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：basic_Donut**
  - 400x400の大きさのウインドウの背景を白色にし、そのウインドウの中央に、**内径が150ピクセル（半径75ピクセル）**、**外径が300ピクセル（半径150ピクセル）**の茶色(103，67，45)のドーナツを描け
  - また、ドーナツの部分（茶色の部分）をクリックすると、Donut!と標準出力するようにせよ（ドーナツ以外の部分をクリックしても反応しないようにせよ）

プログラミング演習I（第5回） 課題

- **基本課題②　basic_Janken**
  - 300x300のウインドウを作成し，(50,150)(150,150)(250,150)の位置にそれぞれ**半径50**ピクセルの円を表示せよ．いずれかの円をクリックしたとき，左ならグー、中ならチョキ，右ならパーの手をあなたは選んだものとする．このときコンピュータはランダムにグーチョキパーを選び，両者のじゃんけんの手が表示され，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．円の外をクリックした場合は反応しないようにし，何度でもじゃんけんできるようにせよ

クリック!

```
あなたはグー
コンピュータはチョキ
あなたの勝ち
```

クリック!

```
あなたはグー
コンピュータはグー
引き分け
```

クリック!

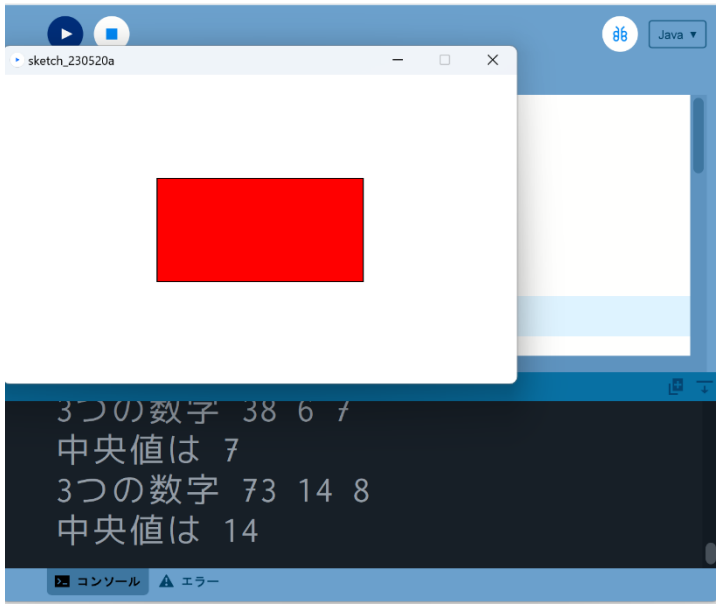
```
あなたはチョキ
コンピュータはチョキ
引き分け
```

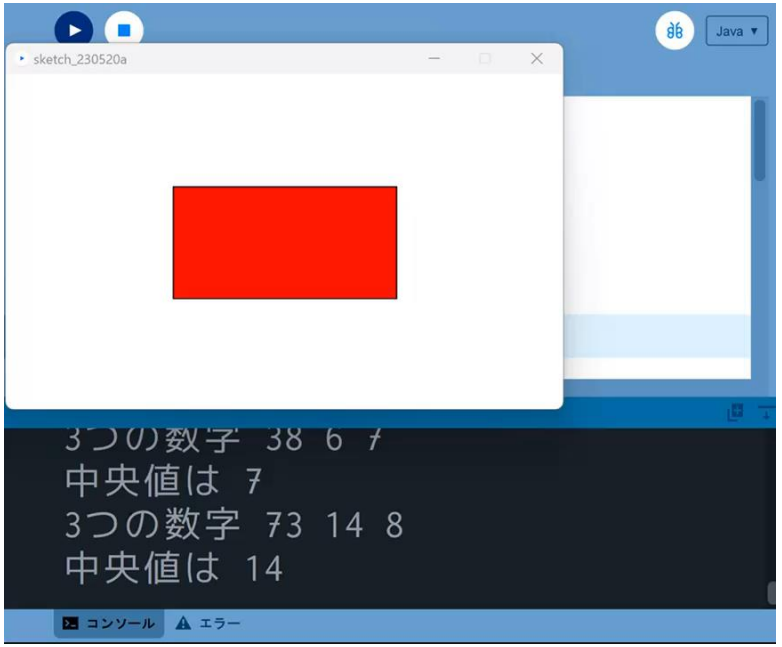
クリック!

```
あなたはグー
コンピュータはパー
コンピュータの勝ち
```

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：basic_Middle**
  - 500x300の白色背景のウインドウ内に、(150, 100)を左上座標とし、横幅200、縦幅100の赤色の四角形を描画せよ。
  - また、その赤色の四角形内をクリックしたときに、1～100までの整数値で3つの数値をランダムに生成し、標準出力に出力せよ。
    - (int)random(1, 101) で1～100までの整数は取得できるよ！
  - さらに、出力した3つの数字の中央値を求め、その結果も出力せよ。
    - 中央値は、真ん中の値





ヒント

- 基本課題①
  - ドーナツの茶色の部分の判定条件はどうなる？
  - mousePressedの中で標準出力しよう！
- 基本課題②
  - どういう条件になるか、手書きしてみよう
    - ユーザのじゃんけんの手はどうとる？
    - コンピュータのじゃんけんの手はランダムに
    - 勝敗の条件は？
  - mousePressedの中で標準出力しよう！
- 基本課題③
  - 四角系の条件判定は？
  - 3つの数字の中央値はどういう特徴をもつ？

