社会と情報
プログラミング超入門①

プログラミング

- 小学校からプログラミング教育が必修化された
- プログラミングを学ぶ意味とは？

プログラミング的思考

- プログラミングを学ぶことで「プログラミング的思考」を身に付けたい
  - 問題を解決するためには必要な手順（アルゴリズム）がある
  - そのアルゴリズムは何をどういう順番で組み合わせれば実現できるか？ を論理的に考えられる力
    ⇒「**プログラミング的思考**」

アルゴリズムの３構造

順次（順接）

・Ａをしたら次にＢをする

分岐

・もし〇〇ならＡをする（そうでなければＢをする）

反復（繰り返し、ループ）

・〇〇である限りずっとAを繰り返す

すべてこの３つで
表現できる

アルゴリズムの例

- カップ麺を作る方法（アルゴリズム）を考えよう

順次（順接）

・Ａをしたら次にＢをする

分岐

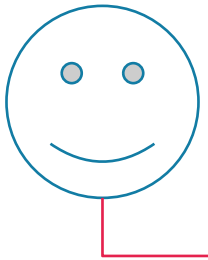
・もし〇〇ならＡをする（そうでなければＢをする）

反復（繰り返し、ループ）

・〇〇である限りずっとAを繰り返す

プログラミングとは

- アルゴリズムをコンピュータが実行できる形にすること



```
if(a-c!=0){
  return (-b+d)/(a-c);
}
else if(-b+d==0){
  return "Indeterminate";
}
else{
  return "Unsolvable";
}
```

プログラミング言語
でコードを書く

```
01100011 01101111 01101110 01110011 01110100 00100000 01100111 01100011
01100100 00111101 00101000 01100001 00101100 01100010 00101001 00111101
00111110 01111011 00001010 00100000 00100000 01101100 01100101 01110100
00100000 01110010 00111101 01100001 00100101 01100010 00111011 00001010
00100000 00100000 01101001 01100110 00101000 01110010 00111101 00111101
00110000 00101001 00100000 01110010 01100101 01110100 01110101 01110010
01101110 00100000 01100010 00111011 00001010 00100000 00100000 01100101
01101100 01110011 01100101 00100000 01110010 01100101 01110100 01110101
01110010 01101110 00100000 01100111 01100011 01100100 00101000 01100010
00101100 01110010 00101001 00111011 00001010 01111101 00111011 00001010
```

機械語に変換する

実行する

この学校でのプログラミング教育

- キロボ/コロボ（１年・ＩＴ活用）
- Excel/Google Spreadsheet（５年・社会と情報）
  - 「それっぽいコードを書く」ことは未経験？

授業全体の目的

- アニメーションの作成を通し、実際にコードを書くことによるプログラミングを体験する
- プログラミング的思考の基礎を改めて身に付けていく

授業の流れ（全５回）

1. ・導入（ボールを表示させる）
2. ・変数（ボールを動かす）
3. ・分岐（ボールを跳ね返らせる）
4. ・作品制作①（アニメーションを再現する）
5. ・作品制作②（アニメーションをアレンジする）

