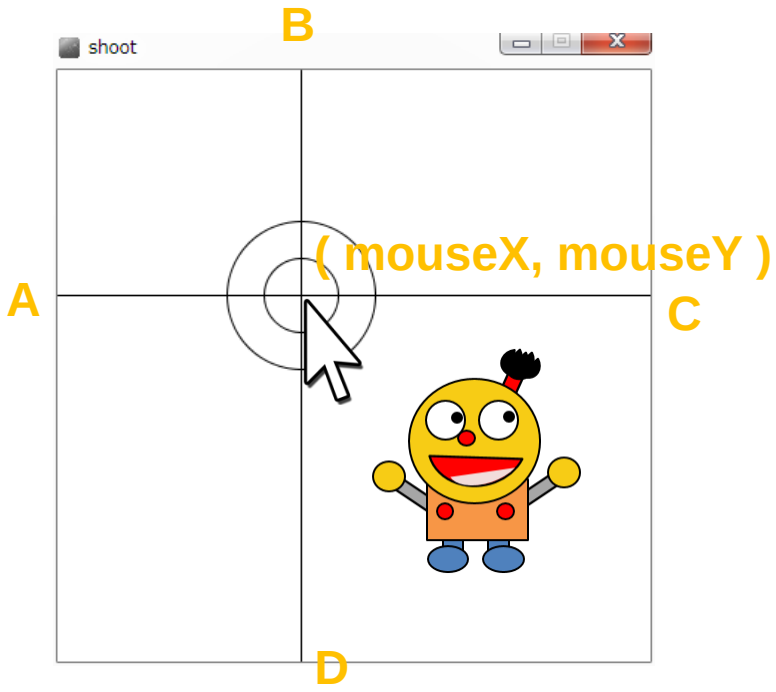
プログラミング演習I（第3回）課題

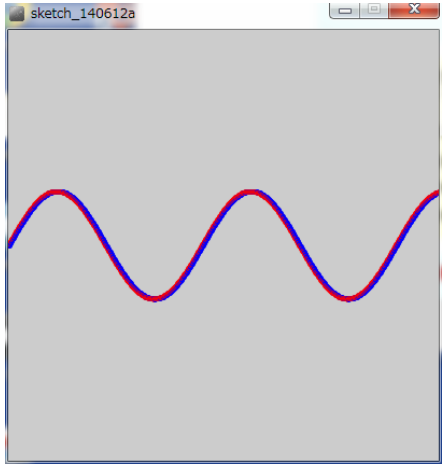
- **基本課題①　スケッチ名：shoot**
  - 白背景の場所に適当なターゲットを描画し（丸でもキャラクタでも良い）、カーソルの上に黒色の十字線と丸（半径25と50）を描画しなさい。ただし、線の両端はウインドウの端になるようにしなさい。十字線と丸はターゲットの手前に描画せよ（ヒント：マウスの座標は(mouseX, mouseY)だが、上端下端、左端右端の座標は？）



座標A (　　,　　)
座標B (　　,　　)
座標C (　　,　　)
座標D (　　,　　)

AからCの線
line(　　,　　,　　,　　);
BからDの線
line(　　,　　,　　,　　);

プログラミング演習I（第3回）課題



- **基本課題②　エラー修正：error**

$$\sin\alpha\cos\beta = \frac{1}{2}\{\sin(\alpha+\beta)+\sin(\alpha-\beta)\}$$

$$\sin\theta\cos\theta = \frac{1}{2}\sin 2\theta$$

  - 三角関数の加法定理を用いると，上記のような式を求めることができる．この式が本当に正しいかどうかを確認したくなったため，右上図のように2つのグラフ（一方を青線，他方を赤線で描画）を作って比較を行おうとしたが，図のような結果を得ることができない．原因を調べてしっかり動作するようにせよ．なお，ぴったり重なっていると差を確認できないため，xの正方向に1ピクセルだけ動かしている．

