

応用的な話題



PROCESSING FRIENDS



物体の移動

内分の公式を利用する

速度を利用して
位置を決める

関数を使って、
直接位置を決める

加速度を利用して、
速度を決めてから
位置を決める

内分の公式を利用した移動

$$\begin{pmatrix} x \\ y \end{pmatrix} = (1 - t) \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} + t \begin{pmatrix} x_1 \\ y_1 \end{pmatrix}$$



良く利用されるので、`lerp(start,end,t)`という関数が用意されている。

tの値の作り出し方の例

	$0 \leq t < 1$	$0 \leq t \leq 1$
変数frameCountを利用	$(\text{frameCount} \% 30) / 30.0$	$(\text{frameCount} \% 30) / (30.0 - 1.0)$
関数millisを利用	$(\text{millis}() \% 1000) / 1000.0$	$(\text{millis}() \% 1000) / (1000.0 - 1.0)$

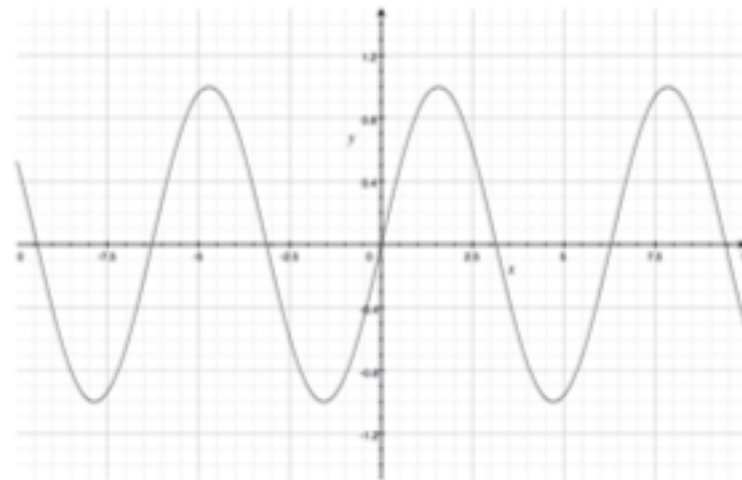
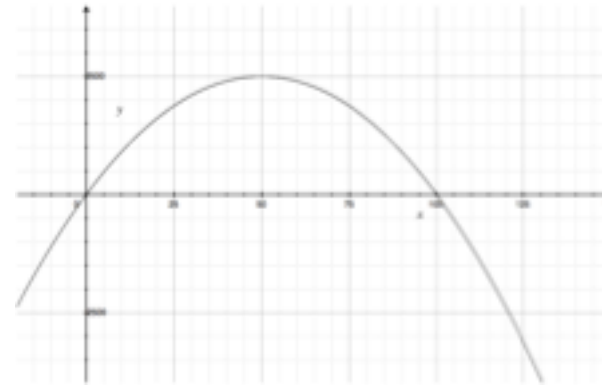
1秒間の描画回数 (FPS,frames per second) : 30回がデフォルト
システム変数frameCountが何回目かの描画を記憶している
1秒間の描画回数を変更 : frameCount(描画回数)