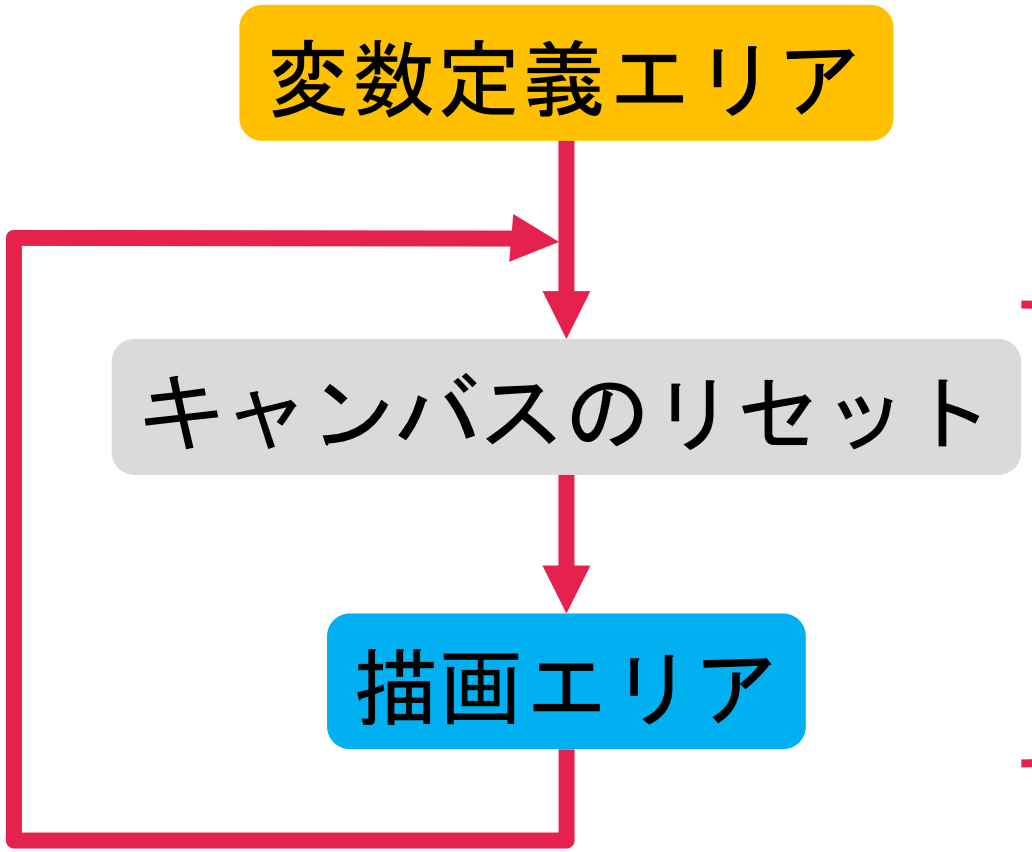
授業で使うシミュレータの紹介

- https://mathinfolec.github.io/canvas/
  - Canvasシミュレータ
  - 本来図形を描く前後に必要な処理を省略できるツール

授業内容の記録について

- 皆さんがシミュレータを使っている状況を記録し、大学での実験にデータを利用する予定です
- 個人が特定されることはありません
- 成績には影響しません
- もしデータを使われたくない場合は教えてください

