コンピュータ基礎演習

第9回

理工学部 情報科学科 隅田 康明

sumida@ip.kyusan-u.ac.jp

遠隔授業期間中の質問

- **困ったら早めに質問・相談！！**
  - 学生側から質問されないと、
    誰が困っているのか、何が分からないのか、分かりません
- メール：やり取りに時間はかかるが一番確実
- Zoom：授業時間中限定
  - 時間は限られるが、作業中の画面を見ながら教えられるので、問題を短時間で解決出来る可能性が高い
- Line OpenChat：授業時間中限定
  - 文字だけのやり取りに限定（画像アップロードは禁止）
  - 質問内容が他の学生にも分かるので注意すること
    - プログラムの全文貼り付け等は厳禁！

授業についての質問メールについて

[授業名(曜日時限)]についての質問	件名
～先生	誰宛か
[授業名(曜日時限)]を受講しています、 20AA999の九産太郎です。	何者か
（質問内容） --	用件
20AA999 九産太郎 九州産業大学　芸術学部　○○学科 1年	署名

2日返信がなければもう一度送って下さい（なるべく見落としたくない）

レポートについての注意

- 雛型と同じものを出してきても０点
  - 当たり前ですが
- ちゃんと自分なりに考えてアレンジすること

今回の授業内容

- 繰り返し、メソッド、座標変換
  - 円形模様メソッドを繰り返しで作成
  - 繰り返しで円形模様を並べる

  - 内容は多いが、やることはそこまで難しくはない
    - 手軽にオリジナル図形を作って敷き詰める

- レポート内容
  - 雛型プログラムのアレンジ
    - 雛型をアレンジしながら慣れていけば良い

for文：繰り返し

- 構文

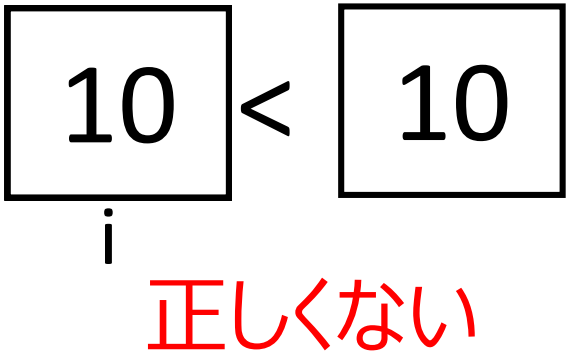
```
for(ループ変数の初期化; 条件; 更新) {
    繰り返したい処理;
}
```

- ループ変数の初期化：変数を宣言し、初期値を代入
  - 例）int i = 0; //int型の変数iを宣言し0を代入
- 条件：繰り返す条件を論理式で書く
  - 例） i < 10; // iが10より小さい間繰り返す
- 更新：ループ変数を増やす、減らすなど
  - 例）i++ //iを1増やす

繰り返しの例

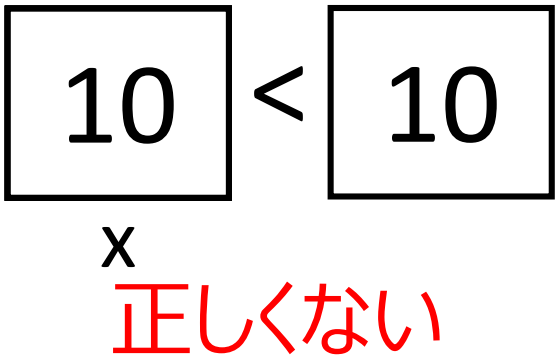
- i を0から10まで 1 ずつ増やして表示する

```
for(int i = 0; i < 10; i++) {
  print(i + " ");
}
```



- x を0から10まで 2 ずつ増やして表示する

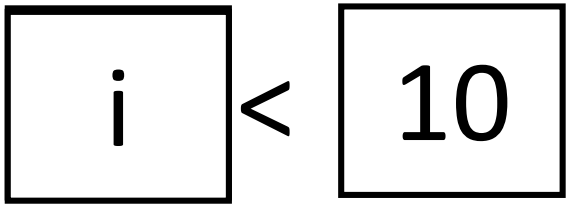
```
for(int x = 0; x < 10; x+=2) {
 print(x + " ");
}
```



繰り返しの例

- i を0から10まで 1 ずつ増やして表示する

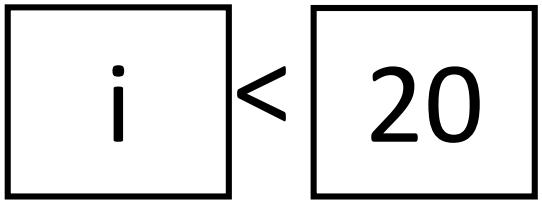
```
for(int i = 0; i < 10; i++) {
  print(i + " ");
}
```



i と 10を比較

- x を0から20まで 2 ずつ増やして表示する

```
for(int x = 0; x < 20; x+=2) {
 print(x + " ");
}
```



i と 20を比較

繰り返しで図形を描く

```
void setup(){
  size(400,400);
  for(int x = 0; x <= 10; x++) {
    ellipse(x*50, 10, 20,20);  //xは50ずつ増える
  }
}
```

