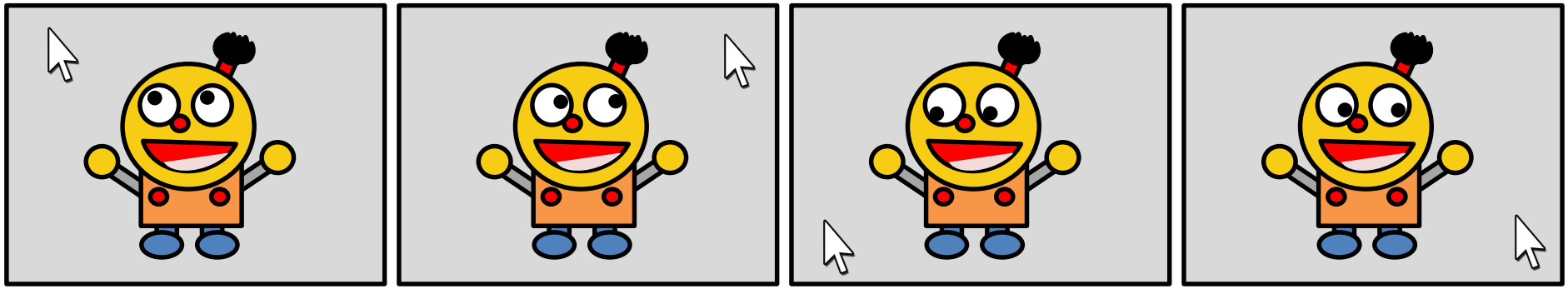
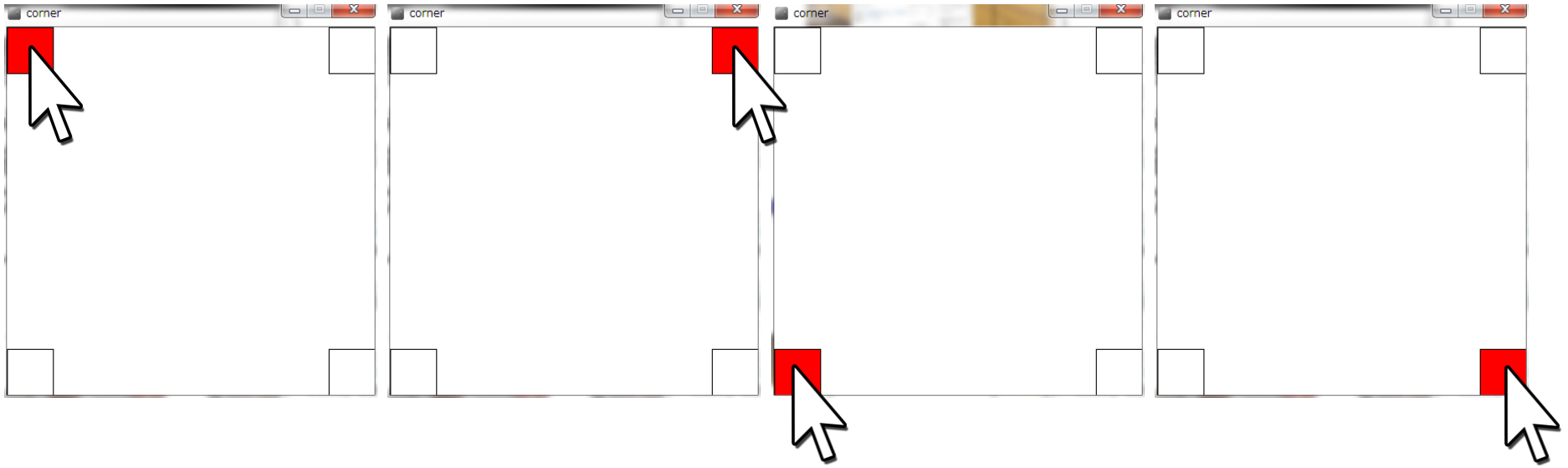
プログラミング演習I（第5回）課題

- **基本課題①　スケッチ名：eye2**
  - カーソルの位置によってキャラクタの目の向きが変わるプログラムを作ってください。
  - ただし、カーソルがキャラクタの顔に対して【左上にある時】【右上にある時】【左下にある時】【右下にある時】の４パターンで表現すること。



