



---

# プログラミング演習2

# ファイル入出力

---

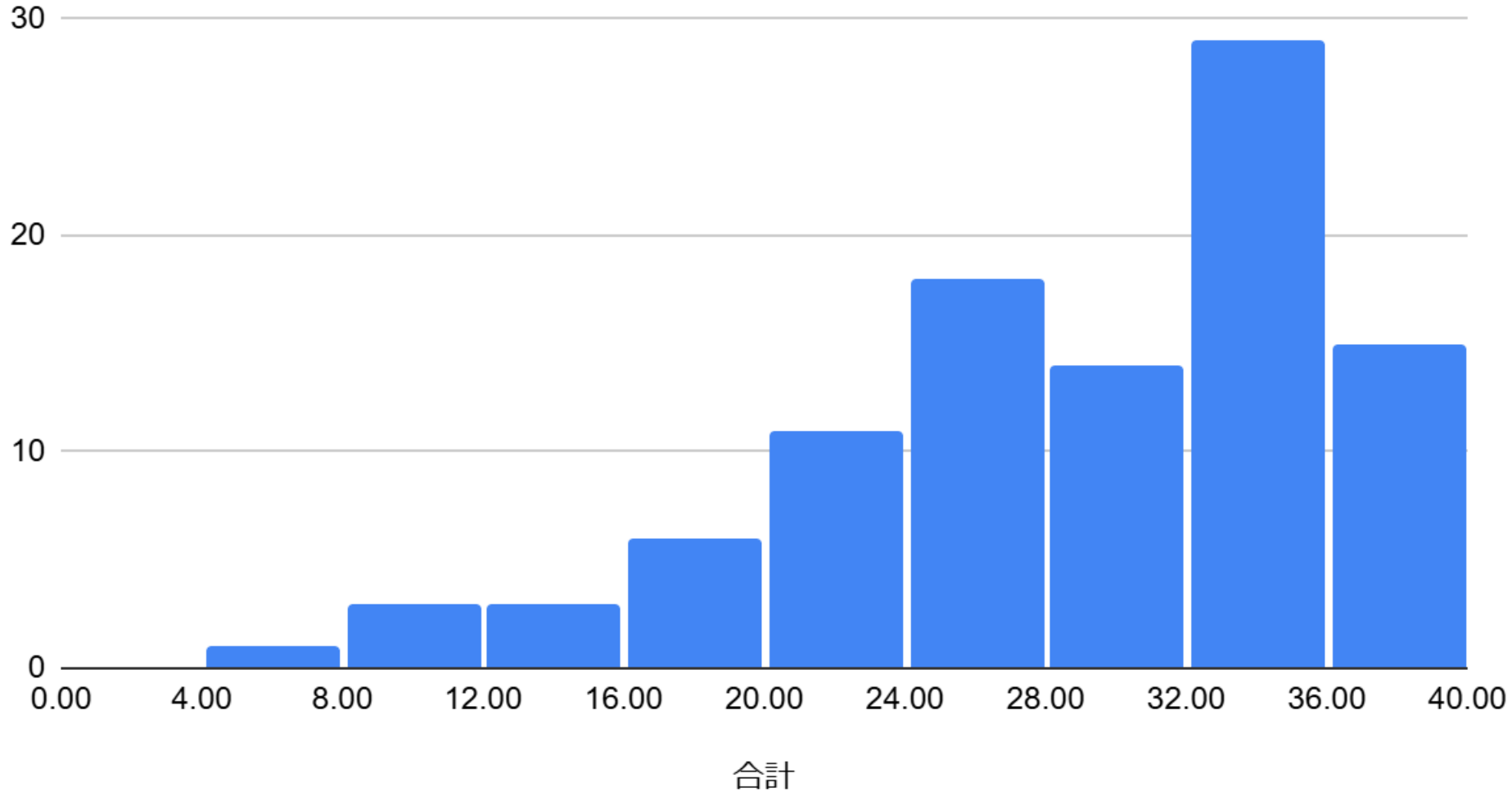
中村、小林、橋本、辻野

# 平均30.58点 / 40.0点

明治大学総合数理学部  
先端メディアサイエンス学科  
中村研究室



## 「合計」のヒストグラム



# 今日のスケジュール

---



- 13:30-13:45 試験の解説 (ざっと)
- 13:45-15:10 今日の解説
- 15:20-16:45 課題に取り組む時間
- 16:45-17:00 課題解説の時間



- 最終回はグループワーク（1グループ3～5名程度）の成果の発表会の予定です
  - グループワークでは、研究室を複数グループ（人数によって変動）に分けますが、誰と組むことになるかはわかりませんので、助け合いレベルアップしましょう
- 再履修組は、昨年度の点数採用 or 別課題に取り組む