- 数値を色々変えて、図形がどう描かれるか確認しよう
  - 適当に変えるのではなく、
    何故そうなるのかを考えること

今回は繰り返しで静止画を描いていく

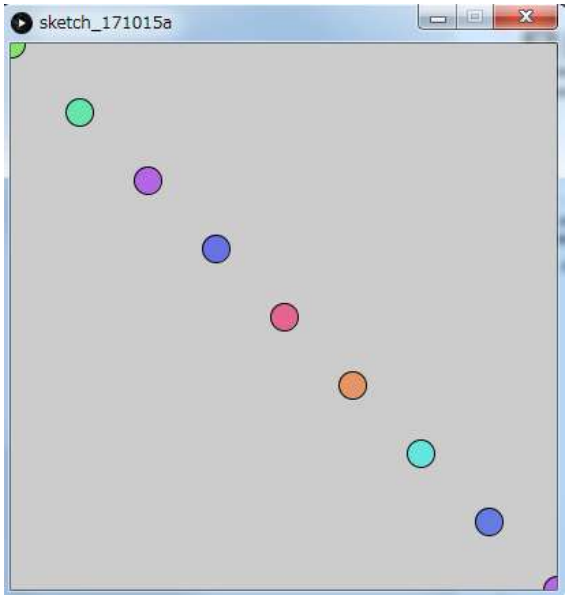
繰り返しながら色を変える

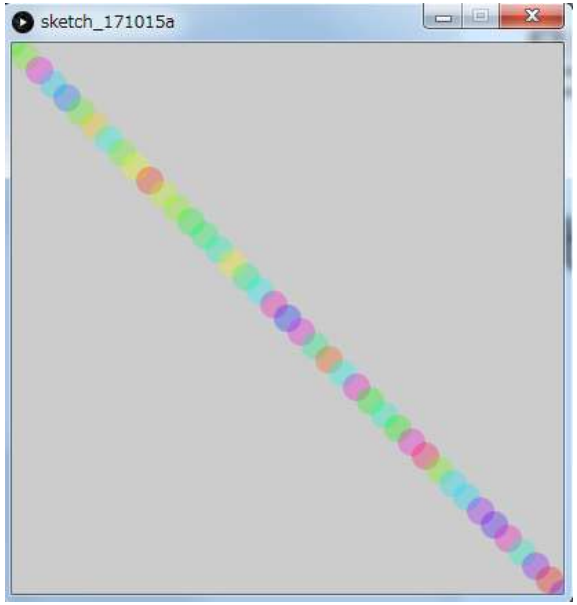
- xは 0, 20, 40, 60, 80, 100, 120 …と増えていく
  - 段々と色が変わっていく

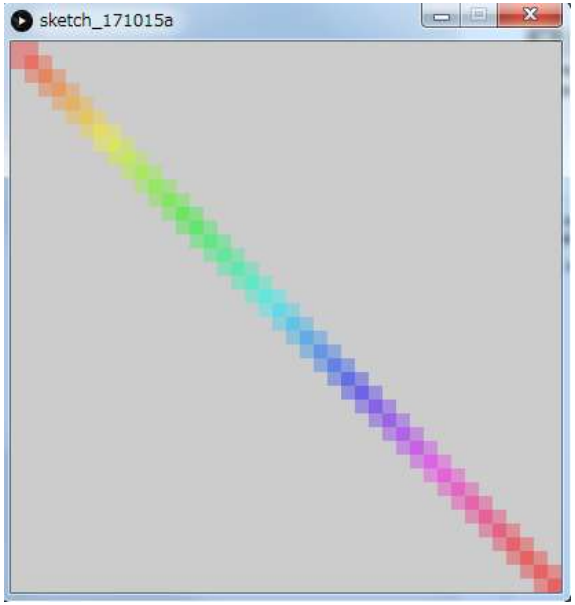
```
void setup(){
  size(400,400);
  noStroke();
  colorMode(HSB, 360, 100, 100, 100);
  for(int x = 0; x <= width; x+=20) {
     fill( x , 100, 100, 50);
     ellipse(x , 10, 20,20);
  }
}
```

問題：斜めに図形を並べてみる

- ヒント
  - y座標が固定されていたら、
    縦方向の座標はいつもに同じ
  - y座標にもx座標と同じ値を使ってみよう
- 小テストの問題：自分でも実行して試してみよう







縦横に丸を並べる

- 横1列に丸を並べる　⇒　for を使って繰り返す
- 縦1列にも丸を並べる　⇒　for文を沢山書けばいい？
  - 縦に1000列並べろと言われたら？
    ⇒　for文自体を繰り返せばいい

多重ループ（for文の中にfor文）

- 横1列に丸を描く

この繰り返し文を繰り返す

```
for(int x = 0; x <= width; x+=20) {
  ellipse(x , 10, 20,20);
}
```

```
for(int y = 0; y <= height; y+=20) {
    for(int x = 0; x <= width; x+=20) {
      ellipse(x , y , 20 , 20);
    }
}
```

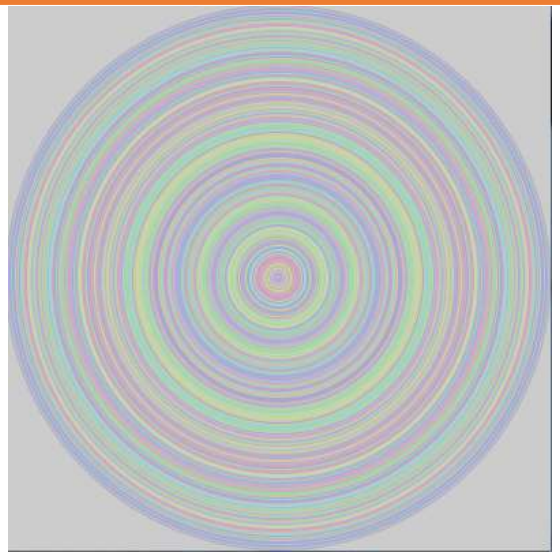
y軸方向の繰り返し

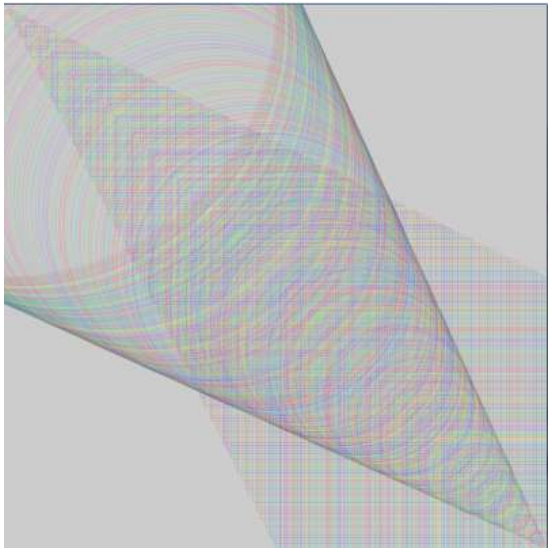
x軸方向の繰り返し

図形を敷き詰めるときのパターンとして覚え、数値を変えて試して慣れていく

今回の小テストではなし。次回以降に出題予定。

問題：繰り返しで図形を配置

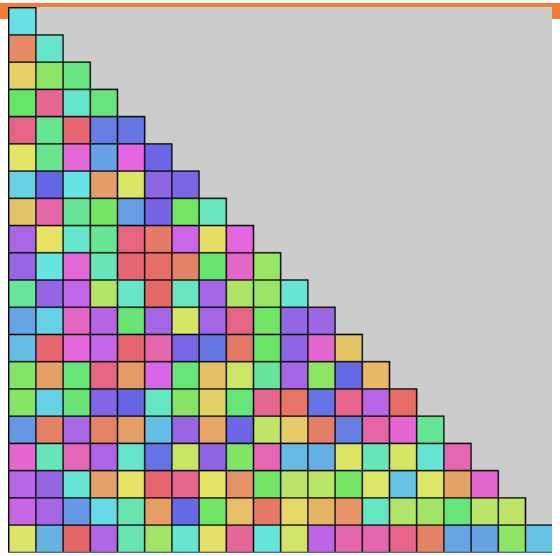




- 問題1：大きくなる円
  - 中心座標は固定
  - 幅、高さをループ変数に
- 問題2：斜め配置の組み合わせ
  - 繰り返しを2回書く
    - 左上から右下への四角
    - 左上から右下への丸