シミュレータの仕組み



必要な変数（データの入った箱）
を定義する

（勝手にリセットされる）

1フレームごとの画面を描く
⇒コマ撮りアニメのようになる

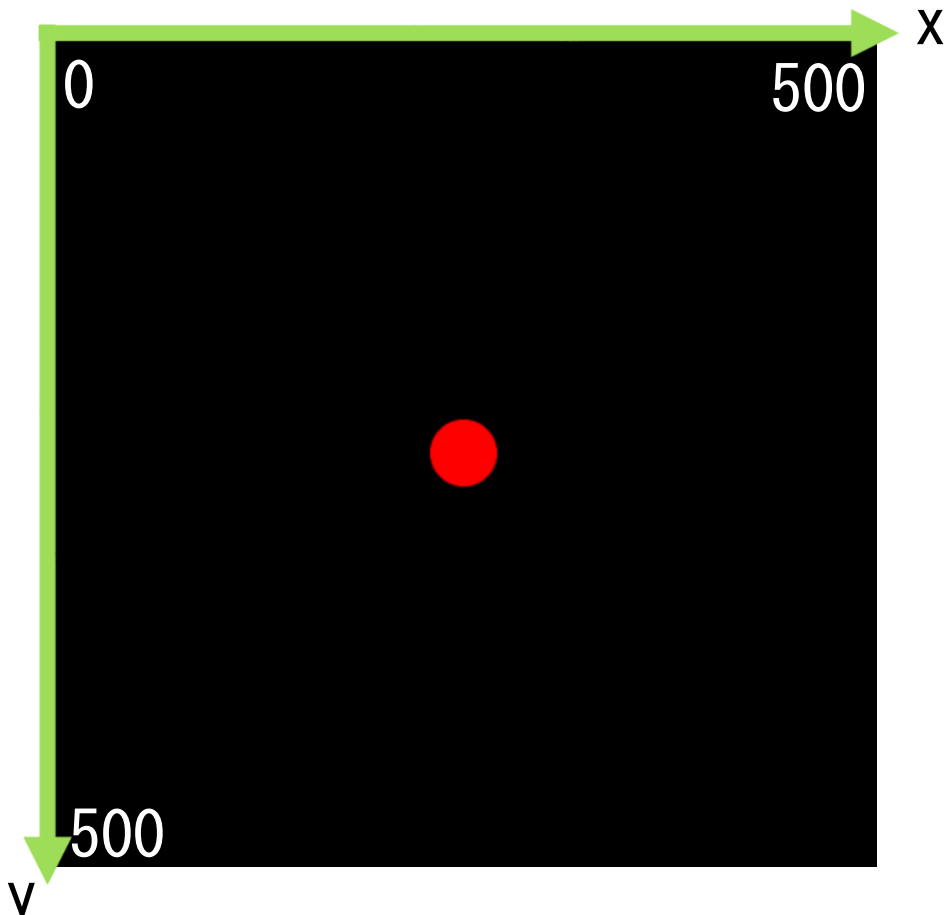
円の表示

これ以降に描く図形の色を赤色に設定したのち、(250,250)を中心に半径20の円を描く
（この処理を毎フレーム行う）

描画エリア

```
setColor("red");
drawCircle(250,250,20);
```

すべて半角で記入する
大文字・小文字は区別する
セミコロンを末尾に必ず付ける

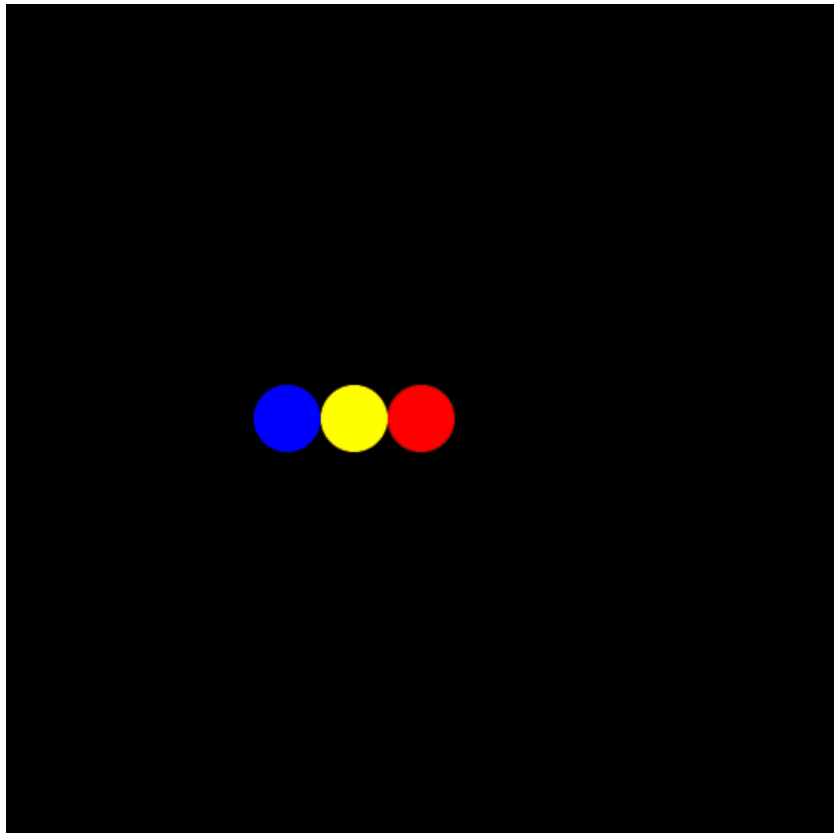


複数の円の表示

- 同じように書いていく

描画エリア

```
setColor("red");
drawCircle(250,250,20);
setColor("yellow");
drawCircle(210,250,20);
setColor("blue");
drawCircle(170,250,20);
```