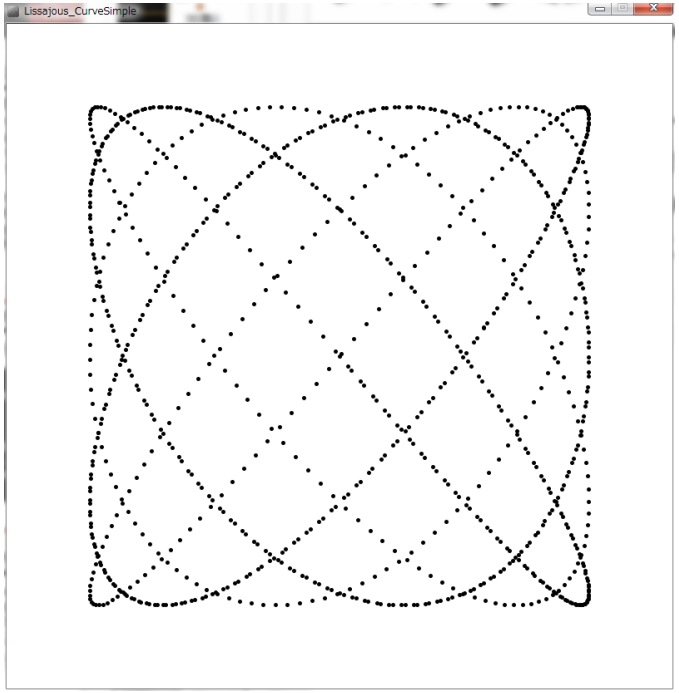
$$y = \sin\theta\cos\theta$$

$$y = \frac{1}{2}\sin 2\theta$$

```
void setup(){
    size( 400, 400 );
    strokeWeight(5);
}
int x;
void draw(){
    int x = 0;
    stroke( 255, 0, 0 );
    float  y1 = 100*sain(radians(x))*cos(rad ians(x));
    stroke( 0, 0, 255 );
    float  y2 = 100%2*sain(radians(2x));
    point( x, y );
    point( x+1, y );
    x++;
}
```

プログラミング演習I（第3回）課題

- **基本課題③　リサージュカーブ：curve**
  - x と y の座標が t によって変化する下記の数式の計算結果の座標をプロットせよ．ただし，t は draw() 毎に0.01ずつ増加するようにせよ．またウインドウサイズは800x800とせよ．
    - $x = 300\sin(at) + 400$
    - $y = 300\sin(bt) + 400$
    - a = 5，b = 6 の時が右図



プログラミング演習I（第3回）課題

- **基本課題④　関数の描画：plot**
  - x と y の座標が t によって変化する下記の数式の計算結果の座標(x1, y1), (x2, y2)をpointを利用してプロットせよ．ただし，t は draw() 毎に0.01ずつ増加するようにせよ．またウインドウサイズは800x800とせよ．
    - $x1 = \frac{(-t^2+40t+1200)\sin(radians(t))}{2} + 400$
    - $y1 = -\frac{(-t^2+40t+1200)\cos(radians(t))}{2} + 800$
    - $x2 = -\frac{(-t^2+40t+1200)\sin(radians(t))}{2} + 400$
    - $y2 = -\frac{(-t^2+40t+1200)\cos(radians(t))}{2} + 800$

```
float t = 0.00;
void draw(){
    x1,x2,y1,y2を定義
    x1,x2,y1,y2の計算
    point( x1, y1 );
    point( x2, y2 );
    t = t + 0.01;
}
```