今回の小テストではなし。次回以降に出題予定。

問題3：階段状に配置

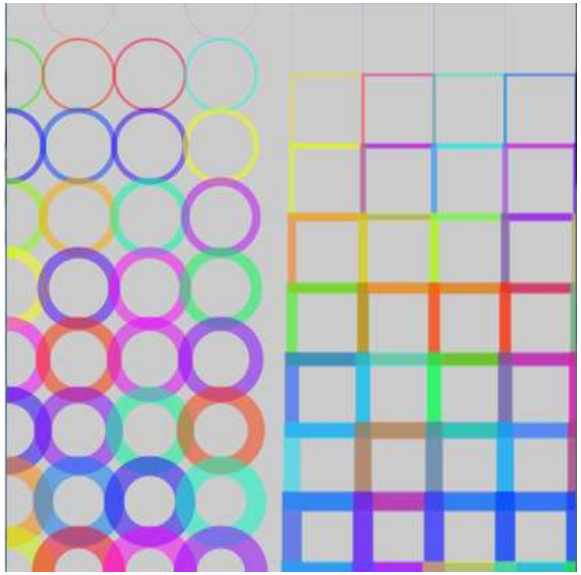


- 原型は2重ループのプログラム
- xの繰り返し条件がポイント
- 1段目は1個分の幅まで
- 2段目は2個分の幅まで
- 3段目は3個分の幅まで
- y段目は??個分の幅になる？

```
for(int y = 0; y <= height; y+=20) {
      for(int x = 0; x <= ???; x+=20) {
        rect(x , y , 20 , 20);
      }
}
```

今回の小テストではなし。次回以降に出題予定。

問題４：左右で違う図形を配置



- 2重ループを2回書く
  - 左側の丸の繰り返し
  - 右側の四角の繰り返し

円形模様メソッドの作成

丸や四角ではなく、自分で作った円形模様を並べる

座標変換

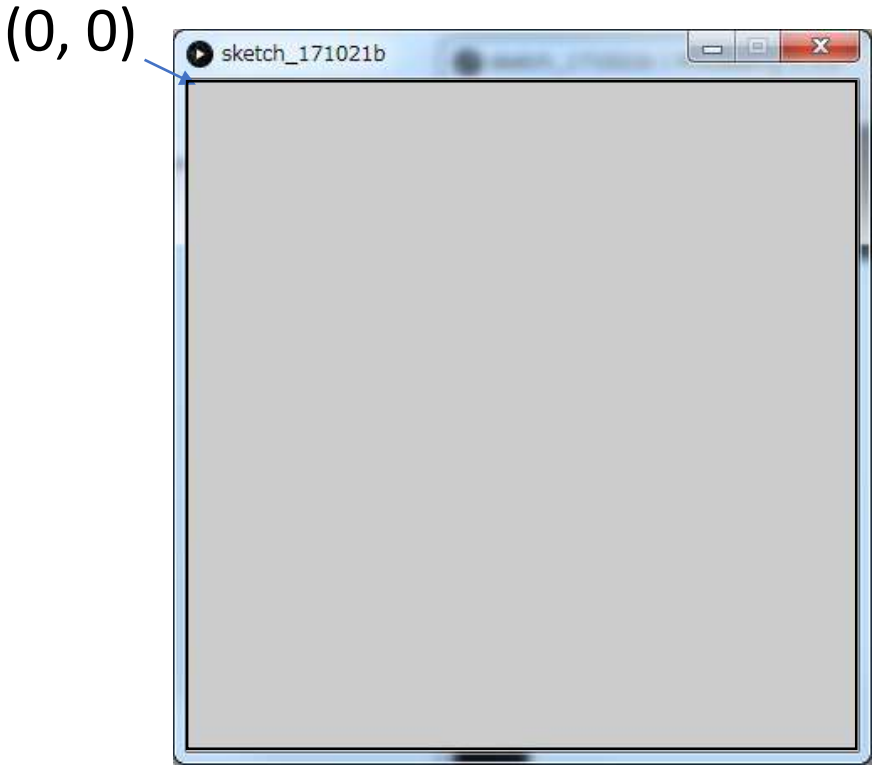
説明を見てよく分からなくてもレポートには支障はない

（使いながらでないと理解するのは難しい）

表示領域と描画キャンバス（座標系）

正しい呼び方

- 実行画面の座標と、描画キャンバスの座標が別々
  - 普段はどちらも実行画面の左上が (0, 0)