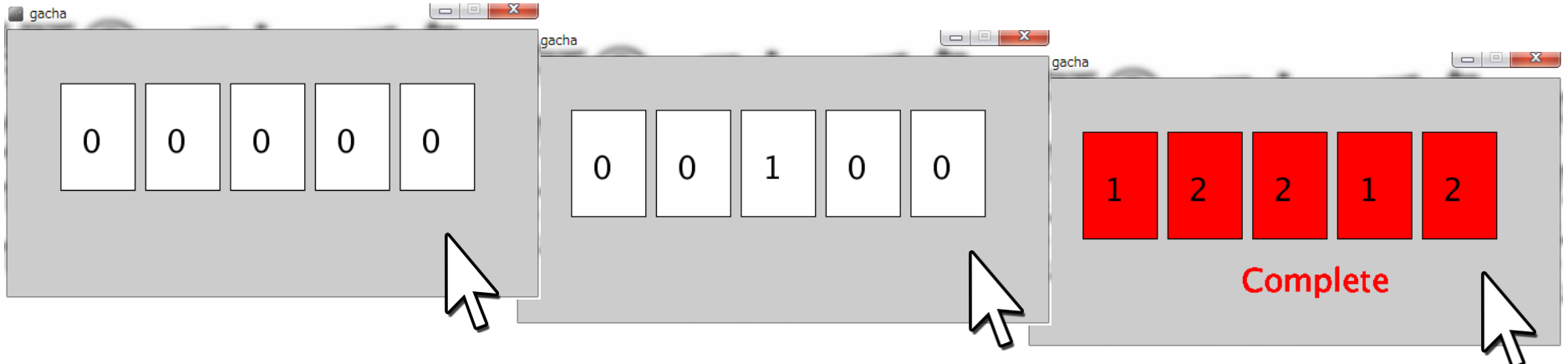
プログラミング演習I（第5回）課題

- **基本課題②　スケッチ名：corner**
  - 400x400のウインドウ（背景白色）の四隅に縦横50ピクセルの四角形を描画し，その内の1つを赤色で塗りつぶせ．また，その赤色の四角形内をクリックすると次の場所（左上→右上→左下→右下→左上の順で変更）に赤色の表示位置を切り替えよ
  - ただし，赤色の四角形外をクリックした時には，場所が変更されないようにせよ（つまり反応しないようにせよ）



プログラミング演習I（第4回）課題

- **基本課題③ スケッチ名：complete_gacha**
  - クリックする度にクリックする度に5種類のカードの1種類がランダムに選ばれ，枚数が1加算されるプログラムを作成し，それぞれのカードが選ばれた枚数を表示するプログラムを作成せよ
  - また，すべてのカードが1枚以上になったら，「Complete」と表示し，カードの色が赤色になるとともに，楽しげな音を鳴らせ



ヒント

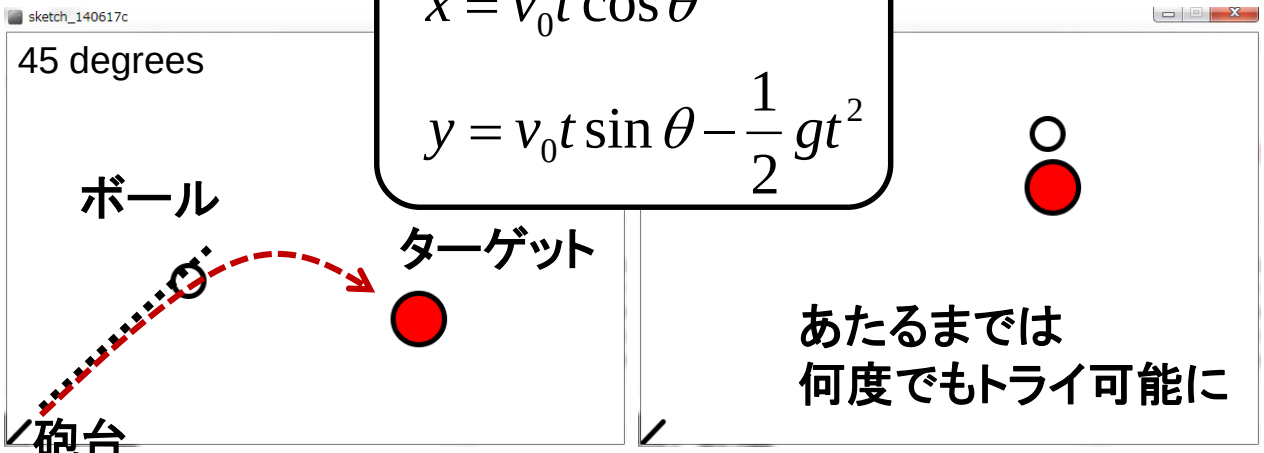
- 基本課題1
  - カーソルの場所に応じて条件分岐を行い，目の描画を切り替えるだけ！
- 基本課題2
  - どの四角形が赤色になっているかを管理する変数を用意して，その変数の値と座標で当たり判定を行い，当たっていたら変数の値を変更するだけ！
- 基本課題3
  - すべてが1枚以上というのはどういう条件なのかを整理するだけ！
  - 同じ音が連続してならないようにするにはどうするか？