プログラミング演習I（第3回）課題

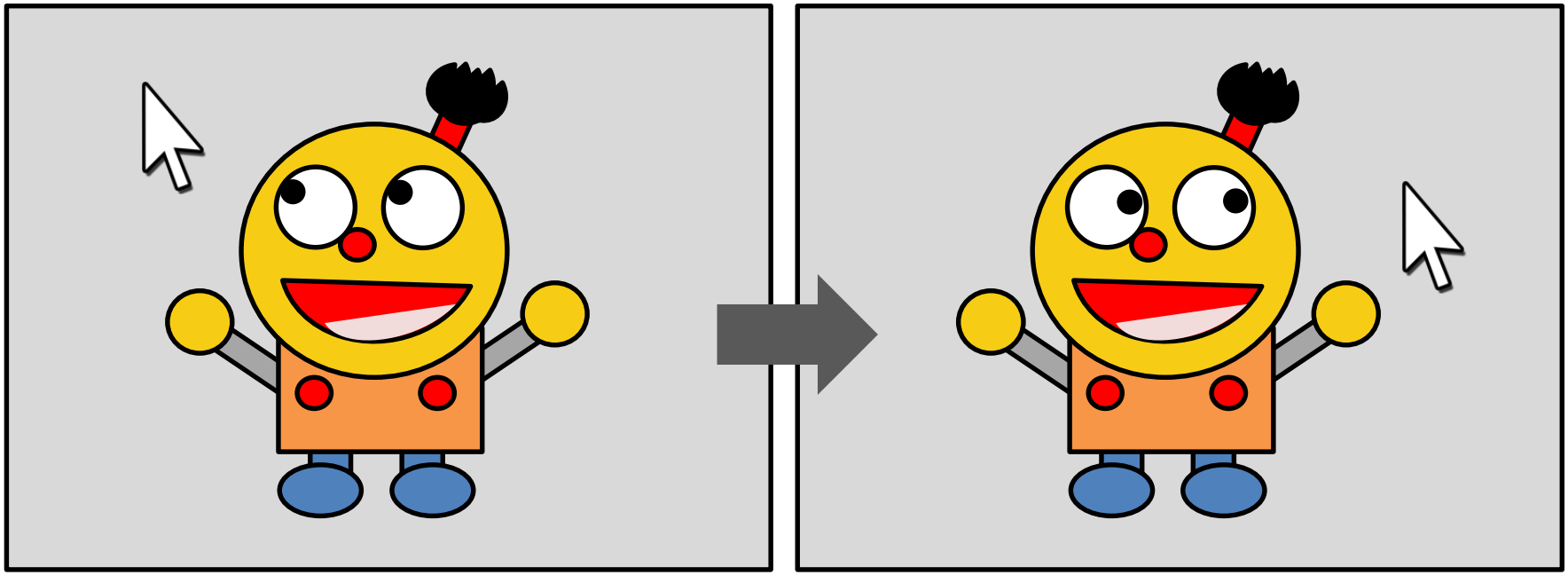
- コンピュータが読めるコードにするとどうなる？

```
x1 =
y1 =
x2 =
y2 =

point( x1, y1 );
point( x2, y2 );
```

プログラミング演習I（第3回）課題

- **<u>発展課題①　スケッチ名：eye</u>**
  - 前回作成したキャラクタを描くプログラムを改造して、キャラクタの黒目がマウスカーソルを追うプログラムを作成せよ。



※目玉が楕円形の場合、黒目の動きは楕円軌道にこだわらなくてもよいです。

プログラミング演習I（第3回）課題

- キャラクタの描画を draw() の中に入れてしまう
- 目玉の中心は（　　，　　）で，黒目を円周上を動かすと考えた時，その半径（または直径）は［　　］である
- ある黒目の中心を (eye1X, eye1Y) とした時に，その目はどう表現できるかを考える
  - eye1X, eye1Y を使って黒目を描画しよう
  - eye1X, eye1Y には適当な値を代入しておこう（初期値等）
- （mouseX, mouseY）と目の中心（　　，　　）のなす角度 theta を計算しよう（次頁のatan2を参照）

  theta = atan2(　　);
- theta を利用して eye1X と eye1Y の座標を計算して求めよう

  eye1X = 　　;