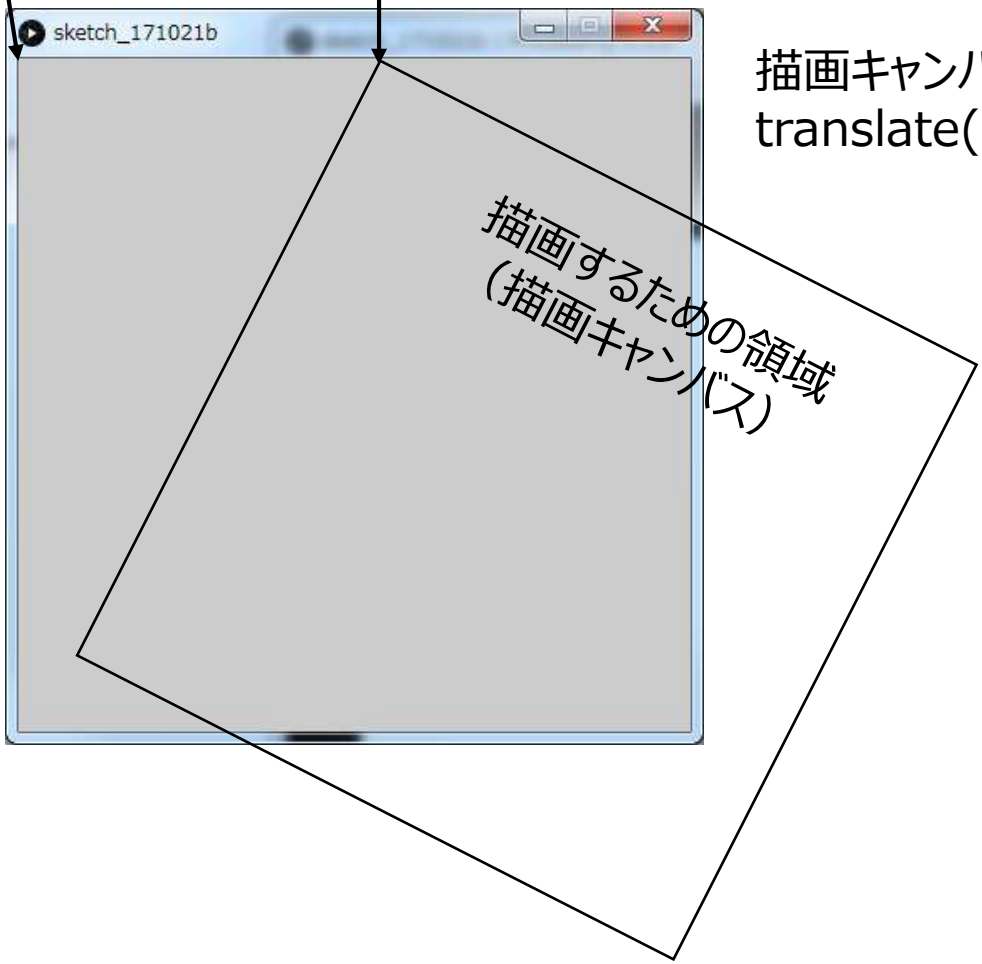
スクリーン座標系とローカル座標系

- スクリーン座標系
  - 左上を原点(0,0)とした座標系
- ローカル座標系
  - 座標変換によって移動・回転・変形した座標系

- スライドや資料での呼び方
  - スクリーン座標系　⇒　実行画面
  - ローカル座標系　⇒　描画キャンバス
  - イメージしやすそうな呼び方で説明する

スクリーン座標系の原点
(通常の(0,0)の位置)

ローカル座標系の原点
(移動した(0,0)の位置)

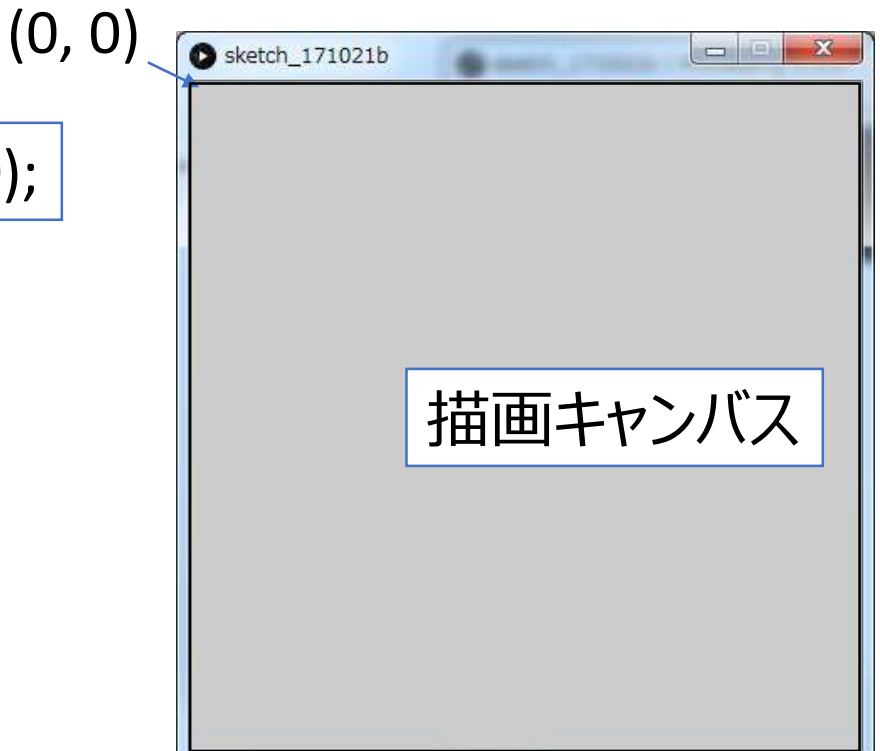


描画キャンバスは
translate()やrotate()で移動できる

座標変換：移動（translate)

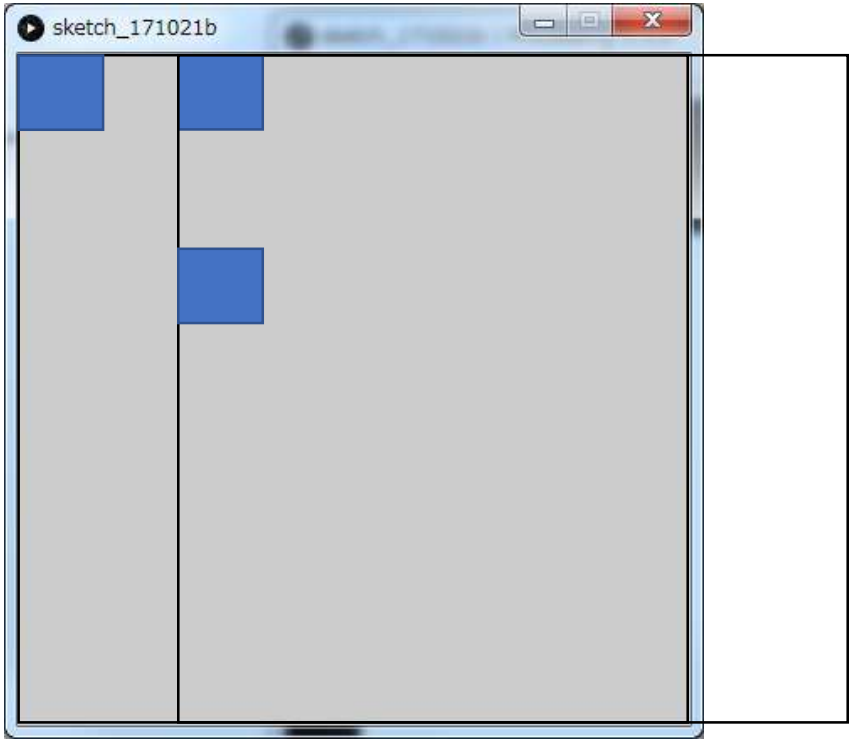
- translate(x, y); の命令を実行すると、
  描画キャンバスを動かすことが出来る
  ＝　座標の原点が移動する

translate(100,0);



座標変換後の図形描画

```
rect(0,0,10,10);
translate(100,0);
rect(0,0,10,10);
translate(0,100);
rect(0,0,10,10);
```