## 最終課題について

明治大学総合数理学部  
先端メディアサイエンス学科  
中村研究室



- 締め切り：2023/01/23（月）13:00厳守
- 提出場所：Google Drive
- 以下の課題を作成せよ（各10点）

（注意）過去に提出したひとは、それとは違うものとする

1. 「クラス」を用いてスマートフォンやコンピュータなどで触れるユーザインタフェースを模倣せよ。または、現実世界の模倣をせよ。ただし、2つ以上のクラスからなるものを実現せよ（10点）
2. 最低5種類のブラシがあるペイントツールを作成せよ。できるだけオリジナリティの高いブラシがあるものとせよ。また、そのペイントツールを利用した出力例も画像として提出せよ。配布する SampleBrushProject を継承して利用しても良い（10点）

# 今日やること

---



- 前回やった記録をどう復元する？
- ハイスコアをどう記録する？
- 実験などの結果をどう記録する？
- 描画した結果をどう再現する？

# クリック数を保存せよ



ウィンドウ上でクリックした回数を表示するプログラムを作成せよ。なお、クリックした回数は随時保存し、次回起動した時にはそのクリック回数から増やしていくようにせよ

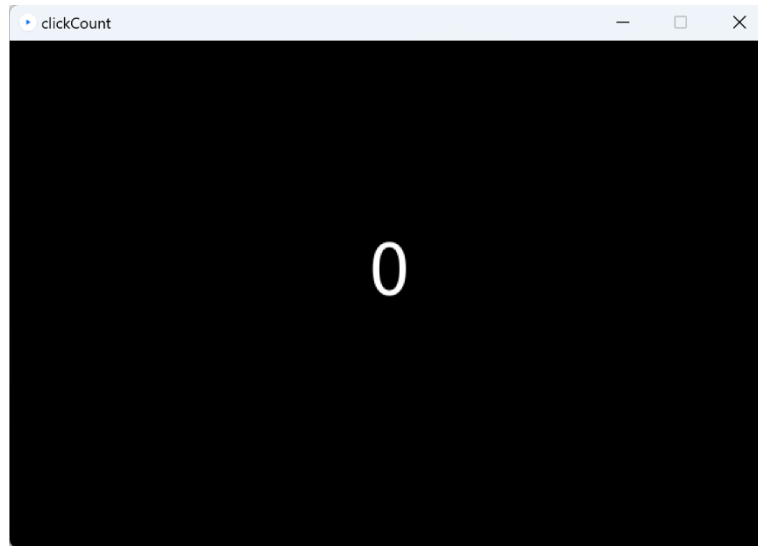
- クリックした回数を表示し続けるのであれば、クリック回数を保存する `click_count` という変数を定義してカウントアップしつつ表示する
- 終了 → 起動したときには当然0に戻ってしまうがどうやったらよいだろうか？

**状態をファイルに保存する！**

# まずは



- 配布したclickCount.zipというファイルを解凍（展開）してclickCount.pdeを開いてください



- クリックしたら数が増える
- 再起動するとどうなる？

# どうやって復元する？



- プログラムは実行するたびにすべての変数の内容が初期化される
- 前回起動していたときの状態を引き継ぐには、どうしたらよいだろうか？

おきのどくですが  
ぼうけんのしょ 1 ばんは  
きえてしまいました。

# 忘れないためどうする？



- 忘れては困るものをパソコンやスマホ以外で記録するときはどうしますか？
  - ノートに書き込む
  - 手帳に記録
  - ポストイットに書く  
などなど



# 記録したり呼び出したり



- これまで

- 変数に値を保存（代入）し、変数を使うことで値を取り出し表示したりしていた

- 丸のx, y座標と、スピード
- ゲームのscore
- 電光掲示板のライトのON/OFF情報

アプリケーションを再起動したら消えてしまう



ファイルに記録して再起動後にも使う！

# ファイルを経由する



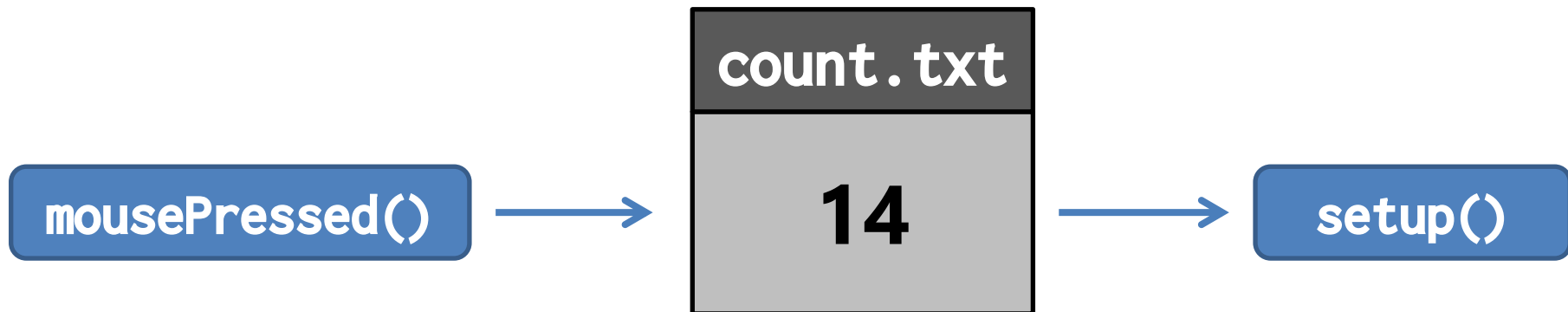
- プログラムとは別のファイルに一旦情報を記録しておき、そのファイルからまた情報を読み出す！
- セーブ&ロードしよう！
  - 記録する
    - 何かの値をファイルに記録しておく  
(ハイスコアや状態など)
  - 読み込む(再生する)
    - 記録した値をファイルから呼び出す  
(ハイスコアや状態など)

