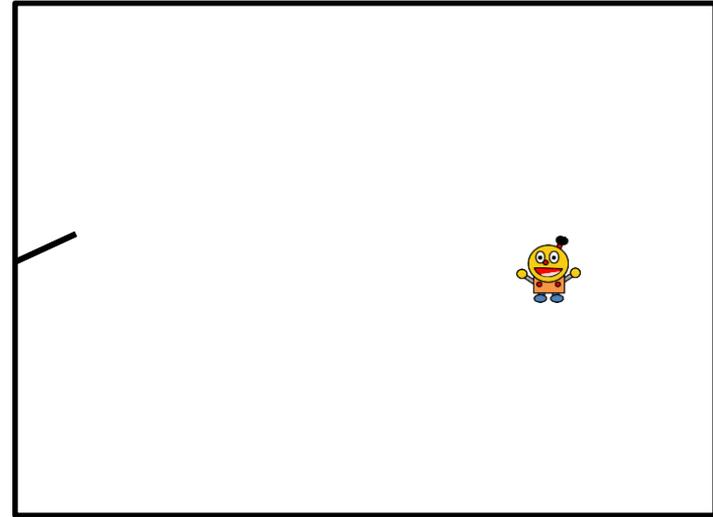
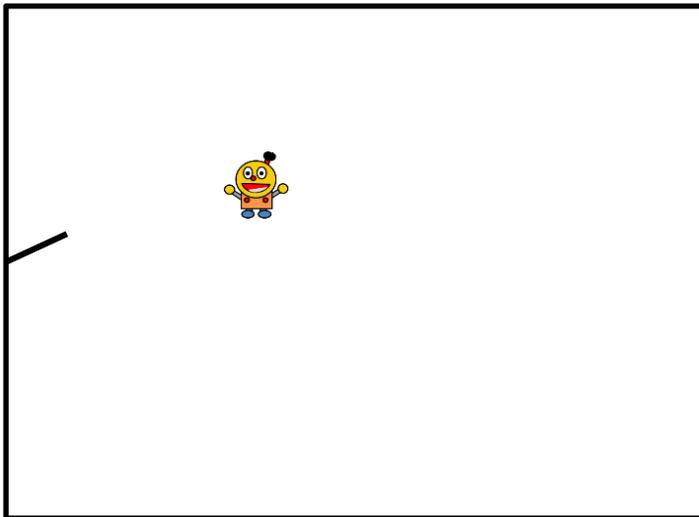


プログラミング演習I (第10回) 課題

• 基本① スケッチ名 : jumpChara

- キャラクタを描画する関数を作成し(参照: 参考資料), その関数を利用してキャラクタがジャンプする(斜方投射される)プログラムを作れ
- 600x400のウィンドウ左下からマウスカーソルがある方向にキャラクタを発射するジャンプ台のプログラムを作ってください。クリックされたタイミングでジャンプ(投射)されるようにすること。また、クリック時のジャンプ台からカーソルまでの距離を初速とせよ。
- ただし、発射位置については下から200ピクセルの場所とせよ
- (右端まで来ると跳ね返るようにしてもよい)



自分のキャラを表示する関数を作ろう

```
void drawCharacter(float x, float y, float angle_deg, float scale_factor ) {  
  
    int offset_x = 190;  ← offset_x と offset_y はキャラの中心座標。  
    int offset_y = 200;  ← 自分のキャラクタに合わせて値を変えること。  
  
    pushMatrix();  
    translate( x, y );  
    scale( scale_factor );  
    rotate( radians( angle_deg ) );  
    translate( -offset_x, -offset_y );
```

ここにchara1で作った、キャラクタを描画する処理をコピーしてください。
ただし、size()とbackground()は入れないこと。

```
    popMatrix();  
}
```

drawCharacter() の使い方

drawCharacter(X座標, Y座標, 回転角度, 拡大率);