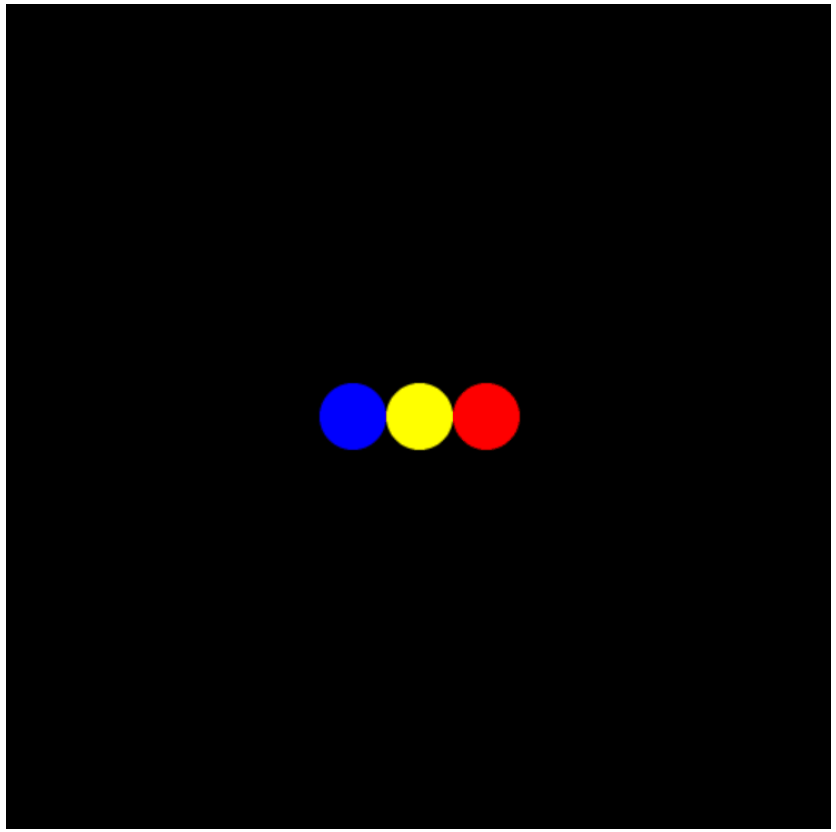


円の位置を修正したい場合

- 修正箇所が多くなると管理が難しい

描画エリア

```
setColor("red");
drawCircle(250290, 250, 20);
setColor("yellow");
drawCircle(210250, 250, 20);
setColor("blue");
drawCircle(170210, 250, 20);
```



変数の導入

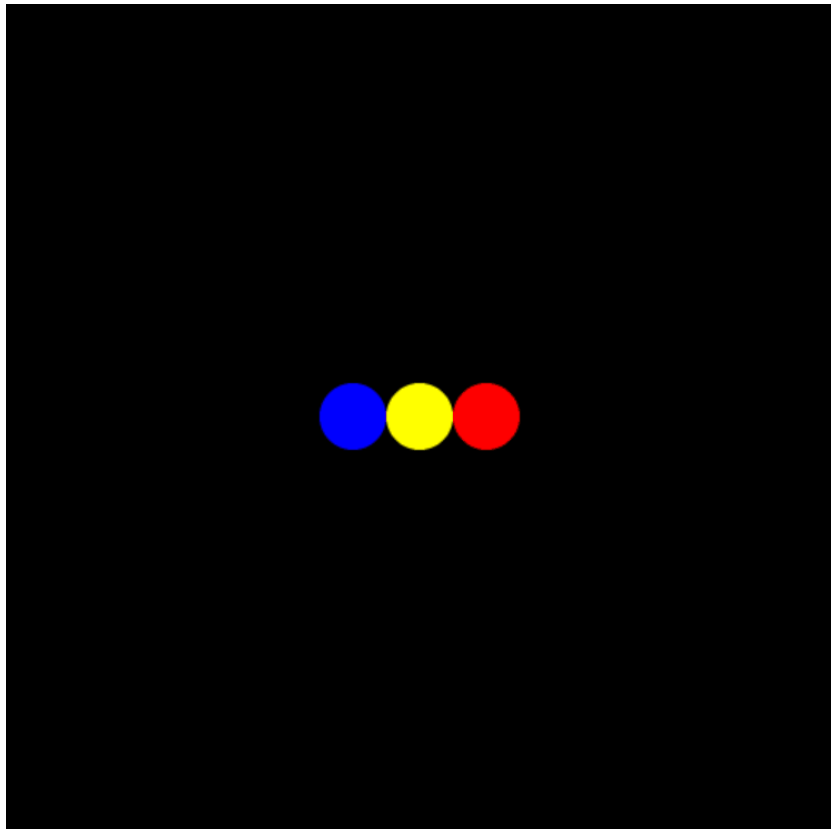
変数定義エリア

```
let x=250;
let y=250;
let r=20;
```

変数(データを入れる箱)の中身だけ変更すれば全体をずらせる

描画エリア

```
setColor("red");
drawCircle(x+40,y,r);
setColor("yellow");
drawCircle(x,y,r);
setColor("blue");
drawCircle(x-40,y,r);
```



（補足）算術演算子

算術	表記
足し算	a + b
引き算	a - b
掛け算	a * b
割り算	a / b
剰余算	a % b
べき乗	a ** b

変数定義エリア

```
let x=250;
let y=250;
let r=20;
```

描画エリア

```
setColor("red");
drawCircle(x+40,y,r);
setColor("yellow");
drawCircle(x,y,r);
setColor("blue");
drawCircle(x-40,y,r);
```

演習

1. 左のコードを写し、変数の使い方と描画方法を理解する
2. 変数x,y,rの中身を変えて動作を確認する
3. 円の大きさを変えても円同士が接するようにする
4. 【応用】円を斜めに配置する