# クリック数を保存せよ



クリックした回数は随時保存し、次回起動した時にはそのクリック回数から増やしていくようにせよ

- `mousePressed()` の度に `click_count` の値を増やし `count.txt` に保存する
- 起動時 (`setup`) の時に `count.txt` の値を `click_count` に読み込む



# ファイルに書き込む



```
saveStrings("ファイル名", String型の配列);
```

- String型の配列の内容を、1行毎にファイルに書き込む

(例) Stringの配列linesがあるとき、lines[0]は1行目に、lines[1]は2行目に保存される

– mousePressedの中に下記を書き込もう！

```
String[] lines = new String[1];  
lines[0] = str(click_count);  
saveStrinsg("count.txt", lines);
```

– **str( 何らかの値 )** で整数などを文字列に変換

# 実行して終了して確認



- clickCountを実行してクリックし、そのクリックした回数が count.txt というファイルに保存されているかを確認しよう！

The screenshot displays a Windows environment with several overlapping windows. In the background, a file explorer window shows a list of files with columns for '名前' (Name), '更新日時' (Last Modified), and '種類' (Type). The file 'clickCount.pde' is highlighted, with a last modified date of '2023/11/12 11:57' and a type of 'Proc'. In the foreground, a black application window titled 'clickCount' displays the number '14' in white. To the right, another file explorer window shows the same file list, but with 'count.txt' also visible. In the bottom right corner, a text editor window titled 'count.txt' is open, showing the number '14' in the main editing area. The text editor has a menu bar with 'ファイル' (File), '編集' (Edit), and '表示' (View).

# ファイルから読み込む



```
String[] lines = loadStrings("ファイル名");
```

- ファイルの中身を1行毎にString配列に格納
  - 1行目の値は `lines[0]` に、2行目の値は `lines[1]` に入っている
  - 以下を `setup` に書き込もう！

```
String[] lines = loadStrings("count.txt");  
click_count = int(lines[0]);
```

- **int( 文字列 )** で、文字列を整数に変換

# さて復元できました？



- やったこと
  - `click_count.txt` に `click_count` という変数の値を文字列にして保存
  - `click_count.txt` から値を読み込み、`int`型に変換して `click_count` という変数に代入
- 基本的な仕組みはこれだけ！

```
int click_count = 0;

void setup() {
  size(600, 400);
  textSize(64);
  String[] lines = loadStrings("count.txt");
  if(lines != null ){
    click_count = int(lines[0]);
  }
}

void draw() {
  background(0, 0, 0);
  text(click_count, width/2, height/2);
}

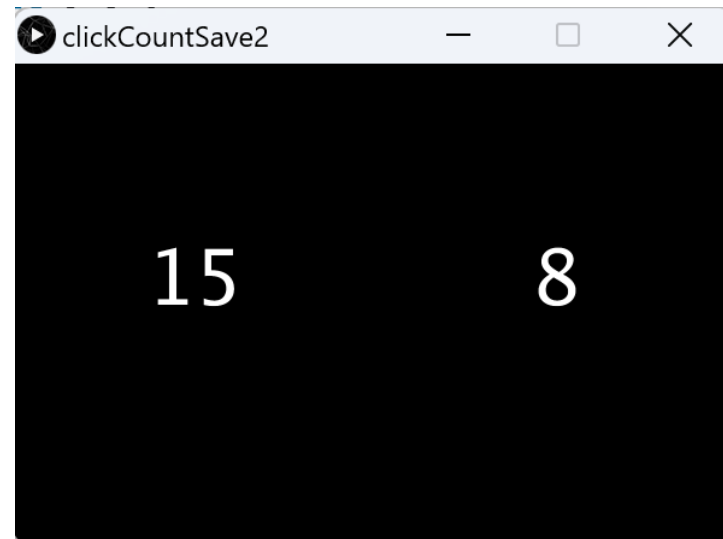
void mousePressed() {
  click_count++;
  String[] lines = new String[1];
  lines[0] = str(click_count);
  saveStrings("count.txt", lines);
}
```



# 左右のクリック数を保存



ウィンドウの左側・右側をそれぞれクリックした回数を表示するプログラムを作成せよ。なお、クリックした回数は随時保存し、次回起動した時にはそのクリック回数から増やしていくようにせよ