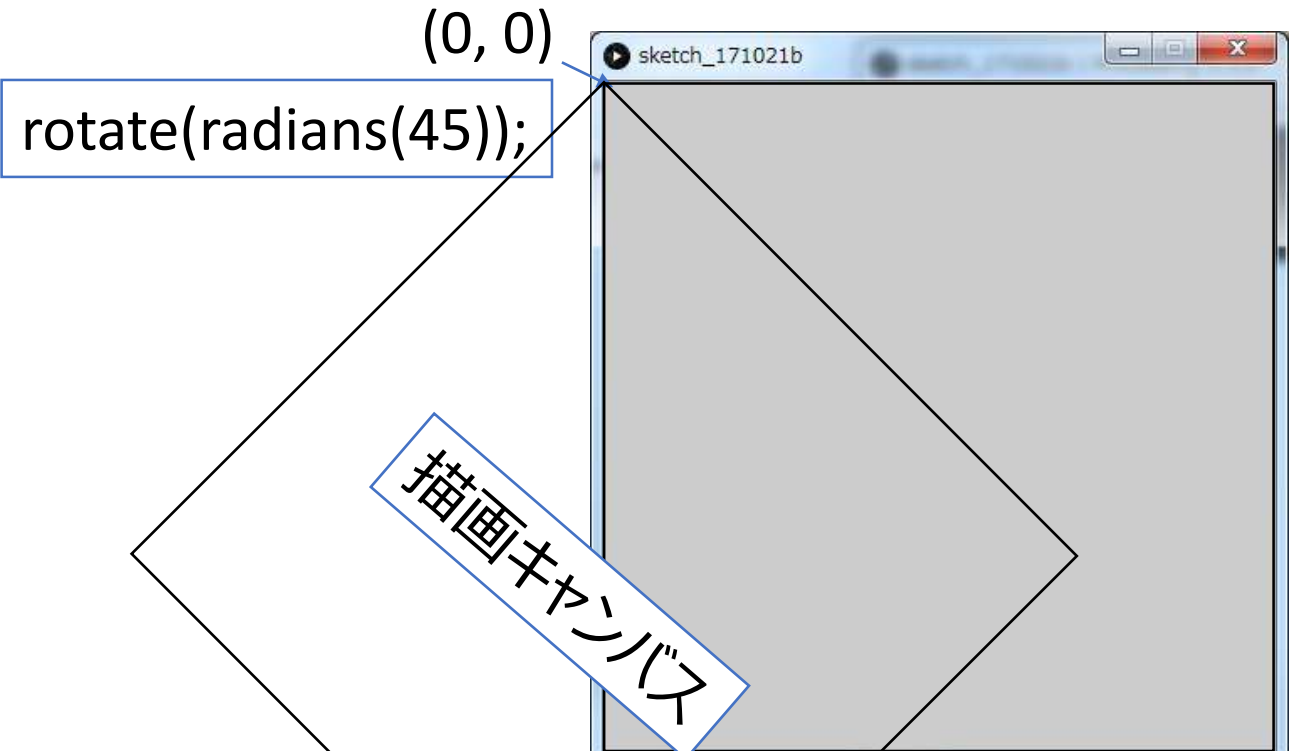


座標変換後にもう一度座標変換すると、その場所から更に移動する

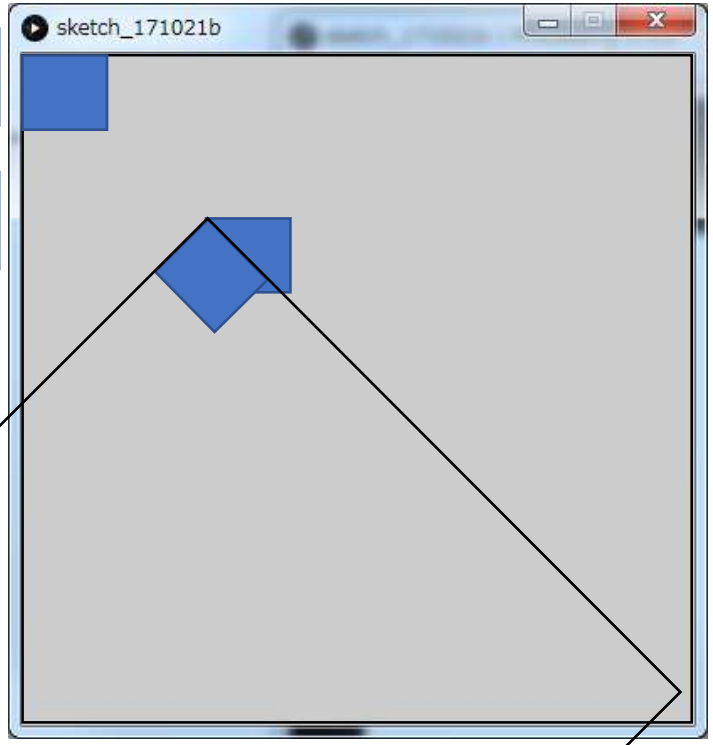
同じ座標でも、座標変換後は移動した描画キャンバス上の座標で描画される

座標変換：回転（rotate）

- rotate(angle); の命令を実行すると、描画キャンバスを回転させることが出来る
  - その時点の原点を中心に回転する



```
rect(0,0,10,10);
translate(100,100);
rect(0,0,10,10);
rotate(radians(45));
rect(0,0,10,10);
translate(100,0);
```



座標を回転させた後に座標を移動すると、回転後の座標に対して移動する

座標を記録する、取り出す

```
translate(100,0);
pushMatrix();
translate(0,100);
popMatrix();
```

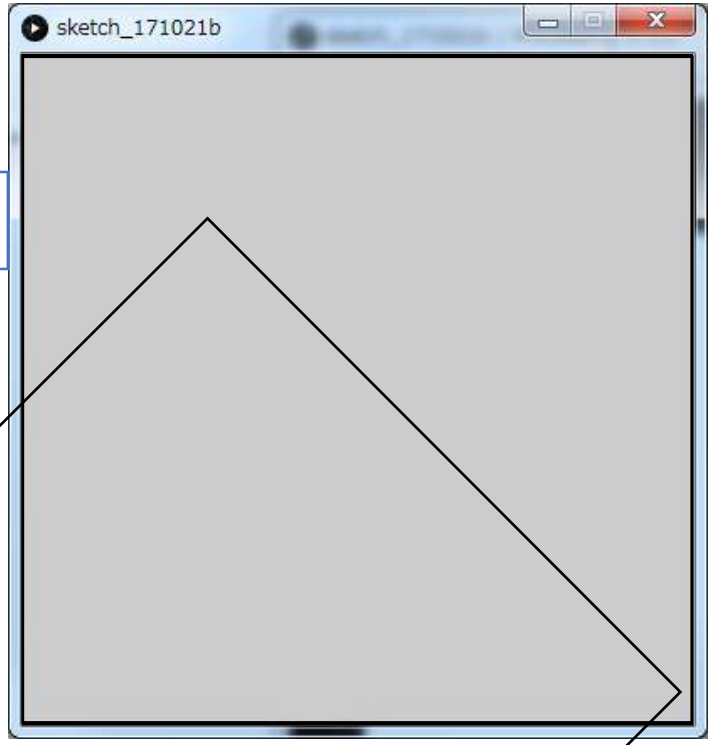
座標をリセットする：resetMatrix()

translate(100,100);

rotate(45);

translate(100,0);

resetMatrix();



resetMatrix(); を実行すると、移動回転させた座標が元に戻る

drawが1回終わる度に座標はリセットされる

座標系の保存と呼び出し

- 何度も座標系を移動・回転していると、
  現時点の座標系を把握するのは困難
  - 訳が分からなくなる
- 座標系を動かす前に座標系を保存して、
  動かし終わったら元に戻すようにしておくと良い
- translate()やrotate()を
  pushMatrix(); //座標系を保存
  popMatrix(); //保存した座標系を呼び出す
  で囲む
  - 座標系を動かすときのテンプレと思って覚えよう

pushMatrix(); popMatrix();