  eye1Y = 　　;
- eye1X, eye1Y を利用して黒目を描画しよう（他の黒目も同様に）

プログラミング演習I（第3回）課題

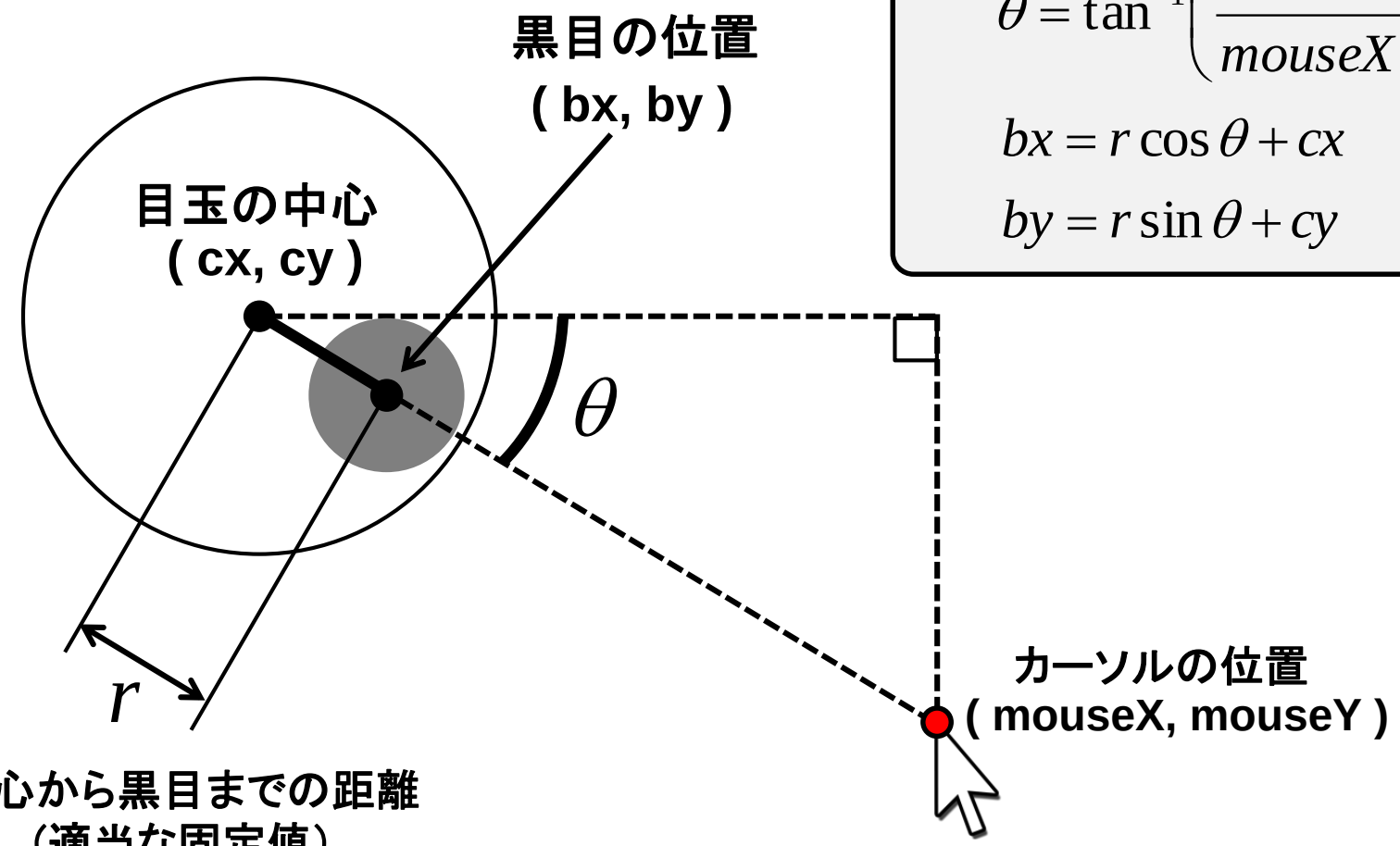
- **基本課題① のヒント**

黒目の位置を求めるための数式

$$\theta = \tan^{-1}\left(\frac{mouseY - cy}{mouseX - cx}\right)$$

$$bx = r\cos\theta + cx$$

$$by = r\sin\theta + cy$$

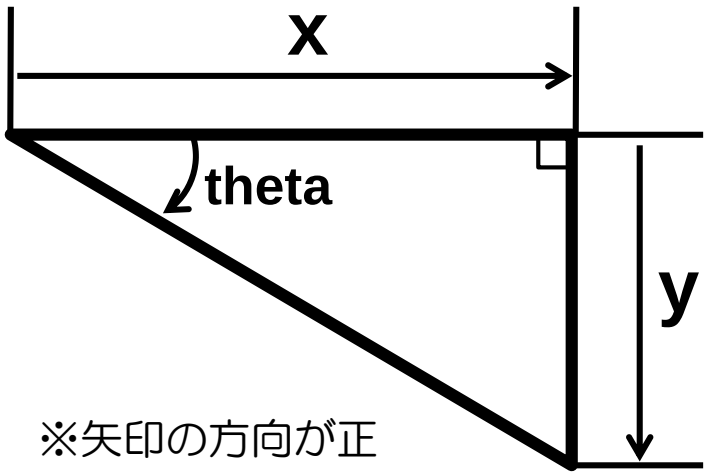


プログラミング演習I（第3回）課題

- **$\tan^{-1}$ の求め方は2つある**

アークタンジェントの計算には
atan() と atan2() があり、それぞれ
値域が異なります。なお、いずれも
計算結果は実数値(float)です。

今回の課題ではatan2() を使うと
よいでしょう（atanの場合は問題が
発生するが理由はわかるかな？）



※矢印の方向が正

theta = atan(y/x);　　値域は　$-\frac{\pi}{2} \le theta \le \frac{\pi}{2}$

theta = atan2(y, x);　　値域は　$-\pi \le theta \le \pi$

今日の上級テクニック

- size()で設定したウィンドウのサイズ情報をプログラム中で使いたいときは、width と height を使おう。

たとえば、画面の中央に円を描くプログラムはこんな感じですが、

```
size( 400, 300 );
