



社会と情報 プログラミング超入門③

前回の復習

- 変数を使うことでアニメーションを表現できる

変数定義エリア

```
let x=250;  
let vx=5;  
let y=250;  
let vy=10;  
let r=20;
```

描画エリア

```
x=x+vx;  
y=y+vy;  
setColor("red");  
drawCircle(x, y, r);
```

変数定義エリア

```
let x=250;  
let vx=5;  
let y=250;  
let r=20;
```

キャンバスのリセット

描画エリア

```
x=x+vx;  
setColor("red");  
drawCircle(x, y, r);
```

前回の復習

1フレーム目

(255,250)



xの値を $x+vx$ にし (xを vx だけ増やし)、
(x, y) を中心に半径 r の円を描く

サンプル2 (第2回)

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)

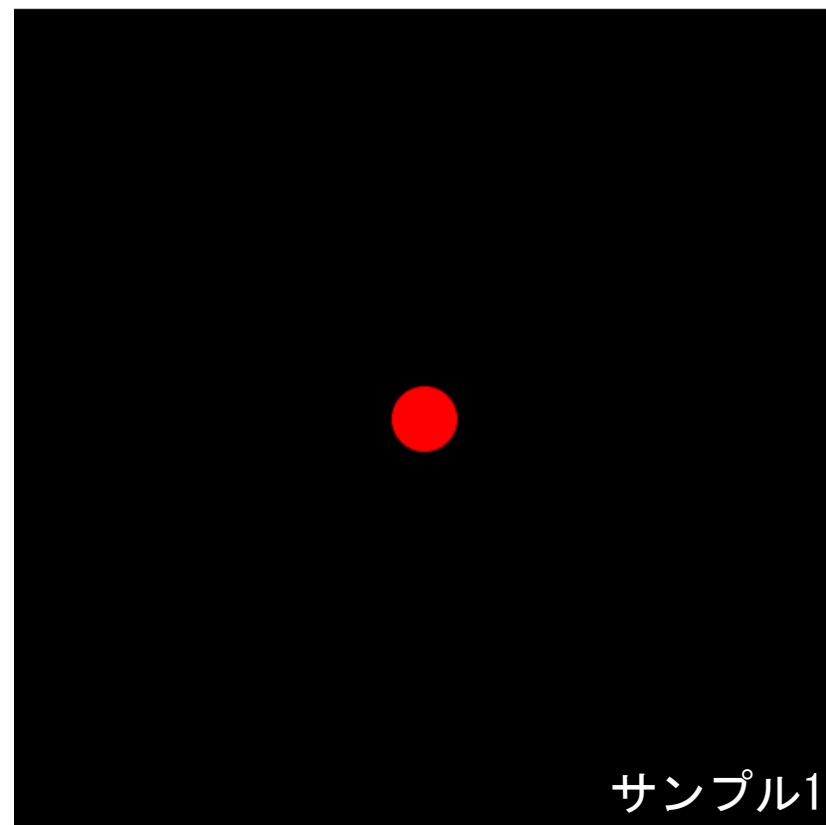
反復(繰り返し、ループ)