回転角度の単位は【度】、拡大率は1.0で等倍。

```
void setup() {
  size( 600, 400 );
}

void draw() {
  background(255);

  // 例: マウスカーソルがある位置に45度傾けて0.5倍でキャラクタを描画
  drawCharacter( mouseX, mouseY, 45.0, 0.5 );
}

void drawCharacter(float x, float y, float angle_deg, float scale_factor ) {
  .....
}
```

プログラミング演習I (第10回) 課題

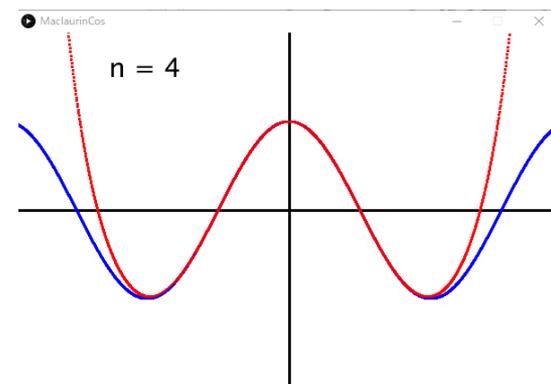
• 基本② スケッチ名 : MaclaurinCos

– $\cos x$ は下記のような式にマクローリン展開(テイラー展開の $a=0$ のもの)可能である。このマクローリン展開を行うための関数Maclaurinを完成させるうえで、累乗と階乗を求める関数が必要となった。累乗を計算するcalcPowerと階乗を計算するcalcFactorialを完成させ、マクローリン展開したものがグラフで一致することを確認せよ。

$$- \cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \frac{x^8}{8!} - \dots + (-1)^n \frac{x^{2n}}{(2n)!} + \dots$$

$$- f(x, n) = (-1)^n \frac{x^{2n}}{(2n)!} \text{ とすると}$$

$$- \text{Maclaurin}(x, n) = \sum_{i=0}^n f(x, i) \text{ となる}$$



プログラミング演習I (第10回) 課題

• 発展① スケッチ名 : DoReMi

- 配布したプログラムの幹音(ドレミファソラシド)となる周波数を求める関数 `getFrequency` を作成し、キーボード操作の上下によって音程を上下させるプログラムを完成させよ。
- まず準備段階として`minim`を環境に導入せよ(後述)
- 関数の引数は幹音のIDとし、返り値はその周波数の値(float)とせよ。
 - ド 261.6Hz
 - レ 293.7Hz
 - ミ 329.6Hz
 - ファ 349.2Hz
 - ソ 392.0 Hz
 - ラ 440.0
 - シ 493.9Hz
- 幹音のIDが0のときは261.6Hzのド、1のときは293.7Hzのレとなるようにすること。なお、1オクターブ上がるとそれぞれ周波数は2倍に、1オクターブ下がると周波数は1/2倍になる。
- 少なくとも3オクターブ分の結果を返すようにせよ。音はある程度聞こえていればOK。

プログラミング演習I (第10回) 課題

• 発展② スケッチ名 : lifegame_func

- 誕生、生存、過疎、過密によってセルが生まれたり死んだりするライフゲームを関数で作ろう。
- ライフゲームでは、対象とするセルの周囲8マスが生きているか死んでいるかを数え、その結果に応じてセルを生きている状態にするか、死んでいる状態にするかを切り替える。
- 100x100のマス目を用意し、セルが生きている場合は緑色の四角形を、死んでいる場合は黒色の四角形を描画するようにせよ。ライフゲームのルールは次ページで説明する。
- 配布する lifegame_func.pde の drawLifegame 関数と checkDeadOrAlive 関数、getCountOfAlive 関数を埋めてライフゲームを完成させよ
- 下記URLの安定状態が幾つか観測されたら成功
<http://ja.wikipedia.org/wiki/ライフゲーム>