ellipse( 200, 150, 50, 50 );
```

これを width と height を使って書くと…

```
size( 400, 300 );
ellipse( width/2, height/2, 50, 50 );
```

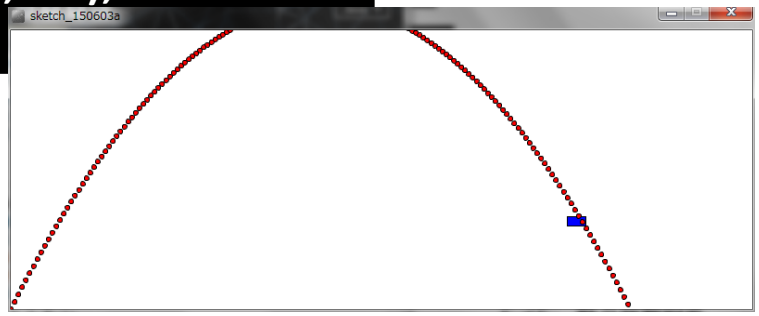
こっちのほうがわかりやすいし、ウィンドウのサイズが変わっても修正箇所が少なくてよい！

- widthとheightはmouseXやmouseYと同じく、Processingがあらかじめ定義している変数です。定義済みの変数は文字がピンク色になります。

プログラミング演習I（第3回）課題

- **発展課題②　スケッチ名：throw**
  - 地上で斜め上方向に赤色のボールを投げたときの様子をシミュレーションするプログラムを作成してください。
  - また、600m先にある高さ10m、横幅20mの青色のターゲットに当たるように投射速度（v_0）と角度を調整せよ。あたったかどうかの判定は目測で良い。なお下記のsetupを利用せよ

```
void setup(){
   size( 800, 300 );
   background(255);
   fill( 0, 0, 255 );
   rect( 600, height-20, 20, 10 );
}
```



ヒント：　斜方投射の式

$$x = v_0 t \cos\theta$$

$$y = v_0 t \sin\theta - \frac{1}{2} g t^2$$

v_0 は初速（投射速度）
g は重力加速度（9.8）
t は経過時間となる

※1フレームの経過時間は0.1秒とせよ。また、1ピクセル=1mと考えてください。