プログラミング演習I （第5回） 課題

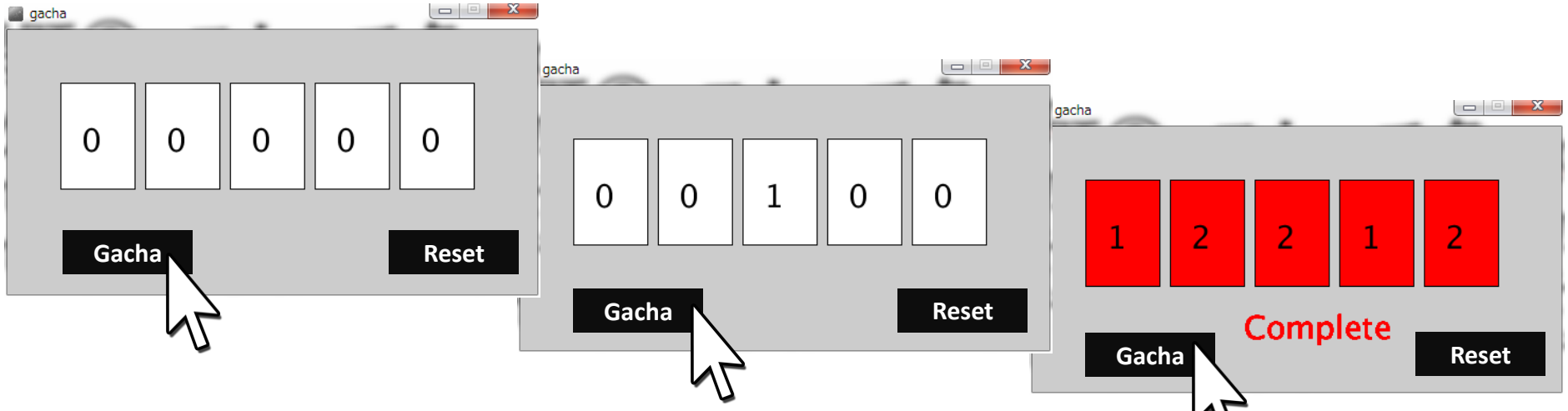
- **発展課題① advanced_Action**
  - 800x400のウインドウを作成し、画面下部に地面のような領域を描画せよ。また、円（または自身のキャラクタ）をX座標を400、Y座標は地面に接するように初期配置し、静止させておけ。
  - キーボードの左キーで左方向への速度を+1し、右キーで右方向への速度を+1せよ。例えば、初期状態から右キーを2回押すと、X座標の正の方向に1フレームあたり2ピクセル動くようになる。また、その状態で左キーを2回押すと静止する。さらに、その状態で左キーを1回押すと1フレームあたり1ピクセル左に動くようになる。
  - さらに、キーボードの上キーで上方向にジャンプするようにせよ。上方向への速度は毎回同じで良い（最高到達点はおなじになる）
  - 画面右端、左端まで来ると、逆側から登場するようにせよ。

  - 動きはこんな感じ：
    https://gyazo.com/bce8324df8086995dd42f1cdfbbd0563


sketch_220521a

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：advanced_CompleteGacha**
  - ウインドウ下部のGacha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
  - なお，どの状態でもウインドウ下部のResetボタンを押すとすべてのカードの枚数を0枚にせよ（すでにCompleteしている場合もすべてを0枚に戻し，カードの色も元通りにせよ）



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
}
```

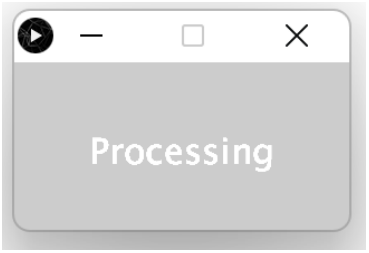
- textSize()は、fill()やstroke()と同様に何回でもパラメータを変えて指定できるので、大きさの違う文字を混在させることができる。

今日使うテクニック

② text() で表示する場所をわかりやすくする方法

- text("出力する文字列", x座標, y座標); の、x座標とy座標が微妙に扱いにくい。
- textAlign(CENTER, CENTER); と書いて、縦横中央揃えしておくとわかりやすくて良いよ！
  - LEFT/RIGHT, TOP/BOTTOMも設定できるよ！

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

void draw() {
  text( "Processing", 150, 75 ); // 中央に表示されるよ！
}
```



今日使うテクニック

③ 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

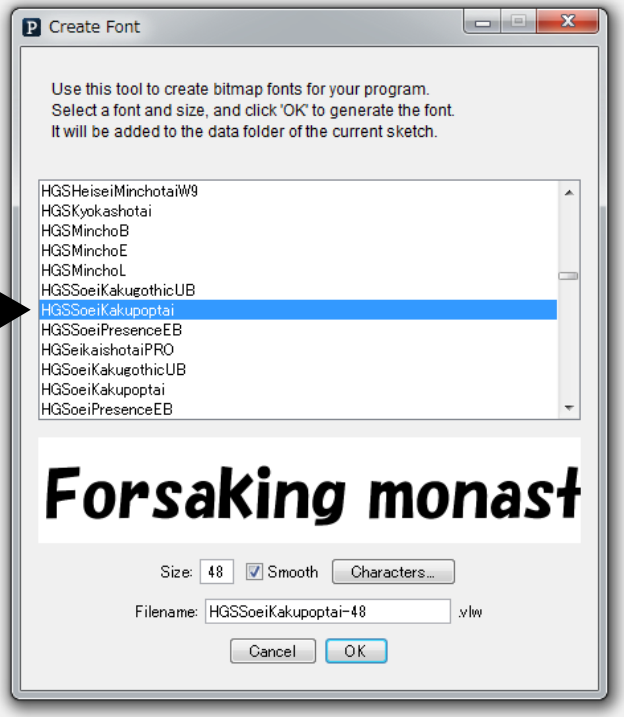
```
PFont myFont; // フォント

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。