プログラミング演習I (第10回) 課題

あるマス(赤フレーム)の縦・横・斜めの8マスの生死の状態(生の数)に注目する

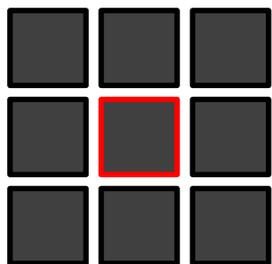
【誕生】 死んでいるセルに隣接する生きたセルがちょうど3つならば次世代が誕生

【生存】 生きているセルに隣接する生きたセルが2つか3つならば次世代でも生存

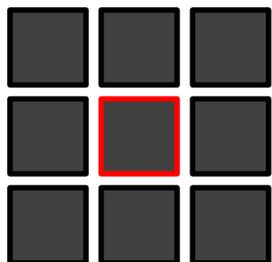
【過疎】 生きているセルに隣接する生きたセルが1つ以下ならば過疎により死滅

【過密】 生きているセルに隣接する生きたセルが4つ以上ならば過密により死滅

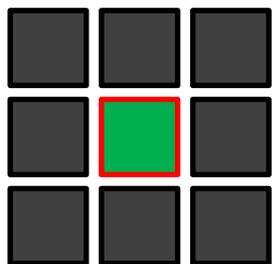
すべて死



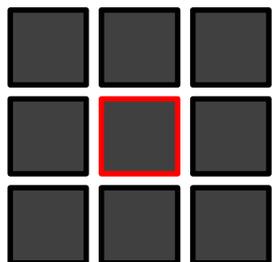
変化なし



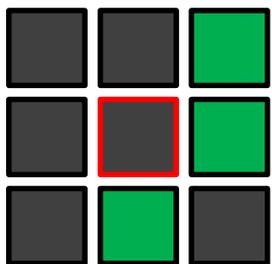
すべて死



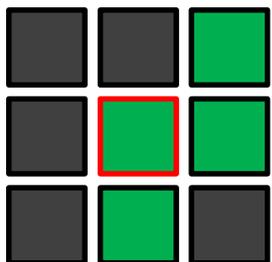
寂しくて死ぬ



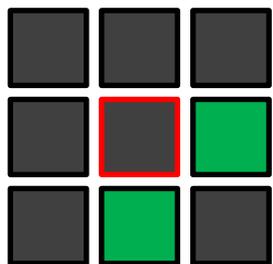
3つのマス



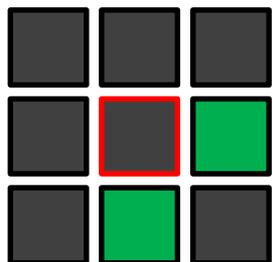
生まれる



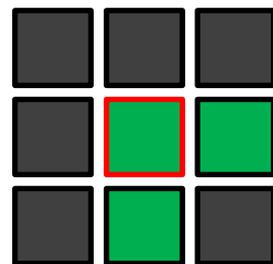
2つの生