- まずは、このパターンを覚えれば図形の回転
  - 理解が進めば、より便利な使い方も出来るようになる

```
pushMatrix();  //座標系を保存

translate(100, 200);
rotate(radians(30));
・・・; //座標変換の命令

popMatrix();  //保存した座標系を呼び出す
```

囲む

pushMatrix()とpopMatrix()の数を合わせること
pushMatrix()の数が少ないとエラーになる

座標変換のメリット、デメリット

- メリット
  - **手軽に図形を回転出来る**
    - 座標変換を使わない場合は、三角関数で回転
    - 繰り返しと組み合わせると、
      簡単に円形模様を作れる
    - どこに配置するか？をあまり考えなくても良くなる
- デメリット
  - 座標の移動、回転を複数回行っていくと、
    その時点の座標がどこなのか、直感的に分かりにくい

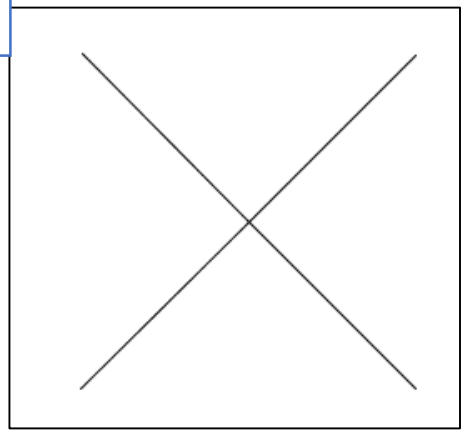
円形模様の作成

座標変換で模様を描く

- 例）線を引いて90度回転、を4回繰り返す

```
size(400,400);
background(255);
translate(200,200); //描画キャンバスの中心に移動
line(0, 0, 100, 100);  //(0, 0) から線を引く
rotate(radians(90)); //90度回転
line(0, 0, 100, 100);  //(0, 0) から線を引く
rotate(radians(90)); //90度回転
line(0, 0, 100, 100);  //(0, 0) から線を引く
rotate(radians(90)); //90度回転
line(0, 0, 100, 100);  //(0, 0) から線を引く
rotate(radians(90)); //90度回転
```

同じ処理

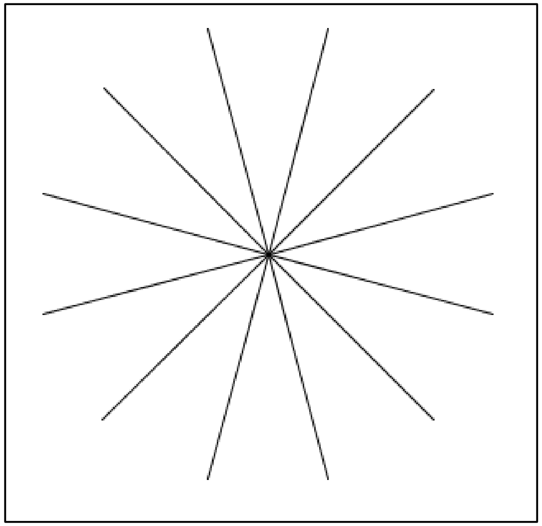


座標変換で模様を描く

- 例）線を引いて30度回転、を12回繰り返す

```
size(400, 400);
background(255);
translate(200, 200); //描画キャンバスの中心に移動
line(0, 0, 100, 100);  //(0, 0) から線を引く
rotate(radians(30)); //30度回転
line(0, 0, 100, 100);  //(0, 0) から線を引く
rotate(radians(30)); //30度回転
line(0, 0, 100, 100);  //(0, 0) から線を引く
rotate(radians(30)); //30度回転
line(0, 0, 100, 100);  //(0, 0) から線を引く
rotate(radians(30)); //30度回転
line(0, 0, 100, 100);  //(0, 0) から線を引く
rotate(radians(30)); //30度回転
line(0, 0, 100, 100);  //(0, 0) から線を引く
rotate(radians(30)); //30度回転
line(0, 0, 100, 100);  //(0, 0) から線を引く
rotate(radians(30)); //30度回転
line(0, 0, 100, 100);  //(0, 0) から線を引く
rotate(radians(30)); //30度回転
line(0, 0, 100, 100);  //(0, 0) から線を引く
rotate(radians(30)); //30度回転
line(0, 0, 100, 100);  //(0, 0) から線を引く
rotate(radians(30)); //30度回転
line(0, 0, 100, 100);  //(0, 0) から線を引く
rotate(radians(30)); //30度回転
line(0, 0, 100, 100);  //(0, 0) から線を引く
rotate(radians(30)); //30度回転
```

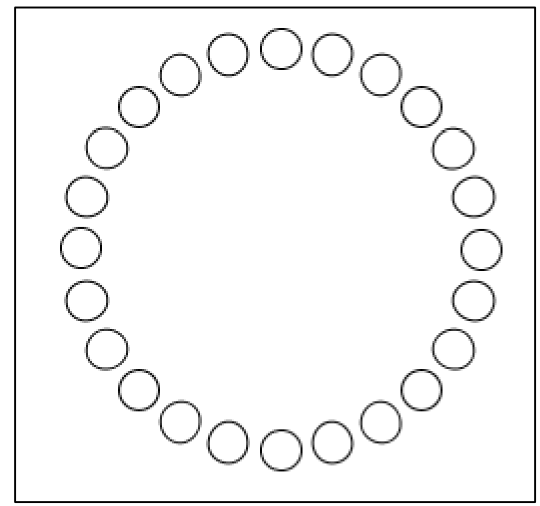
同じ処理



座標変換で模様を描く

- 例）丸描いて15度回転、を24回繰り返す

```
size(400, 400);
background(255);
translate(200, 200); //描画キャンバスの中心に移動
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
ellipse(100, 0, 20, 20);  //(0, 0) から線を引く
rotate(radians(15)); //15度回転
```