- ・ 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

今回の目標

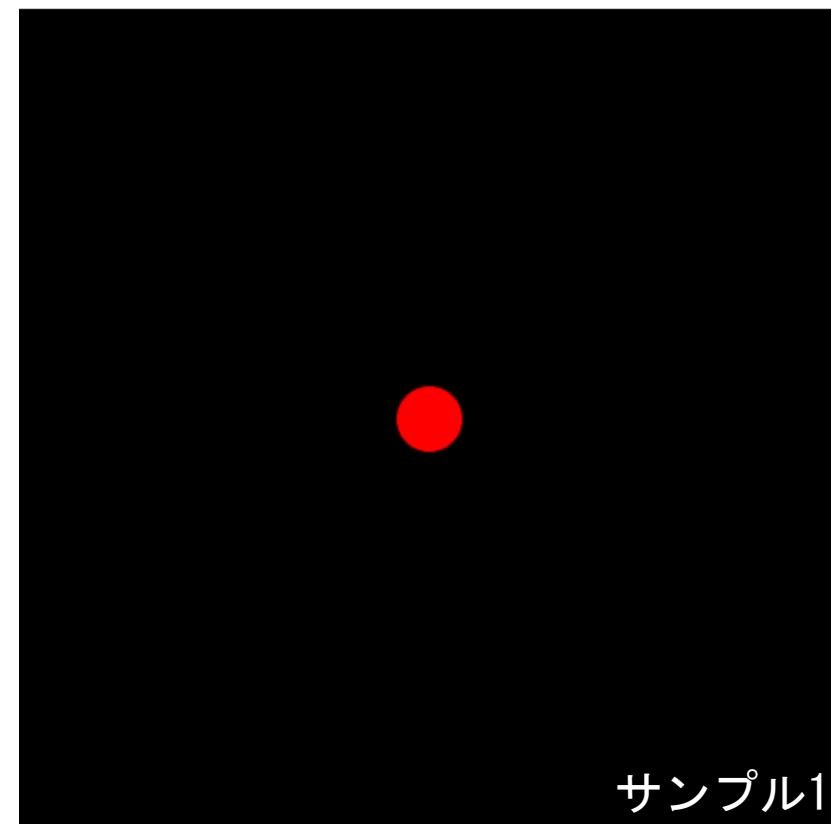
- 分岐を使い、図形が壁で跳ね返るアニメーションを作れるようになる



サンプル1

分岐

- 「もし〇〇なら A をする（そうでなければ B をする）」というもの
- 今回は「もし壁に当たっていたら跳ね返らせる（そうでなければ何もしない）」という処理を実現させる



サンプル1

分岐の表現

```
if(条件式1) {  
    条件式1が成り立っているときの処理;  
}  
else if(条件式2) {  
    条件式1は成り立たず条件式2は成り立っているときの処理;  
}  
else {  
    条件式も条件式2も成り立っていないときの処理;  
}
```

条件式（比較演算子）

比較演算	表記
aとbは等しいか	<code>a == b</code>
aとbは等しくないか	<code>a != b</code>
aはb以上か	<code>a >= b</code>
aはbより大きい	<code>a > b</code>
aはb以下か	<code>a <= b</code>
aはbより小さい	<code>a < b</code>

条件式（論理演算子）

論理演算	表記
AかつB	A && B
AまたはB	A B
Aではない	!(A)

壁での跳ね返し（準備）

変数定義エリア

```
let x=250;  
let y=250;  
let vx=5;  
let vy=10;  
let r=20;
```

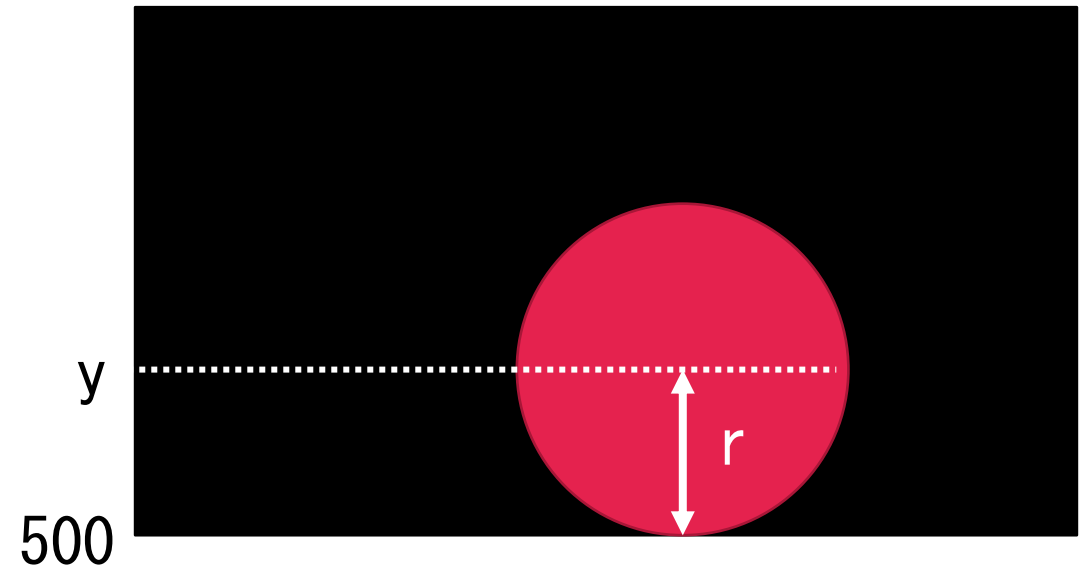
描画エリア

```
x=x+vx;  
y=y+vy;  
setColor("red");  
drawCircle(x, y, r);
```