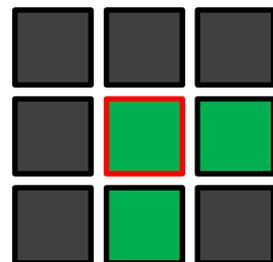
変化なし



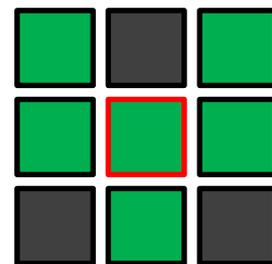
2つか3つの生



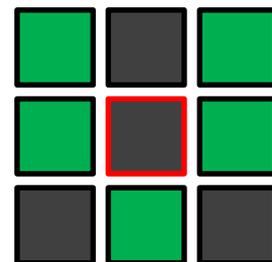
快適で変化なし



4つ以上の生



過密で死ぬ



プログラミング演習I (第10回) 課題

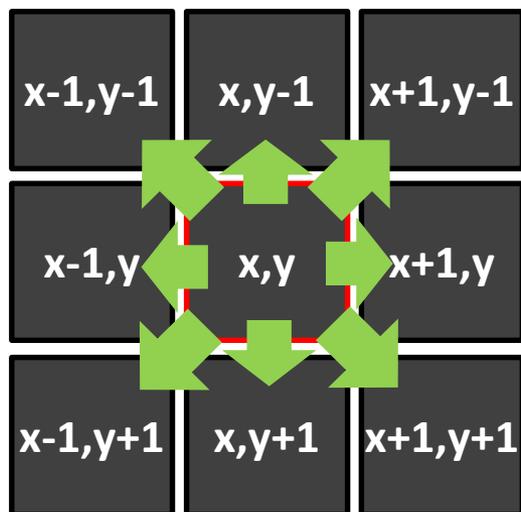
- 整理するとそのセルの周りの生きているセルの数が

3つなら生

2つなら維持

それ以外なら死

チェックする配列の添字は何になるか？



0から始めると
条件分岐が多くて面倒！

-1,-1	0,-1	1,-1
-1,0	0,0	1,0
-1,1	0,1	1,1

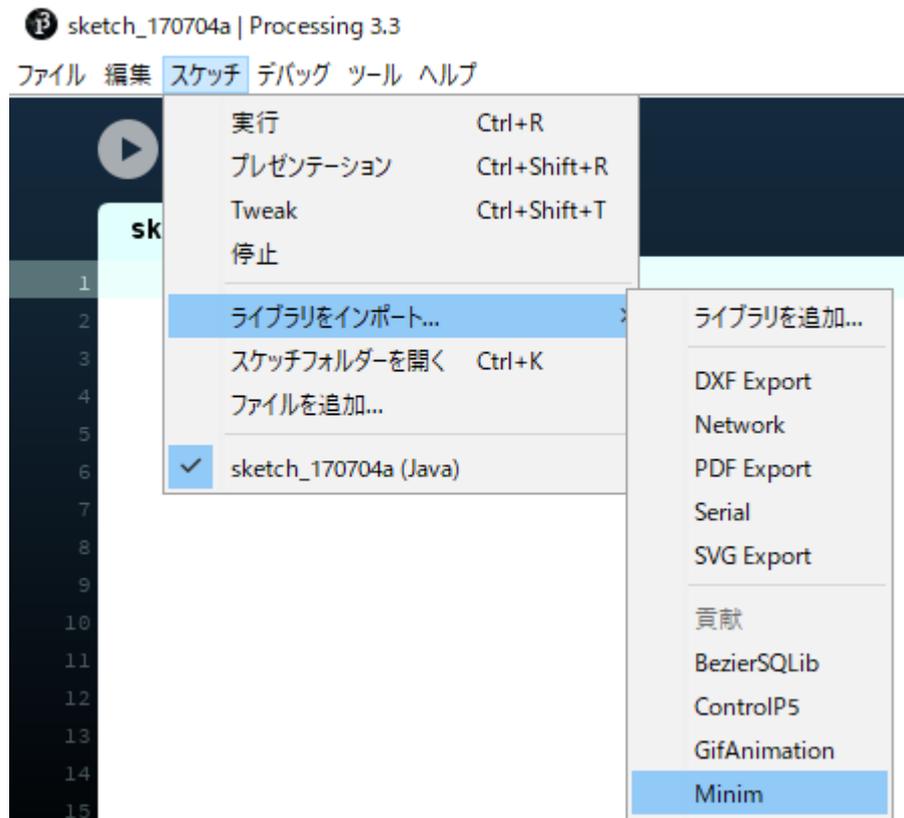
1から始めると
条件分岐が少なくなる

0,0	1,0	2,0
0,1	1,1	2,1
0,2	1,2	2,2

表示しない外周を用意して、周囲の「生」の数を数えると楽！

minimの利用方法

- スケッチ > ライブラリをインポート > Minim で利用できます



- ない自分の環境ではライブラリを追加しよう！

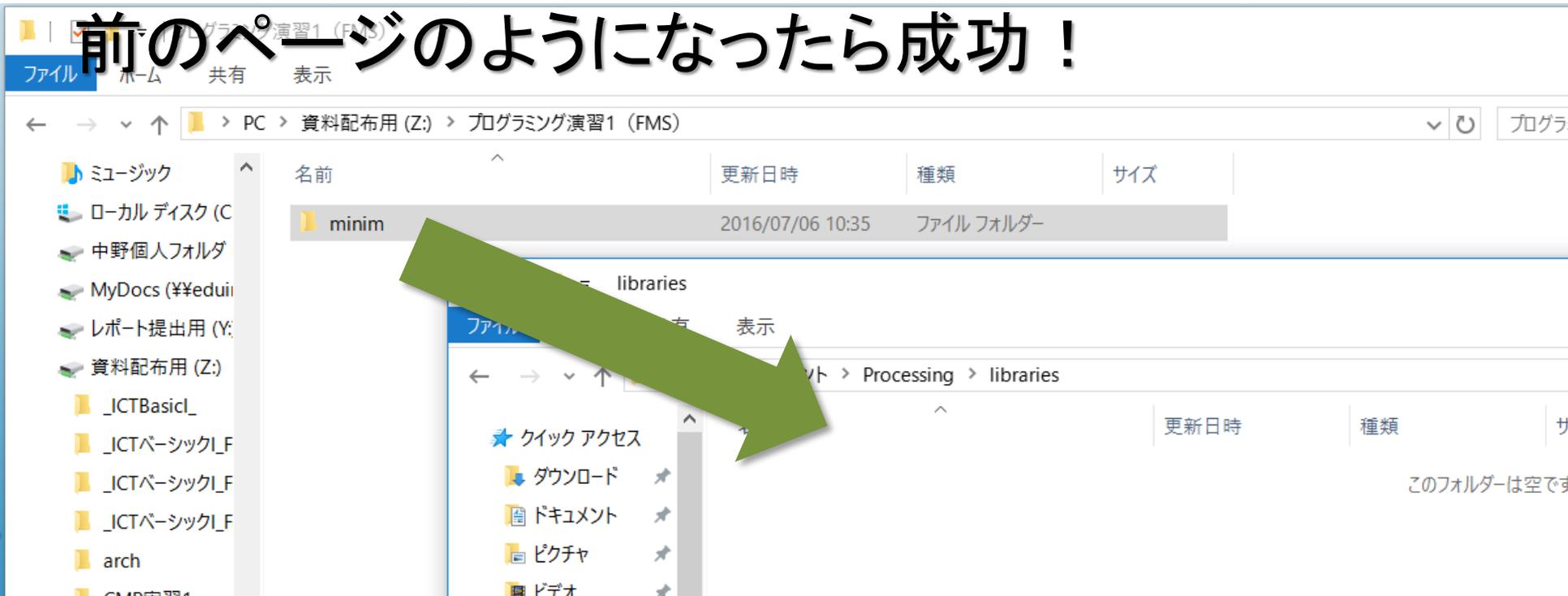
もしない場合は...

まずはProcessingを終了し、資料配布用(Zドライブ)のプログラミング演習1(FMS)フォルダから