まずは `click_count1`, `click_count2` を用意

# 複数の値を保存する



- 現在の状態をファイルに保存

```
click_count1 = 15
```

```
click_count2 = 8
```

- ファイル ( count.txt ) に例えば下記のように保存

```
15
```

```
8
```

- また、`setup()` で1行目は`click_count1`,  
2行目は`click_count2`の値として読み込む

```
int click_count1 = 0;
int click_count2 = 0;

void setup() {
  size(600, 400);
  textSize(64);
  String[] lines = loadStrings("count.txt");
  if(lines != null ){
    click_count1 = int(lines[0]);
    click_count2 = int(lines[1]);
  }
}

void draw() { /* 省略 */ }

void mousePressed() {
  click_count++;
  String[] lines = new String[2];
  lines[0] = str(click_count1);
  lines[1] = str(click_count2);
  saveStrings("count.txt", lines);
}
```



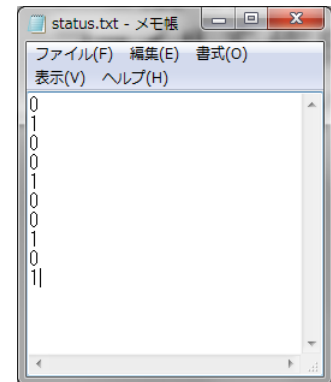
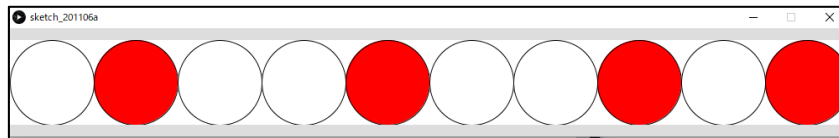
# 電光掲示板



1000x1000のウィンドウに横に10個、縦に1個並んだ電光掲示板について、丸をクリックする度に赤色、白色と塗りつぶしの色が入れ替わるようにせよ。またファイルから赤白の状態を読み込むようにせよ

- メモ帳で、1行に0または1だけを書いた10行のファイルを作成する（ファイル名はstatus.txt）
  - PDEと同じフォルダに保存する
- status.txt をプログラムで読み込み、1行目をlights[0]に、2行目をlights[1]にと値を順に割り当てる
- lightsの値に応じて電光掲示板の状態を制御する

0,1,0,0,1,0,0,1,0,1





```
int[] lights = new int[10];
```

```
void setup()
```

```
{  
  size(1000, 100);  
  for(int x=0; x<10; x++){  
    lights[x] = 0;  
    x++;  
  }  
}
```

```
void draw()
```

```
{  
  background( 255 );  
  for(int x=0; x<10; x++){  
    if(lights[x] == 1){  
      fill(255, 0, 0);  
    } else {  
      fill(255);  
    }  
    circle(100*x+50, 50, 100);  
  }  
}
```

```
void mousePressed()
```

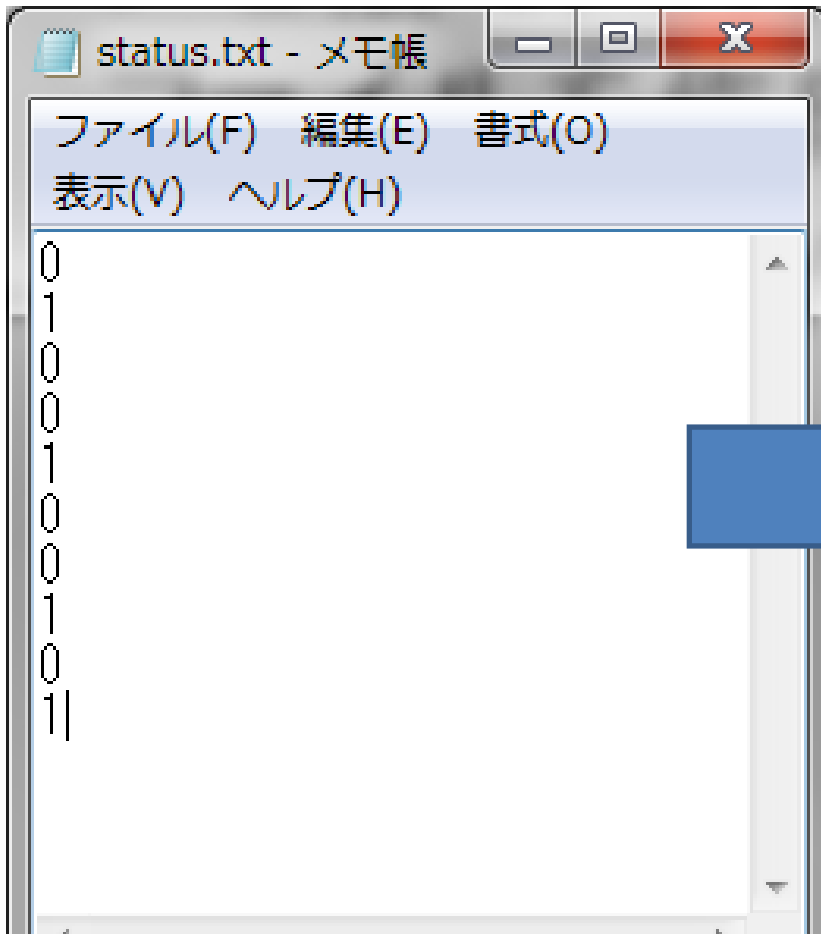
```
{  
  for(int x=0; x<10; x++){  
    if(dist(100*x+50,50,mouseX,mouseY) <= 50){  
      if(lights[x] == 1){  
        lights[x] = 0;  
      } else {  
        lights[x] = 1;  
      }  
    }  
  }  
}
```

