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：janken**
  - 400x300のウインドウを作成し，その中心に150ピクセルの円を表示せよ．また，その円内をクリックするたびにAさんとBさんがランダムにジャンケンの手を出し，その結果を下図のように標準出力するプログラムを作成せよ

クリック!

```
Aさんはグー
Bさんはチョキ
Aさんの勝ち
```

クリック!

```
Aさんはグー
Bさんはグー
引き分け
```

クリック!

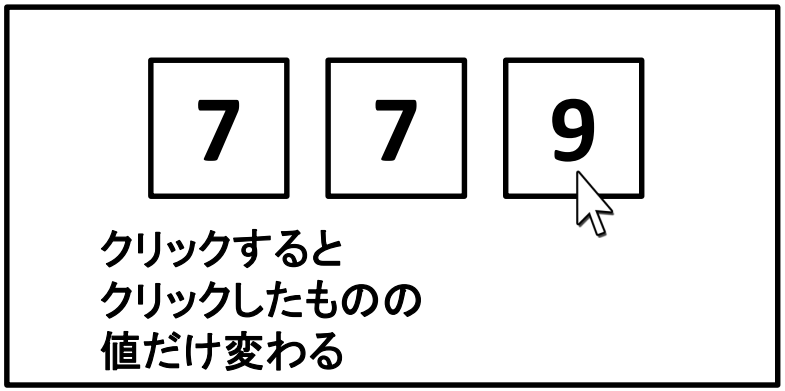
```
Aさんはチョキ
Bさんはチョキ
引き分け
```

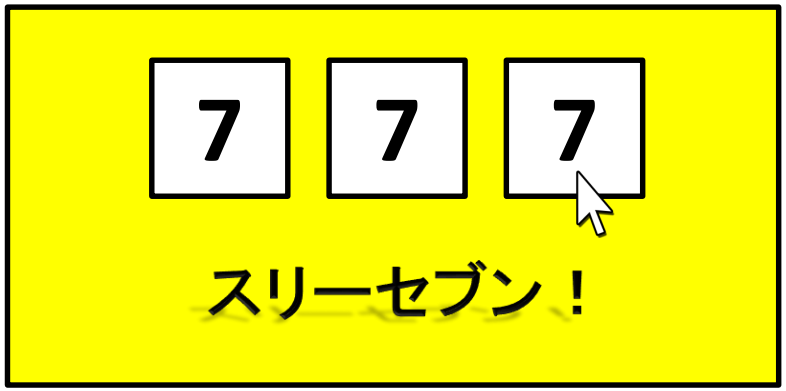
クリック!

```
Aさんはグー
Bさんはパー
Bさんの勝ち
```

プログラミング演習I （第5回） 課題

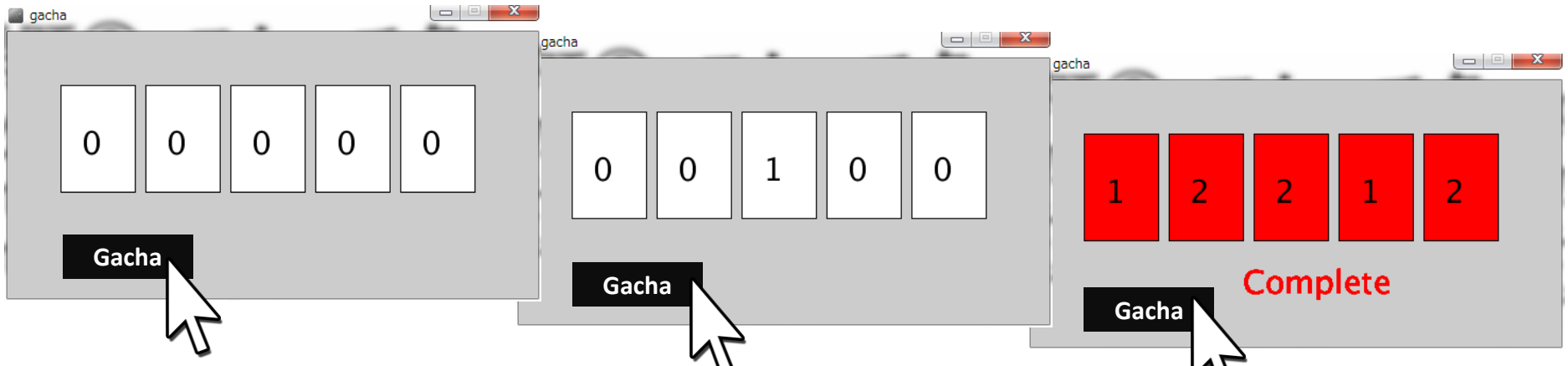
- **基本課題②　スケッチ名： slot**
  - 3つの四角形を表示し、その四角形内をクリックするとその四角形内の数字の目が7か8か9に変化するプログラムを作成せよ
  - 初期値は789の並びとせよ
  - また、3つの数字の目が一致したときに、画面を華やかにせよ（777，888，999のとき）