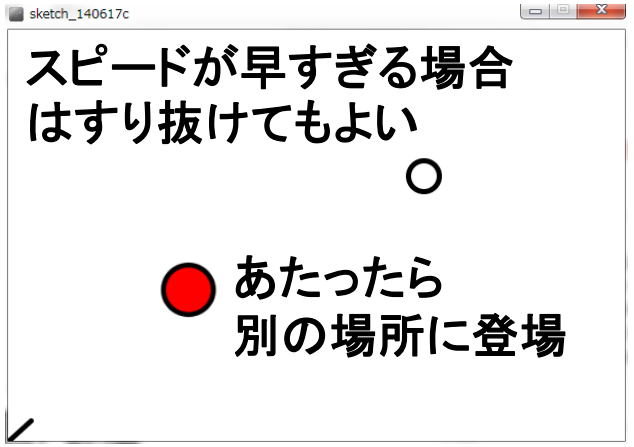
プログラミング演習I（第5回）課題

- **発展課題①　スケッチ名：launch2**
  - 600x400のウインドウ左下から砲台の向きにボール（半径15m）を発射するプログラムを作成せよ。ただし、砲台は**上下キーによって角度を1度ずつ変更**（初期角度は45度）し、**右キーで投射**されるようにすること。また、初速を100m/sとせよ。さらに、現在の角度を左上に表示せよ。
  - ランダムに指定した距離（0～600mの間）に半径25mの赤丸のターゲットを描け。赤丸は上下に1フレームあたり2m動き，上端下端で折り返すようにせよ。なお、ボールがターゲットに衝突するとターゲットを消し、次のターゲットを登場させるようにせよ。1フレームは0.1秒とする。
  - できれば当たった時に爆発音を鳴らせ。

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

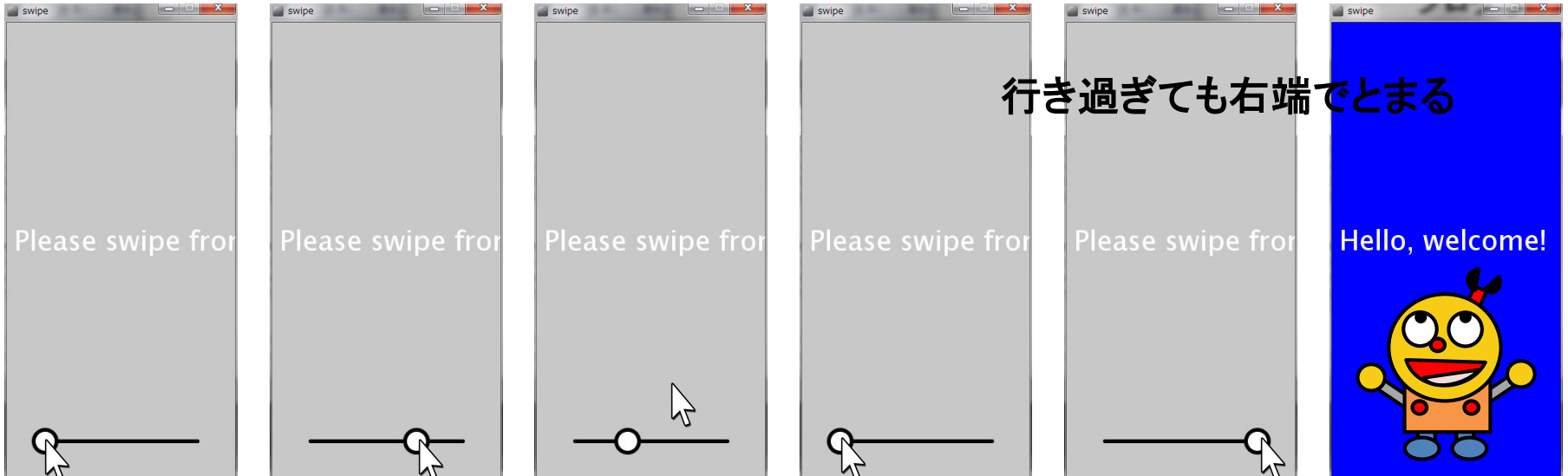
$$y = v_0 t \sin\theta - \frac{1}{2} g t^2$$





プログラミング演習I（第5回）課題

- **発展課題②　スケッチ名：unlock**
  - 縦長のウインドウの下部に左の方に丸型のものを用意し、それを右側にスライド（スワイプ）させると、ロックが解除されキャラクタが描画されている画面へと遷移せよ
  - なお、右端まで移動せずに手を離した場合は、5ピクセル/フレームの速度で最初の位置に戻るようにせよ。また、指定の位置より右や左にはみ出ないようにせよ。



今日使うテクニック

① text()で表示する文字の大きさを変える方法

- 文字の大きさを変えるには　textSize(文字サイズ) を使う。

```
void setup() {
  size(300, 150);
}

void draw() {
  fill(0);
  textSize(50); // 文字の大きさを設定
  text( "Processing", 20, 90 ); // 文字を表示
}
```