# 電光掲示板プログラム



- ファイルから読み込む

```
String [] lines = loadStrings("status.txt");
```



配列	値
lines[0]	"0"
lines[1]	"1"
lines[2]	"0"
lines[3]	"0"
lines[4]	"1"
lines[5]	"0"
lines[6]	"0"
lines[7]	"1"
lines[8]	"0"
lines[9]	"1"


# 電光掲示板プログラム



setup()のみを変更してファイルから読み込む

```
int [] lights = new int [10];

void setup() {
  size( 300, 100 );
  String[] lines = loadStrings( "status.txt" );
  lights[0] = int( lines[0] );
  lights[1] = int( lines[1] );
  lights[2] = int( lines[2] );
  lights[3] = int( lines[3] );
  lights[4] = int( lines[4] );
  lights[5] = int( lines[5] );
  lights[6] = int( lines[6] );
  lights[7] = int( lines[7] );
  lights[8] = int( lines[8] );
  lights[9] = int( lines[9] );
}
```



```
int[] lights = new int [10];

void setup() {
  size( 300, 100 );
  String [] lines = loadStrings( "status.txt" );

  for(int x=0; x<10; x++){
    lights[x] = int( lines[x] );
  }
}
```

# こんなエラーが出たら



- status.txt がPDEと同じフォルダに入っていないということ

```
lights_savveload | Processing 2.0.3
File Edit Sketch Tools Help
lights_savveload
int [] lights = new int [20];
void setup() {
  size( 600, 100 );
  String [] lines = loadStrings( "status.txt" );
  lights[0] = int( lines[0] );
  lights[1] = int( lines[1] );
  lights[2] = int( lines[2] );
  lights[3] = int( lines[3] );
  lights[4] = int( lines[4] );
  lights[5] = int( lines[5] );
  lights[6] = int( lines[6] );
  lights[7] = int( lines[7] );
  lights[8] = int( lines[8] );
  lights[9] = int( lines[9] );
  lights[10] = int( lines[10] );
  lights[11] = int( lines[11] );
  lights[12] = int( lines[12] );
  lights[13] = int( lines[13] );
  lights[14] = int( lines[14] );
  lights[15] = int( lines[15] );
  lights[16] = int( lines[16] );
  lights[17] = int( lines[17] );
  lights[18] = int( lines[18] );
  lights[19] = int( lines[19] );
}
Done Saving.
The file "status.txt" is missing or inaccessible, make sure the URL is valid or that the file has
been added to your sketch and is readable.
5
```

# 電光掲示板プログラム



```
void mousePressed(){
    for(int x=0; x<10; x++){
        if(dist(100*x+50, 50, mouseX, mouseY) <= 50){
            if(lights[x] == 1){
                lights[x] = 0;
            } else {
                lights[x] = 1;
            }
        }
    }
}

String[] saveLines = new String[10];
for(int x=0; x<10; x++) {
    saveLines[x] = str( lights[x] );
}
saveStrings("status.txt", saveLines);
}
```