Xドライブ¥ドキュメント¥Processing¥library

に**minim**というフォルダをコピーする！

前のページのようにになったら成功！



プログラミング演習発表会

- 7月19日 19:00-22:00(5F ホール)
- 進め方
 - 研究室内でリーダーを決め、できるだけリーダーのPCにプログラムを集めてプレゼンを行う
 - 必ず事前にノートPCで動作を確認しておくこと(本番で動かなかったら点数はありません)
 - その場で順序を決めその順序に応じて発表を行うこと
 - 発表時間は【**90秒厳守**】
- 提出締め切り
 - プログラムはいつもの定められたフォルダに【**21日の15時まで**】に提出すること.
 - 提出が確認できない場合は提出点がつきません

レギュレーション

- Processingによる、「スポーツまたはスポーツにまつわるモノを模倣せよ(未来のスポーツでも良い)」
 - 何かしらの動き(アニメーション)があること
 - マウスまたはキーボードの入力に対して、何らかの反応(動きなど)をする事
 - 3Dでなくてよい
- 組・番号, 名前, 何の模倣か, こだわりポイントは何かを発表すること
- 発表時間は1人90秒(厳守 & 強制終了)
- 起動 + 自己紹介があるので見せる時間は60秒程度!
- 必ず研究室単位の発表用PCで動作確認をし、発表練習をしておく。バックアップPCは用意しておくべし!