- textSize()は、fill()やstroke()と同様に何回でもパラメータを変えて指定できるので、大きさの違う文字を混在させることができる。

今日使うテクニック

② text() で表示する文字の書体(フォント)を変える方法

- フォントを変えるには、PFont、createFont()、textFont() を使う。
- **日本語を使いたいときは日本語フォントの指定が必要**
- 以下はHGS創英角ポップ体で「Processing」と書く例

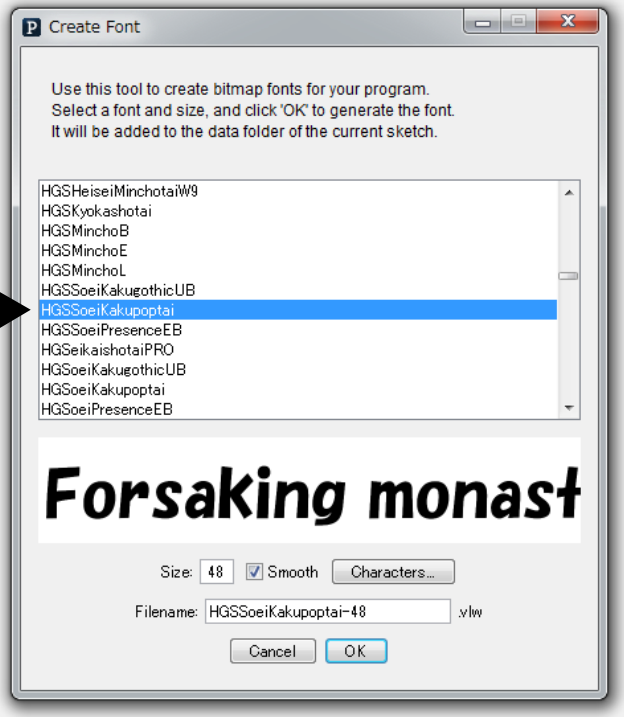
```
PFont myFont; // フォント

void setup() {
  size(300, 150);
  myFont = createFont("HGSSoeiKakupoptai",10);  // フォントを準備
  textFont(myFont);  // フォントを設定
  textSize(50);          // 文字サイズを改めて変更することもできる
}

void draw() {
  fill(0);
  text( "Processing", 20, 90 ); // 文字を表示
}
```

今日使うテクニック

- PFont はフォントを格納する変数につかうデータ型です。
  int や float などと同じような扱い。

- **createFont(フォント名, 文字サイズ)** でフォントを準備する。
  フォント名は、Processingのメニューの
  Tools –> Create Font...
  で出てくるパネルで確認できる。

**このリストにプログラム中で使える
フォント名が表示される。**



- 最後に、**textFont(フォント)** で
  フォントを設定する。

今日使うテクニック(音編)

- 音を鳴らすにはminimというものを利用する(詳しくは12回目で)
- 今日のテクニックを実現するにはおまじないが必要です
- 適当なwavやmp3ファイルをプログラムの上にドロップし下記のコードを実行

```
import  ddf.minim.*;
Minim  minim;
AudioSnippet  crash;
void setup(){
    minim = new Minim( this );
    crash = minim.loadSnippet( "crash.mp3" );
}
void stop(){
    crash.close();
    minim.stop();
    super.stop();
}
```

赤文字は毎回同じおまじない
青文字は音に応じて利用

音を再生する部分に挿入

```
crash.rewind();
crash.play();
```

今日使うテクニック(音編)

マウスクリックの時に
クラッシュ音を鳴らす

crash.mp3 は事前に
ダウンロードし、
プログラムの上に
ドロップしておく！

```
import  ddf.minim.*;
Minim  minim;
AudioSnippet  crash;
void setup(){
  size( 400, 400 );
  minim = new Minim( this );
  crash =
       minim.loadSnippet( "crash.mp3" );
}
void stop(){
  crash.close();
  minim.stop();
  super.stop();
}
void mousePressed(){
  crash.rewind();
  crash.play();
}
```

複数の音を使う場合
変数として定義を
繰り返すだけ！

mp3やwavもOK!!

```
import  ddf.minim.*;
Minim  minim;
AudioSnippet  taiko;
AudioSnippet  kane;
void setup(){
    size( 400, 400 );
    minim = new Minim( this );
    taiko = minim.loadSnippet( "taiko.mp3 " );
    kane = minim.loadSnippet( "kane.wav" );
}
void stop(){
    taiko.close();
    kane.close();
    minim.stop();
    super.stop();
}
void mousePressed(){
    if( mouseX < width/2 ){
       taiko.rewind();
       taiko.play();
    } else {
       kane.rewind();
       kane.play();
   }
}
```