# 電光掲示板プログラム

明治大学総合数理学部  
先端メディアサイエンス学科  
中村研究室



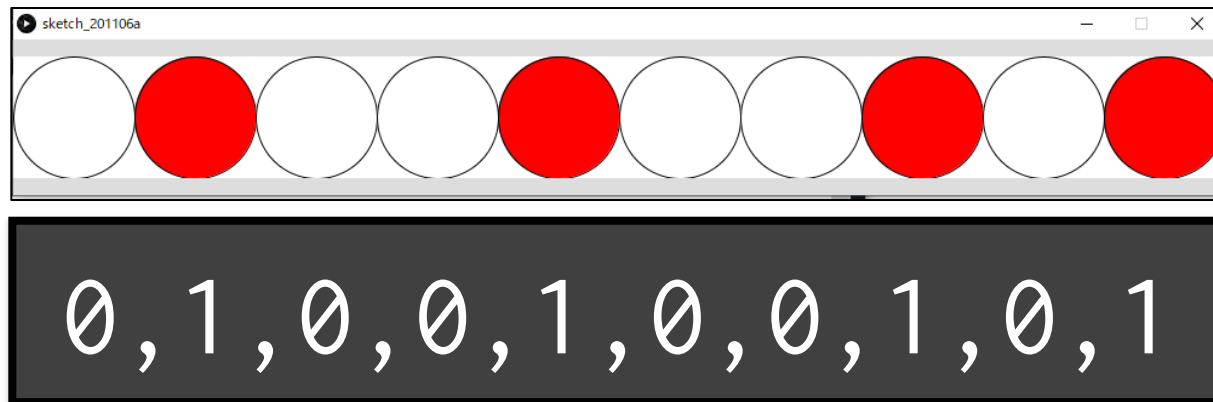
```
void mousePressed(){
  for(int x=0; x<10; x++){
    if(dist(50+100*x, 50, mouseX, mouseY) <= 50){
      // 1,0,1,0と切り替えるならこの方法でもOK
      lights[x] = 1 - lights[x];
    }
  }
}

String[] saveLines = new String[10];
for(int x=0; x<10; x++) {
  saveLines[x] = str( lights[x] );
}
saveStrings("status.txt", saveLines);
}
```

# データの保存方法



- 改行で1つずつではなく、カンマ区切りで保存するやり方もある（こちらが一般的）
  - カンマでそれぞれの状態を羅列



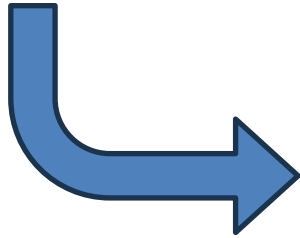
- 読み込むときは、データをカンマ（,）で区切って1つずつ取り出す
  - 取り出し方法は？

# カンマ区切りの取得



```
String[] 文字列配列 = split(文字列, '区切り文字');
```

```
String[] data = split("0,1,0,0,1,0,0,1,0,1", ',');
```



配列	値
data[0]	"0"
data[1]	"1"
data[2]	"0"
data[3]	"0"
data[4]	"1"
data[5]	"0"
data[6]	"0"
data[7]	"1"
data[8]	"0"
data[9]	"1"

# カンマ区切りの取得



```
String[] 文字列配列 = split(文字列, '区切り文字');
```

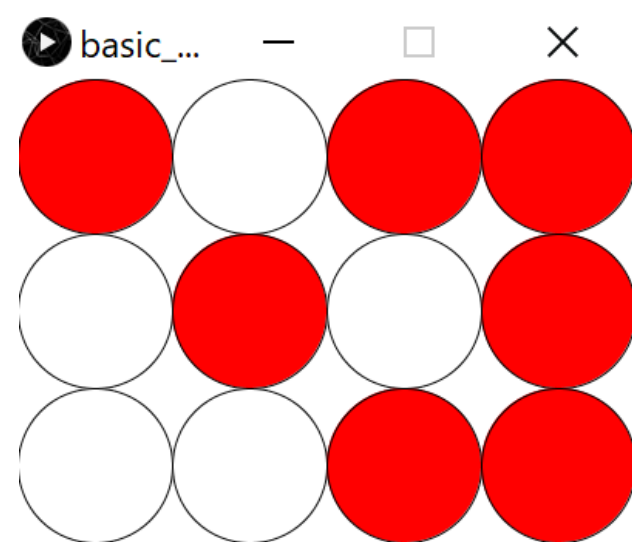
```
String[] data = split("0,1,0,0,1,0,0,1,0,1", ',');  
  
// 0番目、1番目、2番目の値を整数に変換して代入  
for(int i=0; i<data.length; i++){  
    lights[i] = int( data[i] );  
}
```

# 電光掲示板（多次元）



電光掲示板のプログラムを改良して縦に3個、横に4個となるように変更し、左下のように記述したstatus.txtから状態を読み込み下図のように出力せよ。また、値を変えて動作を確認せよ。さらに、クリックのたびに保存し、次回の起動時に結果を引き継ぐようにせよ

```
1,0,1,1
0,1,0,1
0,0,1,1
```





```
int[][] lights = new int[4][3];
```

```
void setup() {  
  size(400, 300);  
  for(int x=0; x<4; x++){  
    for(int y=0; y<3; y++){  
      lights[x][y] = 0;  
    }  
  }  
}
```

```
void draw() {  
  background( 255 );  
  for(int x=0; x<4; x++) {  
    for(int y=0; y<3; y++) {  
      if(lights[x][y] == 1){  
        fill(255, 0, 0);  
      } else {  
        fill(255);  
      }  
      circle(x*100+50, y*100+50, 100);  
    }  
  }  
}
```

```
void mousePressed() {  
  for(int x=0; x<4; x++){  
    for(int y=0; y<3; y++){  
      if(dist(x*100+50, y*100+50, mouseX, mouseY) <= 100){  
        lights[x][y] = 1 - lights[x][y];  
      }  
    }  
  }  
}
```

