プログラミング　　2 回目　　担当：松田　侑樹

年　　組　氏名　（　　）

この時間のねらい：

1. ウィンドウの開き方、基本的な図形の作成方法を理解しよう。
2. いろいろな形の図形を作成する方法を学ぼう。

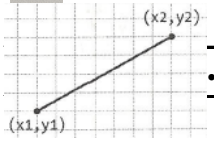
今回から、いろいろな関数を学んでいきましょう。「Processing」の強みは、図形や文字などを描画させることです。描画のためには、（　　）が必要です。⟶ `size(480, 480);`

右上のように（　　）の形で記述されるものを（　　）と言います。

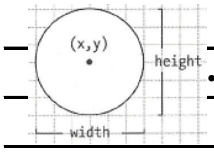
関数を組み合わせて、作成していくのが（　　）になります。

それでは、ウィンドウが表示されたら、図形を描いてみましょう。

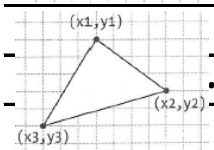
・・・（　　）。→ `point(x1, y1);`



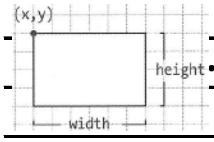
・・・（　　）。→ `line(x1, y1, x2, y2);`



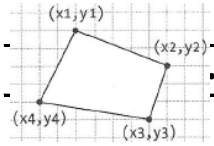
・・・（　　）。→ `ellipse(x, y, width, height);`



・・・（　　）。→ `triangle(x1, y1, x2, y2, x3, y3);`



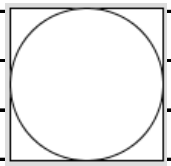
・・・（　　）。→ `rect(x, y, width, height);`



・・・（　　）。→ `quad(x1, y1, x2, y2, x3, y3, x4, y4);`

以上の関数を使用してみましょう。

2つの図形を重ねてみるとあることが分かります。右のようになります。



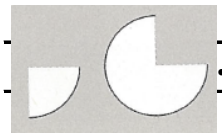
このとき、四角と丸のどちらが先に置かれたかといえば、（　　）ですよね。

（自分でそれぞれの図形を置くことを考えてみれば当然のはずです。）

このように上から順番に処理を行うことを（　　）と言います。

開発を行うとこれが（　　）の原因になることがあります。めちゃくちゃダルいので気を付けましょう。

また、円の一部だけを表示する、すなわち、（　　）を描画する関数は下のようになります。



・・・（　　）。→ `arc(x, y, width, height, 始まり, 終わり);`

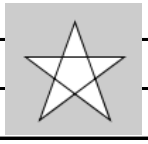
以上が基本的な図形の書き方になります。裏面で応用と装飾に関してまとめます。

メモ欄

表面で、基本図形の作成方法を学びました。もっと違う形を描きたい場合について学びます。

メモ欄

練習として☆マークを描くことを考えましょう。右のようにできれば上々です。



```
beginShape();
vertex(x1, y1);
vertex(x2, y2);
vertex(x3, y3);
vertex(x4, y4);
vertex(x5, y5);
endShape();
```

```
beginShape();
vertex(x1, y1);
vertex(x2, y2);
vertex(x3, y3);
vertex(x4, y4);
vertex(x5, y5);
endShape(CLOSE);
```

左の関数を利用して作成します。

1つ1つの点をプロットするように書いていきます。

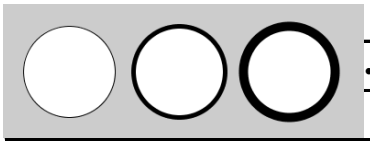
2つのプログラムで違うのは何でしょうか？プログラム的にはendShape(); が違いますね。

それぞれ実行してみると違うのは（　　）ということです。

これを使えば、（直線ではありますが、）基本図形以外の図形を作成できます。いろいろと用途を考えてみましょう。

ここから先は、図形の装飾に関してです。

想像しやすいのは（　　）の太さが上がると思います。下のように変更ができます。



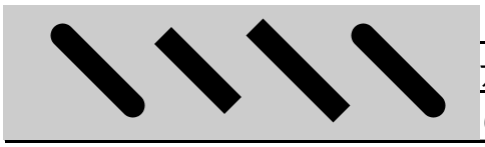
・・・ strokeWeight(太さ：ピクセル数で指定);

今回の引数は長くありますが、数字を書くだけです。また、ピクセルというのは（　　）のことで、皆さんの使用するパソコンの画面にある小さい四角が何個分かということを表します。

個人的には（　　）程が良いかと思います。（場合によりますが・・・）

ちなみに画像のものは左から（　　）、（　　）、（　　）、となっています。

続いては線の両端の状態を変化させる方法です。

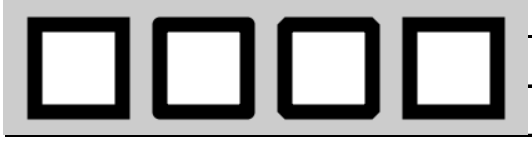


画像の右3種類の形状を指定できます。それぞれの指定方法は左から順に下のようになります。

（一番左は指定なしです。）

```
strokeCap(SQUARE);  strokeCap(PROJECT);  strokeCap(ROUND);
```

最後に角の形状を変更する、関数を紹介して終わりにします。



画像の右3種類の形状を指定できます。それぞれの指定方法は左から順に下のようになります。

（先述と同じように一番左は指定なしです。）

```
strokeJoin(ROUND);  strokeJoin(BEVEL);  strokeJoin(MITER);
```

ちなみに、以上の関数を使用するとそれ以降全て適応となるので注意してください。