3つの円形模様
書いている命令はほぼ同じ

- 違い：「図形を描いて、座標系を回転」を何度**繰り返す**か

```
line(0, 0, 100, 100);  //(0, 0) から線を引く
rotate(radians(90)); //90度回転
```

- 同じようなことを何度も書く
  - 繰り返しを使えば簡単！
  - しかも、簡単に変更出来る！

決まった回数処理を繰り返す

```
for(int i = 0; i < 10; i++) {
  実行したい処理;
}
```

- 変数ｉが1ずつ増えて、0~9まで繰り返す
  - つまり、{}内の処理を10回実行する
  - さっきの円形模様に当てはめると・・・

座標変換と繰り返し

- 放射状に線を描いてみる
  - 36度ずつ回転させながら10本線を描く

```
size(400,400);
background(255);
translate(200,200); //描画キャンバスの中心に移動
for(int i = 0; i < 10; i++){
  rotate(radians(36));  //36度ずつ回転(360度 / 10)
  line(0,0,0,100);
}
```

座標変換と繰り返し

- 放射状に線を描いてみる
  - 10度ずつ回転させながら36本線を描く

```
size(400,400);
background(255);
translate(200,200); //描画キャンバスの中心に移動
for(int i = 0; i < 36; i++){
  rotate(radians(10));  //10度ずつ回転(360度 / 36)
  line(0,0,0,100);
}
```

繰り返し回数と回転角度

- 10度ずつ回転させながら36本

```
for(int i = 0; i < 10; i++){
   rotate(radians(36));  //36度ずつ回転(360度 / 10)
   line(0,0,0,100);
 }
```

- 5度ずつ回転させながら72本

```
for(int i = 0; i < 72; i++){
   rotate(radians(5));  //5度ずつ回転(360度 / 72)
   line(0,0,0,100);
 }
```

回転角度×繰り返し回数＝360
になるようにすると、きっちり円形になる

円形模様メソッドを作成

- 前回のキャラクター作成の繰り返し版

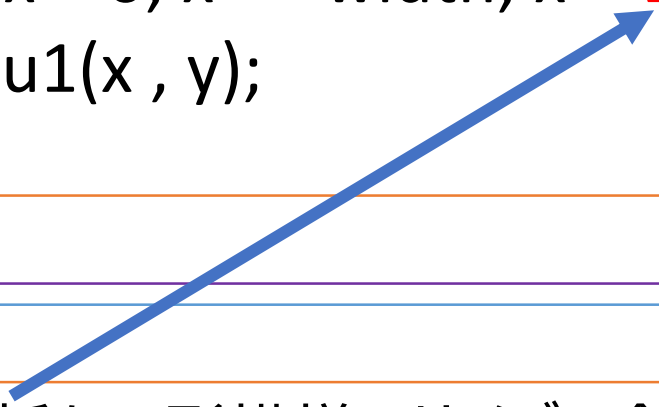
```
void setup() {
  size(400, 400);
  moyou1(50,50);
}
void moyou1(float x, float y) {
  pushMatrix();
  translate(x, y); //(x, y)に描画キャンバスを移動
  for (int i = 0; i < 36; i++) {
    rotate(radians(10));  //10度ずつ回転(360度 / 36)
    line(0, 0, 0, 50);
  }
  popMatrix();
}
```

ここをアレンジ

円形模様を繰り返しで配置

- 丸を縦横に並べるプログラムを利用

```
for(int y = 0; y <= height; y+=100) {
    for(int x = 0; x <= width; x+=100) {
      moyou1(x , y);
    }
}
```



繰り返しの更新を円形模様のサイズに合わせて変更

色々と変えて試してみよう

多重ループ（for文の中にfor文）

- 横1列に丸を描く

```
for(int x = 0; x <= width; x+=20) {
  ellipse(x , 10, 20,20);
}
```

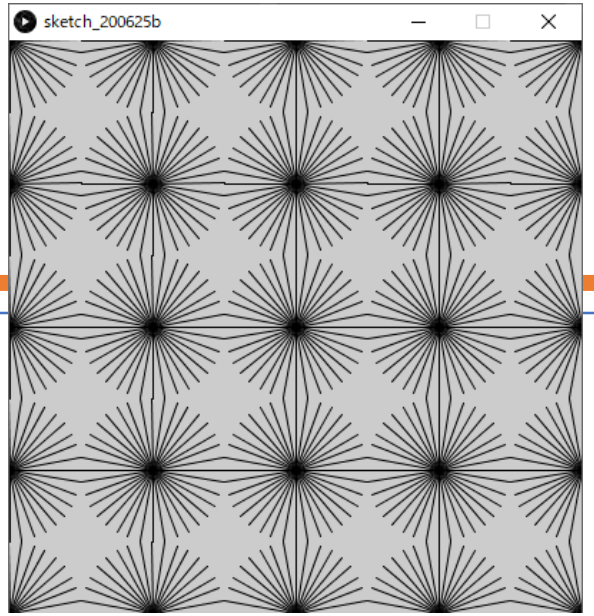
この繰り返し文を繰り返す

```
for(int y = 0; y <= height; y+=20) {
    for(int x = 0; x <= width; x+=20) {
      ellipse(x , y , 20 , 20);
    }
}
```

y軸方向の繰り返し

x軸方向の繰り返し

今回のひな形




```
void setup() {
  size(400, 400);
  for (int y = 0; y <= height; y+=100) {
    for (int x = 0; x <= width; x+=100) {
      moyou1(x, y);
    }
  }
}
void moyou1(float x, float y) {
  pushMatrix();
  translate(x, y); //書きたい場所に描画キャンバスを移動
  for (int i = 0; i < 36; i++) {
    rotate(radians(10));  //10度ずつ回転(360度 / 36)
    line(0, 0, 0, 50);
  }
  popMatrix();
}
```

自分の作った模様の名前に変える

if文を1つ以上入れること
（アレンジ例を参考に）

独自の円形模様を作成

今回のレポート

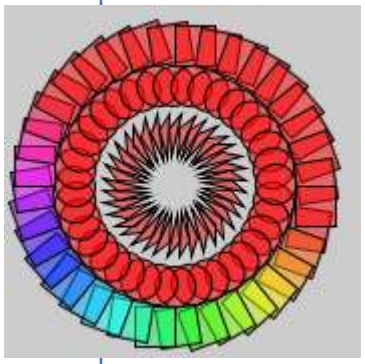
- オリジナルの円形模様を作り、繰り返しで並べる
  - オリジナルの円形模様作成
    - 色を付ける、図形を追加、ランダムを使ってみる
      - 作った円形模様を動かしたい場合は、
        ランダムは使わない方が良い
    - 円形模様メソッドを複数作れば加点
  - 繰り返しで並べる（雛型は縦横の並び、このままで可）
    - 斜め、階段状等、別の並べ方をしたら加点