# 電光掲示板（多次元）



## ヒント

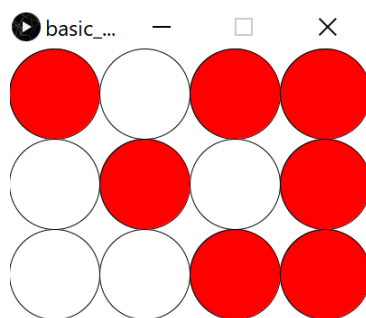
- `status.txt` というファイルをまず用意する（半角であることに注意！）
- `lights` という名前の4x3の配列を作って電光掲示板として実現
- 起動した際にファイル `status.txt` から1行ずつ取得し、そのそれぞれの値を配列に代入していく
  - 1行の値を読み込む時は、各値がカンマ区切りであることを考慮し、カンマごとに区切る場合は `split` を使う
- クリックされるたびに保存することにし、保存する際にも、4x3の配列の値をString型の配列に放り込んでいく

# 電光掲示板（多次元）



- 0行目: `lines[0]`は1,0,1,1という値
  - カンマで区切ったとき、1と0と1と1という値が続いていることになるが、この0番目、1番目、2番目がそれぞれ左上、中左上、中右上、右上に対応つまり0番目は`lights[0][0]`、1番目は…
- 1行目: `lines[1]`は0,1,0,1という値
- 2行目: `lines[2]`は0,0,1,1という値

```
1,0,1,1
0,1,0,1
0,0,1,1
```



<code>lights[0][0]</code>	<code>lights[1][0]</code>	<code>Lights[3][0]</code>
<code>lights[0][1]</code>	<code>lights[1][1]</code>	<code>lights[3][1]</code>
<code>lights[0][2]</code>	<code>lights[1][2]</code>	<code>lights[3][2]</code>

# カンマ区切りの取得



```
String[] 文字列配列 = split(文字列, '区切り文字');
```

```
// linesという配列に1行ずつ文字列を読み込む
String[] lines = loadStrings( "status.txt" );

// lines.lengthですべての行数を取得できる
for(int y=0; y<lines.length; y++ ){
    // line[i] ( i行目 ) の文字列をカンマで分割
    String[] data = split( lines[y], ',' );
    // 0番目、1番目、2番目の値を整数に変換して代入
    lights[ ? ][ ? ] = int( data[0] );
    lights[ ? ][ ? ] = int( data[1] );
    lights[ ? ][ ? ] = int( data[2] );
    lights[ ? ][ ? ] = int( data[3] );
}
```

# splitで分割して読み込む



loadStringsしてカンマで分割して読み込み！

```
int[][] lights = new int[4][3];

void setup() {
  size(400, 300);

  // linesは3行入っている
  String[] lines = loadStrings("status.txt");
  for(int y=0; y<3; y++){
    // 「,」で分割すると4つに分けることができる
    String[] data = split(lines[y], ",");

    // 整数に変換して代入しよう！
    lights[0][y] = int(data[0]);
    lights[1][y] = int(data[1]);
    lights[2][y] = int(data[2]);
    lights[3][y] = int(data[3]);
  }
}
```

```
1,0,1,1
0,1,0,1
0,0,1,1
```

# splitで分割して読み込む



loadStringsしてカンマで分割して読み込み！

```
int[][] lights = new int[4][3];

void setup() {
  size(400, 300);

  // linesは3行入っている
  String[] lines = loadStrings("status.txt");
  for(int y=0; y<lines.length; y++){
    // 「,」で分割すると4つに分けることができる
    String[] data = split(lines[y], ",");
    for(int x=0; x<data.length; x++){
      // 整数に変換して代入しよう！
      lights[x][y] = int(data[x]);
    }
  }
}
```

```
1,0,1,1
0,1,0,1
0,0,1,1
```

# カンマ区切りで保存！



str()で文字列に変換してカンマでつなぐ

```
// 3行分なので3つの要素からなる文字列配列を作る
String[] saveLines = new String[3]
// 1行ずつ値をカンマ区切りで代入していく
for(int y=0; y<lights.length; y++){
    // str()で文字列に変換して、「,」で4つの値をつなぐ
    // 4つだけなので直指定してますが、forで繰り返してもOK
    saveLines[ ? ] = str( ??? ) + ","
                    + str( ??? ) + ","
                    + str( ??? ) + ","
                    + str( ??? );
}
// saveStringsでsaveLinesを1行ずつstatus.txtに保存します
saveStrings("status.txt", saveLines);
```

# strでくっつけて保存



```
void mousePressed() {  
  for(int x=0; x<4; x++){  
    for(int y=0; y<3; y++){  
      if(dist(x*100+50,y*100+50,mouseX,mouseY)<=50){  
        lights[x][y] = 1 - lights[x][y];  
      }  
    }  
  }  
}
```