プログラミング演習I（第5回）課題

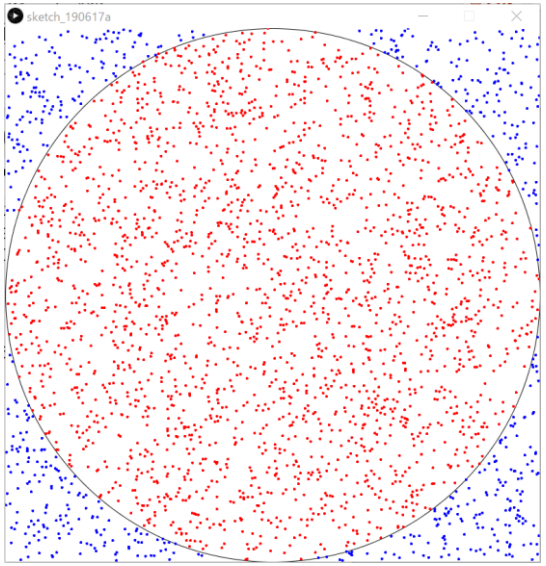
- **基本課題③　スケッチ名：complete_gacha**
  - ウインドウ下部のガチャボタンをクリックする度に5種類のカードの1種類がランダムに選ばれ，枚数が１加算されるプログラムを作成し，それぞれのカードが選ばれた枚数を表示するプログラムを作成せよ（ただしボタン以外では反応しないようにせよ）
  - また，すべてのカードが1枚以上になったら，Completeとウインドウ内にtextを用いて表示し，カードの色を赤色にせよ