円形模様のアレンジ

- 色を付ける、図形を追加、繰り返し回数・回転角度を変更
  - まずは図形を線から変更、別の図形を追加しよう
- 条件分岐の利用
  - if(i < 10) :もしも、iが10より小さければ
  - if(i % 2 == 0): iを2で割った余りが0ならば
    - 偶数なら、奇数ならの条件になる(アレンジ例参照)
  - if(x >= 100) : もしも、xが100以上なら
    - 描画位置(横の座標)によって書く図形を変えるなど
- 注意点
  - 繰り返し回数を多くしすぎないように

アレンジ例

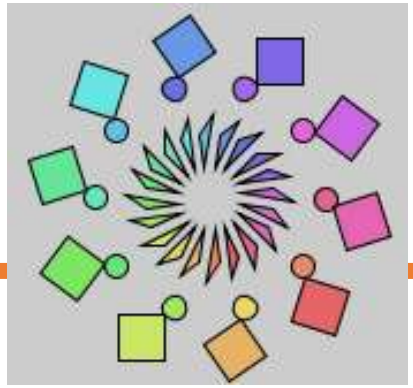
```
void moyou1(float x, float y) {
  pushMatrix();
  translate(x, y); //書きたい場所に描画キャンバスを移動
  for (int i = 0; i < 108; i++) {
    rotate(radians(10));  //10度ずつ回転(360度 / 36)
    fill(i*18, 100,100,50);  //1周で360. 色相環が1周する.
    if (i < 36) {  //iが36未満なら(1周目)
      rect(60, 0, 20, 20);
    } else if(i<72){  //そうでなく、iが72未満なら(2周目)
ellipse(50, 0, 20, 20);
    } else {  //そうでなければ(3周目)
      triangle(10,10,20,20,20,30);
    }
}
  popMatrix();}
```



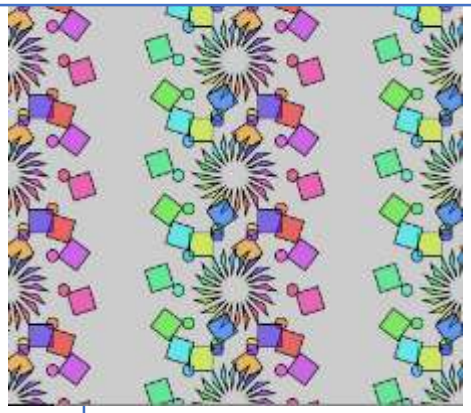
3回転させて、1週ごとに図形を変える

これは、繰り返し回数が多いので動かす図形には向かない

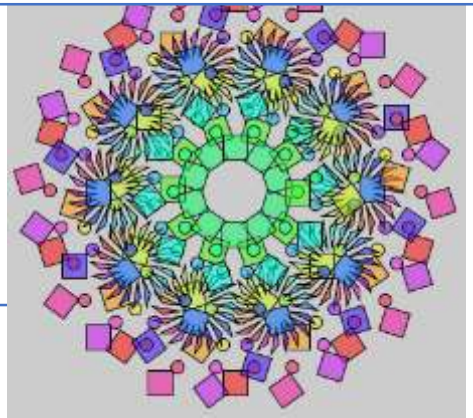
アレンジ例



縦と横の繰り返し回数を変える



円形模様を使って円形模様を作る
（工夫が必要）



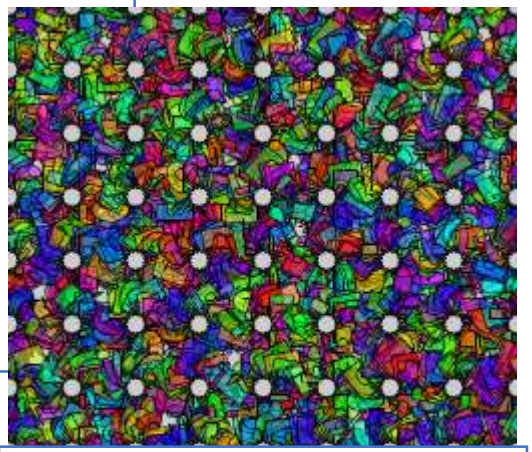
```
void moyou1(float x, float y) {
  pushMatrix();
  translate(x, y); //書きたい場所に描画キャンバスを移動
  for (int i = 0; i < 20; i++) {
    rotate(radians(18));  //18度ずつ回転(360度 / 20)
    fill(i*18, 100,100,50);  //1周で360. 色相環が1周する.
    if (i%2 == 0) {  //iを2で割った余りが0なら ( iが偶数なら)
      rect(50, 20, 20, 20);
    } else {  //そうでなければ ( i が奇数なら)
      ellipse(50, 0, 10, 10);
    }
    triangle(10,10,20,20,20,30);
  }
  popMatrix();
}
```

アレンジ例(ランダム利用)

- 注意：後の回で作った図形を動かしたいなら、
  ランダムはあまり使わない方が良いかも

```
void moyouRandom(float x, float y) {
  pushMatrix();
  translate(x, y); //書きたい場所に描画キャンバスを移動
  for (int i = 0; i < 36; i++) {
    rotate(radians(10));  //10度ずつ回転(360度 / 36)
    fill(random(360), 100,100,50);
    rect(10,20,random(20),random(30));
    ellipse(50,20,random(20),random(30));
  }
  popMatrix();
}
```

すべてランダムは、
ただ乱雑な図形を描くだけなのでNG



繰り返しの更新を
小さくして敷き詰めてみる

レポートチェックリスト（第9回）

- ミニテストを受験した(レポートの前と後)
- 雛型のプログラムをアレンジした
  - オリジナルの円形模様メソッドを作成した
    - 図形を追加した
    - 繰り返しの中で条件分岐を使っている
  - 円形模様メソッドを繰り返しで並べて描画した

- PowerPointでレポートを作成した
  - タイトル、作品紹介(工夫点)、プログラムの3枚
- Moodleでレポートを提出した

加点項目

- 複数の円形模様メソッドを作成した
  - 1つの画面内に2つ以上の円形模様を繰り返しで表示
- 円形模様の並びを、雛型以外の方法で並べた
  - 斜めに配置、円状に配置、交互に配置、階段状など

- 繰り返し分を複数書き、1画面の中で複数の並べ方をした
  - forが沢山ある
- 前回のプログラムの敵キャラに円形模様を使った
  - これは別のプログラムとして提出