

```

/*
 * MCR Gate System シリアル通信
 * Processing 側サンプル
 * 開発環境 Ver3.3
 */

// シリアルライブラリを取り入れる
import processing.serial.*;

// myPort (任意名) というインスタンスを用意
Serial myPort;

/*===== 変数宣言 =====*/
int x; //図形の X 座標の変数を用意
int NUM = 6; //センサーの数
int[] receivedData = new int[NUM]; //センサーの値を格納する配列
int secondIcntM = 0; // OUT コース:分
int secondIcntS = 0; // OUT コース:秒
int secondITIME = 0; // OUT コース:センチ秒
int inTime_m = 0; // IN コース:分
int inTime_s = 0; // IN コース:秒
int inTime_c = 0; // IN コース:センチ秒

/*===== 初期設定 =====*/
void setup(){
  // 画面サイズ
  size(1080, 780); // 1024 x 768 だと下が切れる
  textFont(createFont("x", 15)); // 日本語を表示用。バージョンアップで不要?
  //frameRate(10); // 1 秒間に 10 回 draw()実行 (使うかもしれないので残す)

  // シリアルポートの設定
  // Arduino が接続されているシリアルポートにあわせて書き換える
  myPort = new Serial(this, "COM3", 250000);
}

/*===== 描画ループ =====*/
void draw(){

  // 受信バッファから描画用変数に格納
  secondIcntM = receivedData[2];
  secondIcntS = receivedData[3];
  secondITIME = receivedData[0];
  inTime_m = receivedData[4];

```

```

inTime_s = receivedData[5];
inTime_c = receivedData[1];

// 分,秒,センチ秒のバラバラのデータを時計形式に結合
String outTime = nf(secondIcntM, 2)+":"+nf(secondIcntS, 2)+":"+nf(secondITIME, 2);
String inTime = nf(inTime_m, 2)+":"+nf(inTime_s, 2)+":"+nf(inTime_c, 2);
println(outTime);          // デバッグ用に出力
println(inTime);          // デバッグ用に出力

background(255);          // 背景を指定色:白で塗りつぶす

//===OUT コースタイム表示===
textSize(90);             // 文字サイズ
fill(0, 0, 255);         // 文字色:BLUE
text("OUT", 30, height*15/100); // 表示するテキスト, x座標, y座標

textSize(220);           // 文字サイズ
fill(0, 0, 255);        // 文字色:BLUE
text(outTime, 80, height*40/100);

//===IN コースタイム表示===
textSize(90);            // 文字サイズ
fill(255, 0, 0);        // 文字色:RED
text("IN", 30, height*65/100); // 表示するテキスト, x座標, y座標

textSize(220);           // 文字サイズ
fill(255, 0, 0);        // 文字色:RED
text(inTime, 80, height*90/100);

}

void serialEvent(Serial p){
  if ( myPort.available() >= 7 ) { // ヘッダ + 時間データ で合計7バイト
    if ( myPort.read() == 'H' ) { // ヘッダ文字を見つけたところから読み取る
      receivedData[0] = myPort.read(); // OUT:分読み込み
      receivedData[1] = myPort.read(); // OUT:秒読み込み
      receivedData[2] = myPort.read(); // OUT:センチ秒読み込み
      receivedData[3] = myPort.read(); // IN:分読み込み
      receivedData[4] = myPort.read(); // IN:秒読み込み
      receivedData[5] = myPort.read(); // IN:センチ秒読み込み

      /*int high1 = myPort.read();

```

```
int low1 = myPort.read();
int high2 = myPort.read();
int low2 = myPort.read();
receivedData[0] = high1*256 + low1; // OUT:分読み込み
receivedData[1] = high2*256 + low2; // OUT:秒読み込み
receivedData[2] = myPort.read();// OUT:センチ秒読み込み
receivedData[3] = myPort.read(); // IN:分読み込み
receivedData[4] = myPort.read(); // IN:秒読み込み
receivedData[5] = myPort.read(); // IN:センチ秒読み込み*/
}
}
}
```