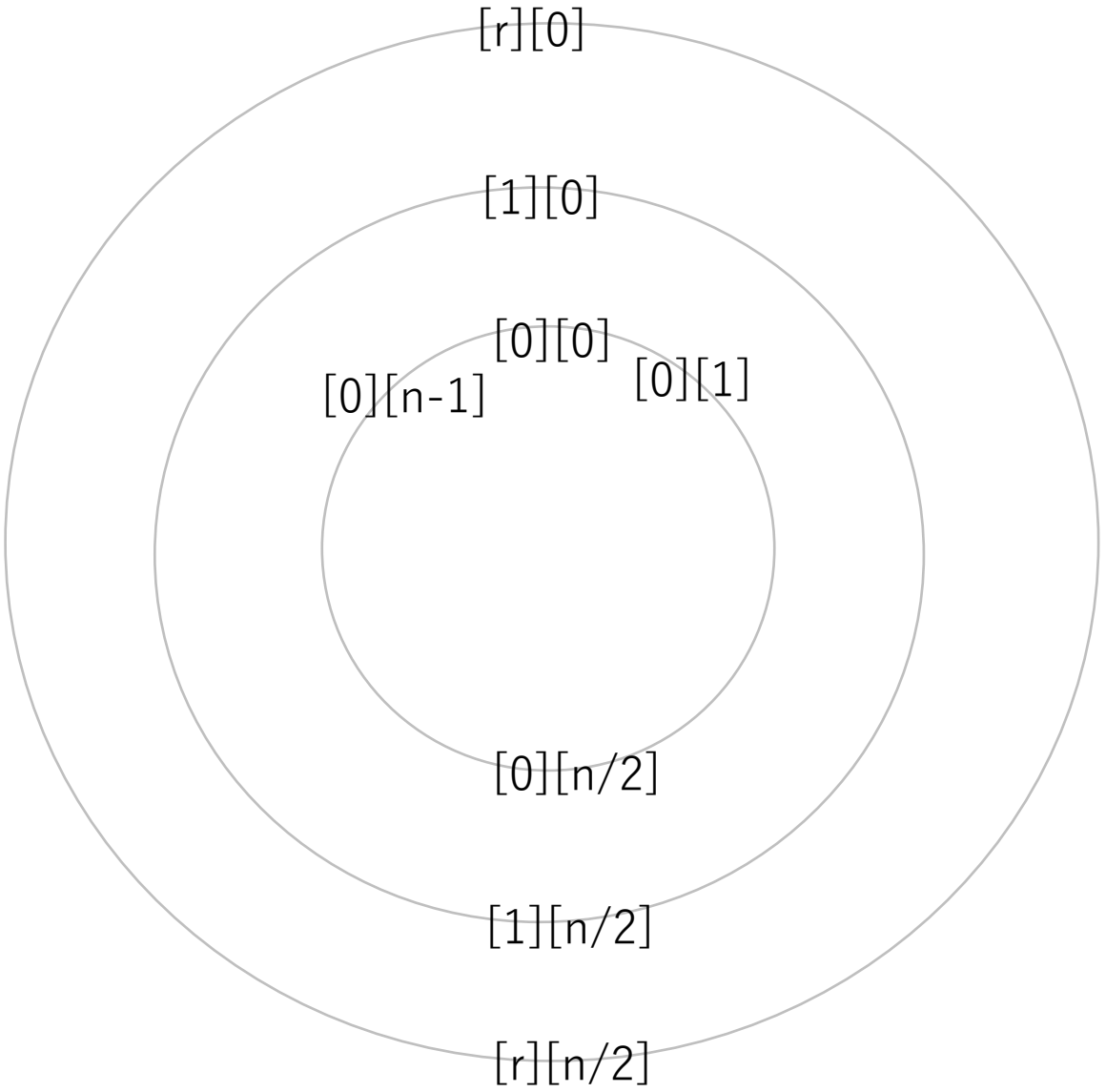
nとrの値を変えてみよう。

n=100にして、background(0);をコメントアウトしてみると・・？



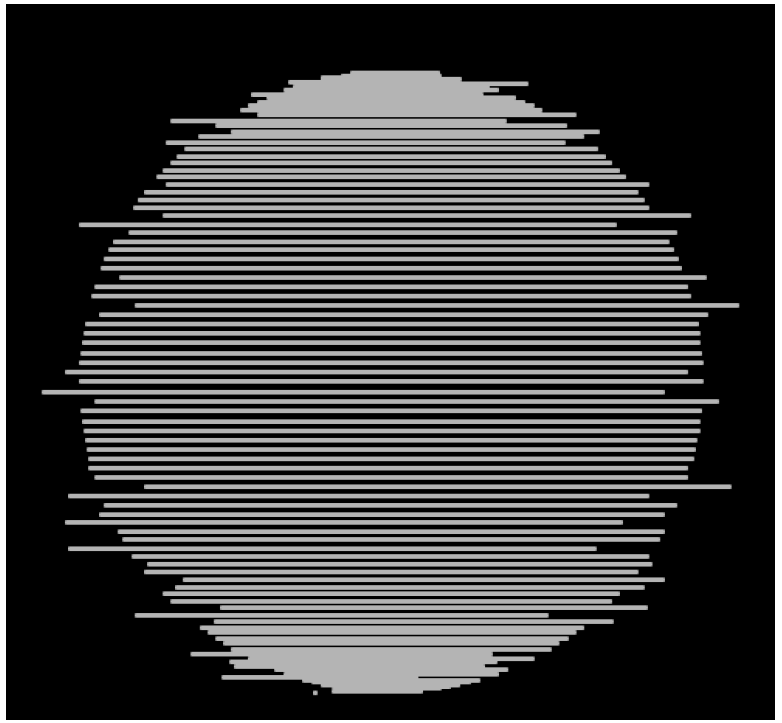
作り方 (009.pde)

- 配列の添字と点は以下のように対応する。
- あとはこの点の数だけseed[][]も作り、それを利用してnoise()でランダムにずらす。
- 塗る色は中心からの距離で変えるので、添字の[ここ][]の値を利用してHSBで塗る。



分割することでのアート (010.pde)

- 円を分割する：蚤気楼
- 円をスライスするように分割し、ノイズでそれぞれを揺らす。



作り方 (010.pde)

- 最初に三角関数で線を引くための片方の端の点を配列に格納。この時、角度θは90~270で算出。($0 \leq x \leq -1, -1 \leq y \leq 1$)

- さらに、線をずらすためのノイズを作成するにあたり、実際に線をずらすのはノイズの値が0.65より大きいか、0.35より小さい時だけにする。
- noise()<0.35 または 0.65<noise()
- 0<noise()<0.35… -100~0
- 0.65<noise()<1…0~100

- mapを使って飛んでいるノイズの範囲を-100~100の範囲に広げる。

ジェネラティブ・アート

- ジェネラティブ・アートは概念。
- 実行するたびに異なる描画が行われる→noise()やrandom()によるランダムな値の利用が重要。マウス座標を使うのもあり？
- 動的なアートでも、その中で一瞬の様子を静止画として切り取れば、毎回違うアートが現れることになる。

創作の時間

- ここまでにやってきたnoise()などを使ってみて、自分なりにアートを作ってみよう。どんなアートを作るかは、「図形を分割すること」や「グラデーションすること」、「回転させること」などに 発想を得ると良いかもしれない。

- 作ったアートは、twitterの#kc3_2018にて、静止画や動画として投稿してもらえると嬉しいです。