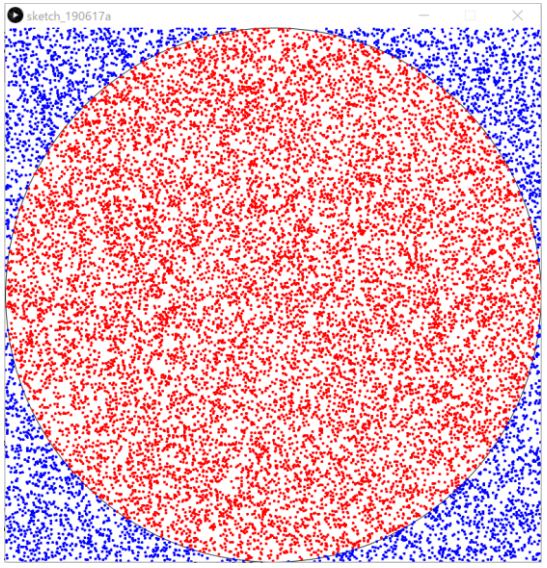


プログラミング演習I （第5回） 課題

発展課題① スケッチ名：MonteCalro

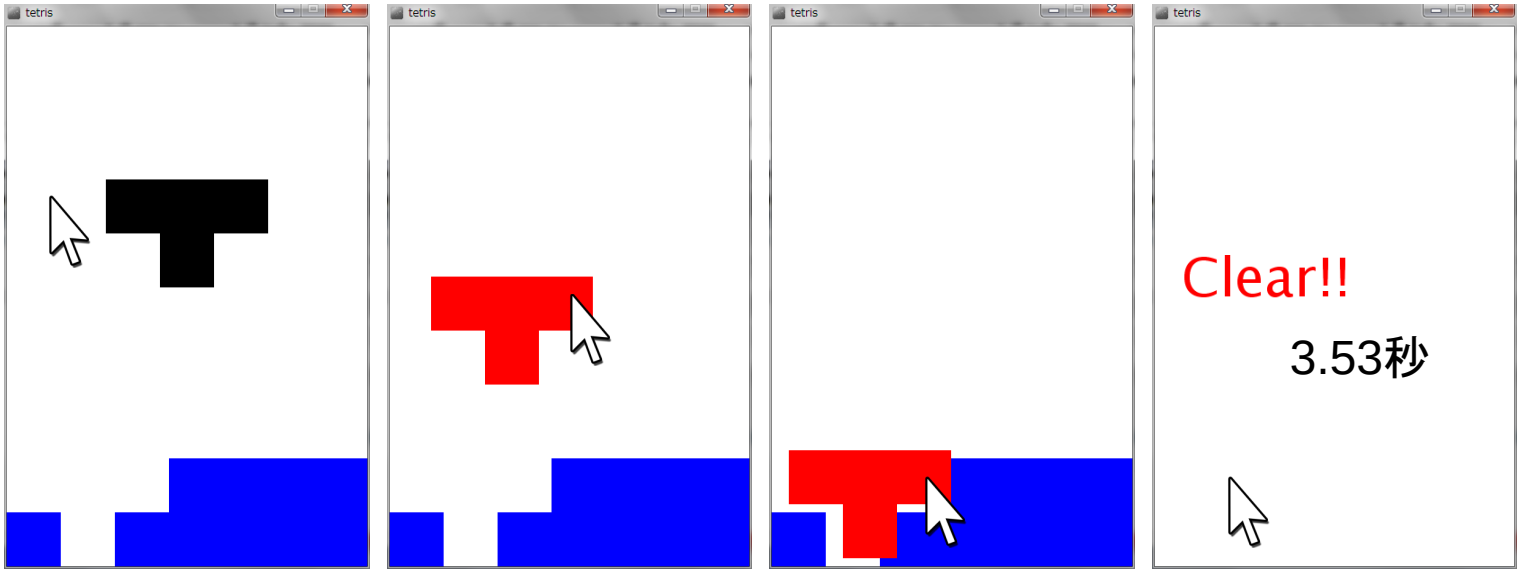
- モンテカルロ法とは，確率に関係のないものを確率を利用して計算するというものである．ここで円周率の近似値を求めたい．
- まず，800x800のウインドウ内に直径800ピクセルの円を描け
- 800x800の正方形の面積は800x800，直径800ピクセルの円の面積は800x800xπとなるため，円の面積/正方形の面積はπ/4となる
- draw()のたびに，画面内でランダムにX座標，Y座標を取得し，その点が円の外側の時には直径3ピクセルの青丸，内側の時には直径3ピクセルの赤丸を表示せよ
- 円内の点の数/全ての点の数 ≒ 円の面積/正方形の面積 ≒ π/4 を利用すると，π ≒ 4 * 円内の点の数 / 全ての点の数となる．これを利用して円周率を draw() を100回実施するたびに標準出力せよ