下の壁に当たっているかの判定

- y が $500-r$ 以上ならば当たっている

```
if ( $y \geq 500 - r$ ) {  
    当たっているときの処理;  
}
```

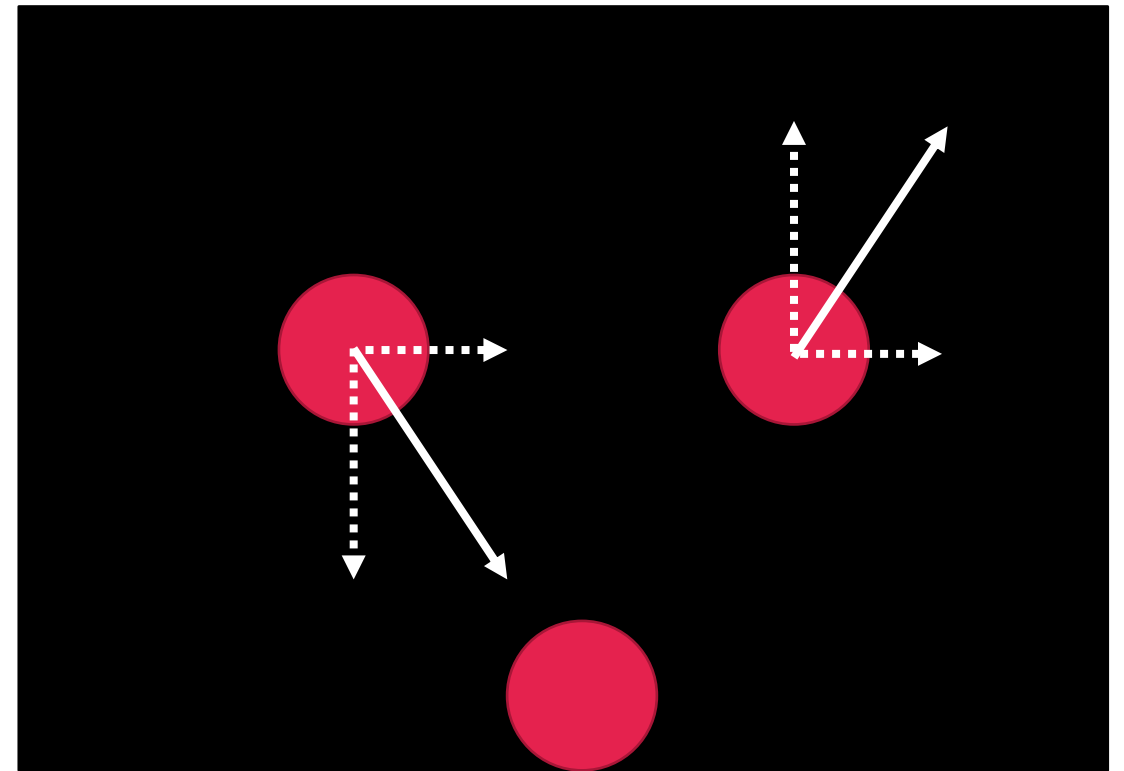


上図の位置かそれより下なら (y 座標がこれ以上なら) 当たっている

跳ね返ったときの動き

- y方向の速度が逆転する

```
if (y >= 500 - r) {  
    vy = -1 * vy;  
}
```



x方向とy方向に分解して考える

跳ね返りの実装（下の壁のみ）

変数定義エリア

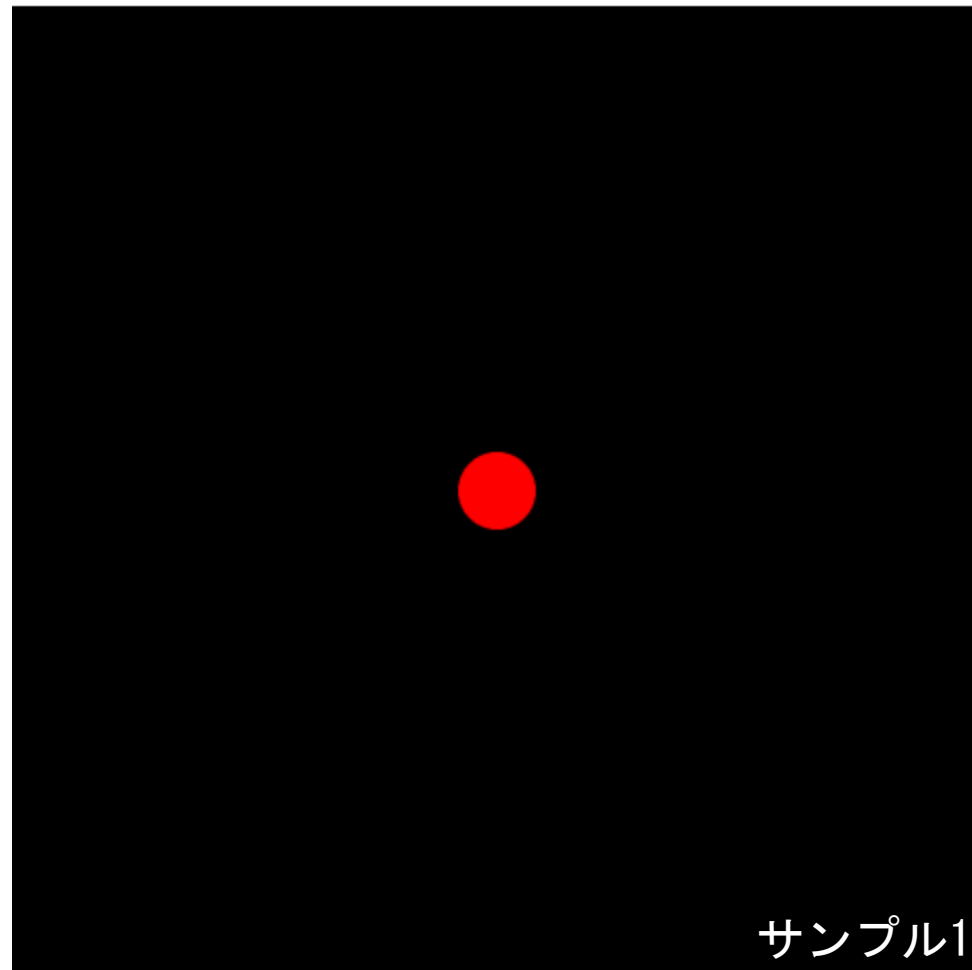
```
let x=250;  
let y=250;  
let vx=5;  
let vy=10;  
let r=20;
```

描画エリア

```
x=x+vx;  
y=y+vy;  
if (y>=500-r) {  
    vy=-1*vy;  
}  
setColor("red");  
drawCircle(x, y, r);
```

動作の結果

- 下の壁のみ実装したので
こういった結果になる



サンプル1

変数定義エリア

```
let x=250;  
let y=250;  
let vx=5;  
let vy=10;  
let r=20;
```

描画エリア

```
x=x+vx;  
y=y+vy;  
if (y>=500-r) {  
    vy=-1*vy;  
}  
setColor("red");  
drawCircle(x, y, r);
```

演習

1. 左のコードを写し、分岐の仕組みと動作について理解する
2. 変数の値などを自由に変えてみる
3. 他の方向で跳ね返る際の処理を作る
4. 【応用】跳ね返った回数を変数countに入れ、`display(count);`を描画エリアの末尾に追記して画面に表示させる