// 保存するのは3行

```
String[] lines = new String[3];
```

```
for(int y=0; y<3; y++){  
  lines[y] = str(lights[0][y]) + "," + str(lights[1][y]) + ","  
            + str(lights[2][y]) + "," + str(lights[3][y]);  
}
```

```
saveStrinsg("status.txt", lines);
```

```
}
```

```
1,0,1,1  
0,1,0,1  
0,0,1,1
```

# タブ区切りの取得



```
String[] 文字列配列 = split(文字列, '区切り文字');
```

```
// linesという配列に1行ずつ文字列を読み込む
String[] lines = loadStrings( "status.txt" );

// lines.lengthですべての行数を取得できる
for(int y=0; y<lines.length; y++ ){
    // line[i] ( i行目 ) の文字列をタブで分割
    String[] data = split( lines[y], '¥t' );
    // 0番目、1番目、2番目の値を整数に変換して代入
    lights[ ? ][ ? ] = int( data[0] );
    lights[ ? ][ ? ] = int( data[1] );
    lights[ ? ][ ? ] = int( data[2] );
    lights[ ? ][ ? ] = int( data[3] );
}
```

Macの場合は ¥  
ではなく \

# splitで分割して読み込む



loadStringsしてタブで分割して読み込み！

```
int[][] lights = new int[4][3];

void setup() {
  size(400, 300);

  // linesは3行入っている
  String[] lines = loadStrings("status.txt");
  for(int y=0; y<3; y++){
    // 「タブ」で分割すると4つに分けることができる
    String[] data = split(lines[y], "\t");

    // 整数に変換して代入しよう！
    lights[0][y] = int(data[0]);
    lights[1][y] = int(data[1]);
    lights[2][y] = int(data[2]);
    lights[3][y] = int(data[3]);
  }
}
```

1	0	1	1
0	1	0	1
0	0	1	1

# splitで分割して読み込む



loadStringsしてタブで分割して読み込み！

```
int[][] lights = new int[4][3];

void setup() {
  size(400, 300);

  // linesは3行入っている
  String[] lines = loadStrings("status.txt");
  for(int y=0; y<lines.length; y++){
    // 「タブ」で分割すると4つに分けることができる
    String[] data = split(lines[y], "\t");
    for(int x=0; x<data.length; x++){
      // 整数に変換して代入しよう！
      lights[x][y] = int(data[x]);
    }
  }
}
```

1	0	1	1
0	1	0	1
0	0	1	1

# タブ区切りで保存！



str()で文字列に変換してタブでつなぐ

```
// 3行分なので3つの要素からなる文字列配列を作る
String[] saveLines = new String[3]
// 1行ずつ値をカンマ区切りで代入していく
for(int y=0; y<lights.length; y++){
    // str()で文字列に変換して、「タブ」で4つの値をつなぐ
    // 4つだけなので直指定してますが、forで繰り返してもOK
    saveLines[ ? ] = str( ??? ) + "¥t"
                    + str( ??? ) + "¥t"
                    + str( ??? ) + "¥t"
                    + str( ??? );
}
// saveStringsでsaveLinesを1行ずつstatus.txtに保存します
saveStrings("status.txt", saveLines);
```

# strでくっつけて保存



```
void mousePressed() {  
  for(int x=0; x<4; x++){  
    for(int y=0; y<3; y++){  
      if(dist(x*100+50,y*100+50,mouseX,mouseY)<=50){  
        lights[x][y] = 1 - lights[x][y];  
      }  
    }  
  }  
}
```

// 保存するのは3行

```
String[] lines = new String[3];
```

```
for(int y=0; y<3; y++){  
  lines[y] = str(lights[0][y]) + "¥t" + str(lights[1][y]) + "¥t"  
            + str(lights[2][y]) + "¥t" + str(lights[3][y]);  
}
```

```
saveStrinsg("status.txt", lines);
```

```
}
```

1	0	1	1
0	1	0	1
0	0	1	1

# ネットからデータを読み込



- ProcessingのloadStringsは便利なことにネット上の情報を読み込むこともできる！
- こんな感じで書くだけ！ 超楽！

```
String[] lines = loadStrings("URL");
```

```
String[] lines = loadStrings("https://lecture.nkmr.io/2023/tokyo_weather_data");
```

# ネットからロード



```
String[] lines = loadStrings("URL");
```

```
// 0から10なので11個の要素からなる配列を作る
```

```
int[] scores = new int[11];
```

```
void setup(){
```

```
    size( 400, 400 );
```

```
    textSize( 25 );
```

```
    for(int i=0; i<11; i++){
```

```
        scores[i] = 0;
```

```
    }
```

```
    String[] lines =
```

```
        loadStrings("https://lecture.nkmr.io/2023/tokyo_weather_data.csv");
```

```
    for(int i=0; i<lines.length; i++){
```

```
        scores[ int( lines[i] ) ]++;
```

```
    }
```

```
    for(int i=0; i<scores.length; i++){
```

```
        println(i, scores[i]);
```

```
    }
```

```
}
```