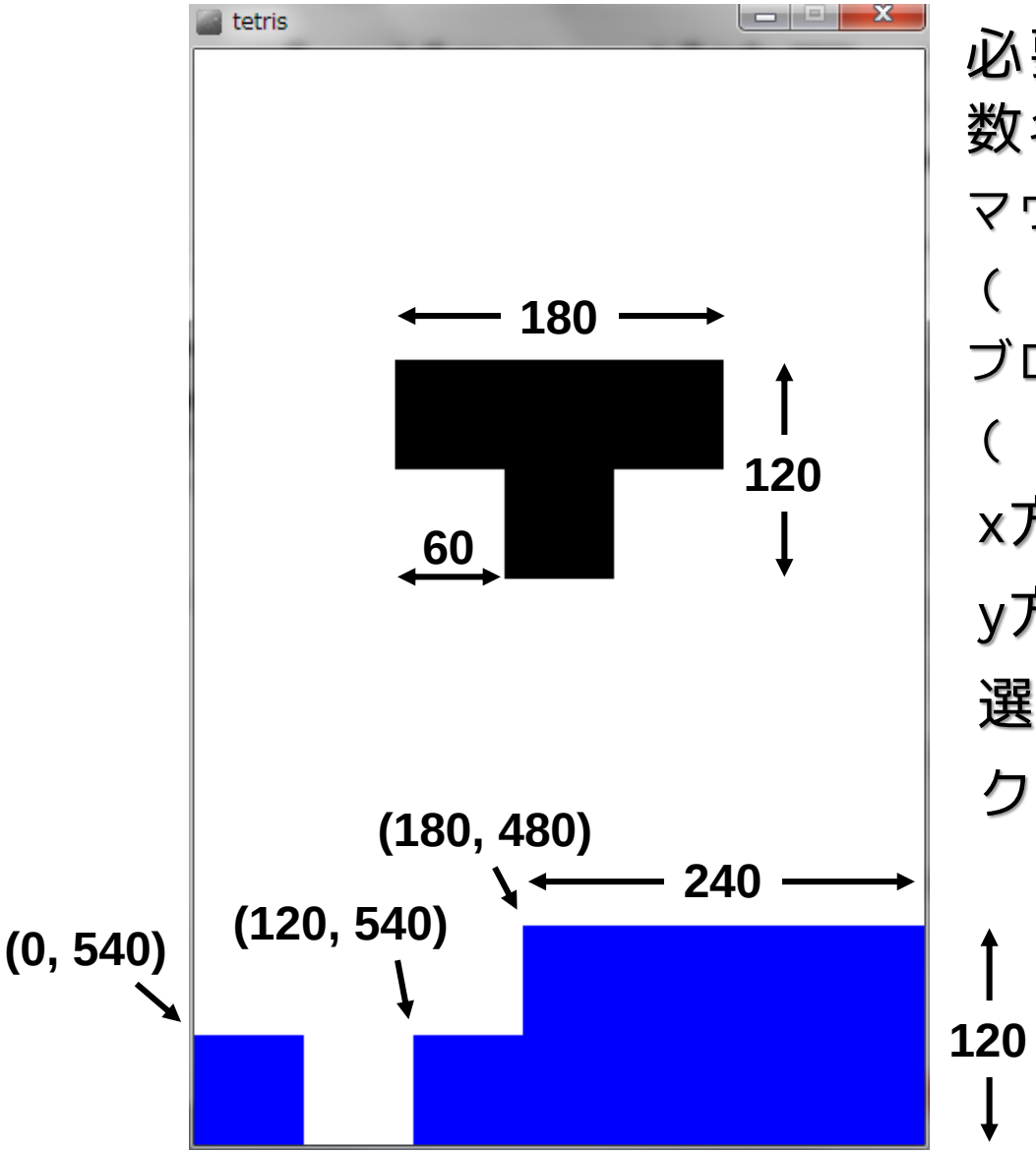


プログラミング演習I （第5回） 課題

- **発展課題②　スケッチ名：Tetris**
  - 420x600のウインドウ内に、下図に示す横180ピクセル縦12ピクセルの凸型の図形を描き（黒色）、図形内でマウスボタンを押すと選択状態（赤色）になり、ボタンを離すまで移動できるようにせよ。
  - なお、移動においては相対位置をキープするようにして下さい。また、ドラッグ中は図形が赤色になるようにせよ。
  - また、下図のように他のブロックを用意し、その隙間にピッタリハマった場合に「Clear!!」と表示したうえで経過時間を表示せよ



ヒント



必要な変数は何か？　右に変数名を書き込もう！

マウスの座標

（　　，　　）

ブロックの基本座標

（　　，　　）

x方向の相対座標:

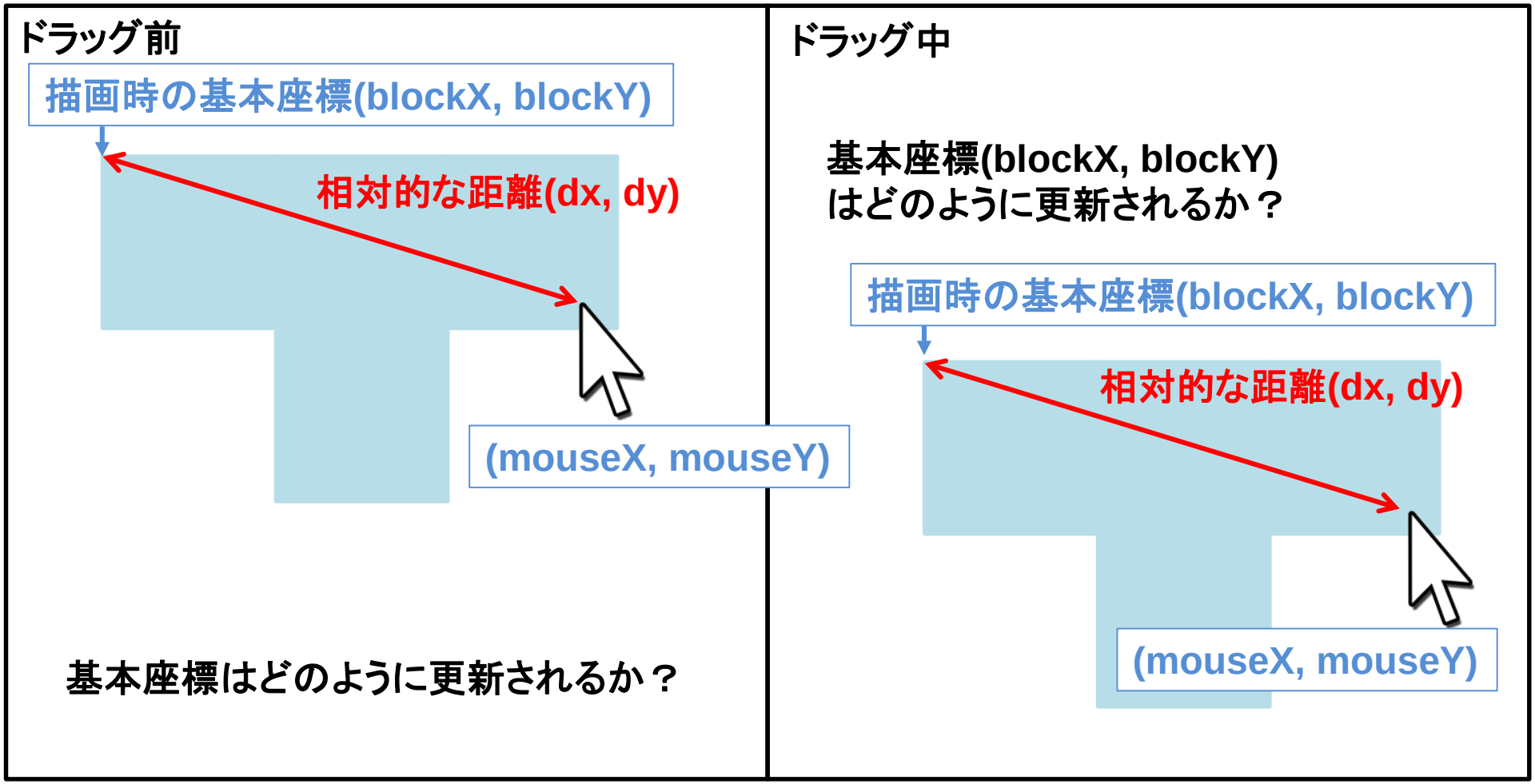
y方向の相対座標:

選択の有無:

クリアしたかどうか:

ヒント

- 相対座標はどのようにして保持し，その時の座標にどのようにして反映するか整理！



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

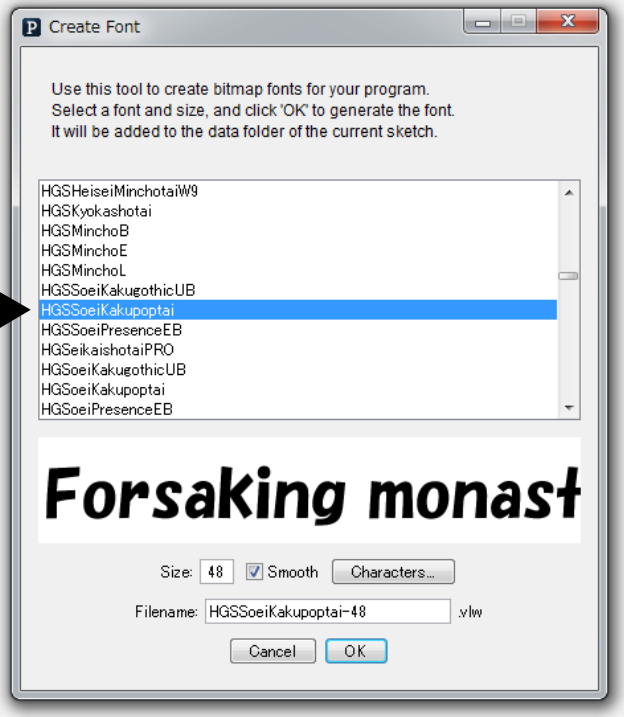
```
PFont myFont; // フォント

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

③ millis()でミリ秒単位の経過時間を取得する

- アプリケーションが起動されてからの時間は millis() で取得することが可能

```
int iStartMillis;
boolean bFlagStart = false; // スタートしたかどうかのフラグ
void setup() {
  size(300, 150);
  fill( 0 ); // 文字色を黒色に設定
}
void draw() {
  background( 255 );
  if( bFlagStart ){ // スタートしていたら～
    text( millis()-iStartMillis, 20, 90 ); // 差分で経過時間を表示
  }
}
void mousePressed(){
  bFlagStart = true; // クリックされたらスタートフラグを立てる
  iStartMillis = millis(); // スタートの経過時間をセット
}
```