直接関数を利用

例えば、

$$y = -x^2 + ax$$

$$\begin{cases} x = at \\ y = b \sin t \end{cases}$$



速度と加速度

$$\text{速度} = \frac{\text{移動距離}}{\text{時間}}$$

$$v(t) = \lim_{\Delta t \rightarrow 0} \frac{x(t + \Delta t) - x(t)}{\Delta t}$$

$$a(t) = \lim_{\Delta t \rightarrow 0} \frac{v(t + \Delta t) - v(t)}{\Delta t}$$

$$x(t + \Delta t) \approx x(t) + v(t)\Delta t$$

$$x(t + \Delta t) \approx x(t) + v(t) \Delta t = 1 \text{ とすると}$$

$$v(t + \Delta t) \approx v(t) + a(t)\Delta t$$

$$v(t + \Delta t) \approx v(t) + a(t) \Delta t = 1 \text{ とすると}$$

v:velocity, a:acceleration

物体の移動（速度を利用）

速度を求める

```
vx = -speed * sin(angle);  
vy = speed * cos(angle);
```

位置の座標に
速度の値を加える

```
xPos = xPos + vx;  
yPos = yPos + vy;
```

新しい位置が
わかる

atan2関数

atan2(Y軸方向の変化量,X軸方向の変化量) : 角度を求める

$P(x_0, y_0)$ と $Q(x_1, y_1)$

直線PQの角度 $\text{atan2}(y_0 - y_1, x_0 - x_1)$

物体の移動（加速度を利用）

加速度を求める

```
ax = 0;  
ay = -1.0;
```

速度に加速度の
値を加える

```
vx = vx+ax;  
vy = vy + ay;
```

速度に加速度の
値を加える

```
xPos = xPos + vx;  
yPos = yPos + vy;
```

新しい位置が
わかる

Processing言語には2次元や3次元の点の位置を表すデータ型がある

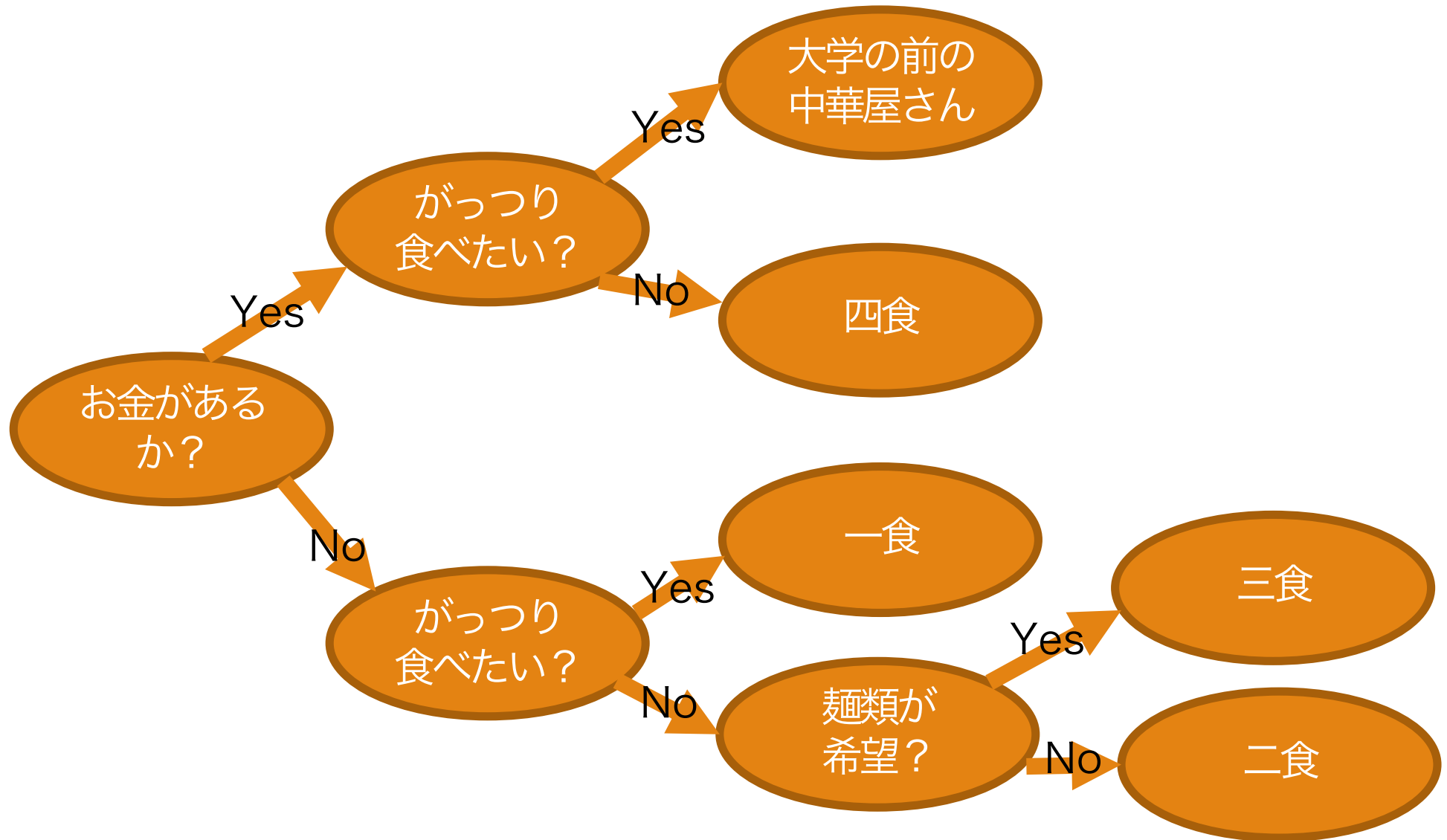
PVector型

```
PVector ball = new PVector();  
ball.x = width/2;  
ball.y = height/2;
```

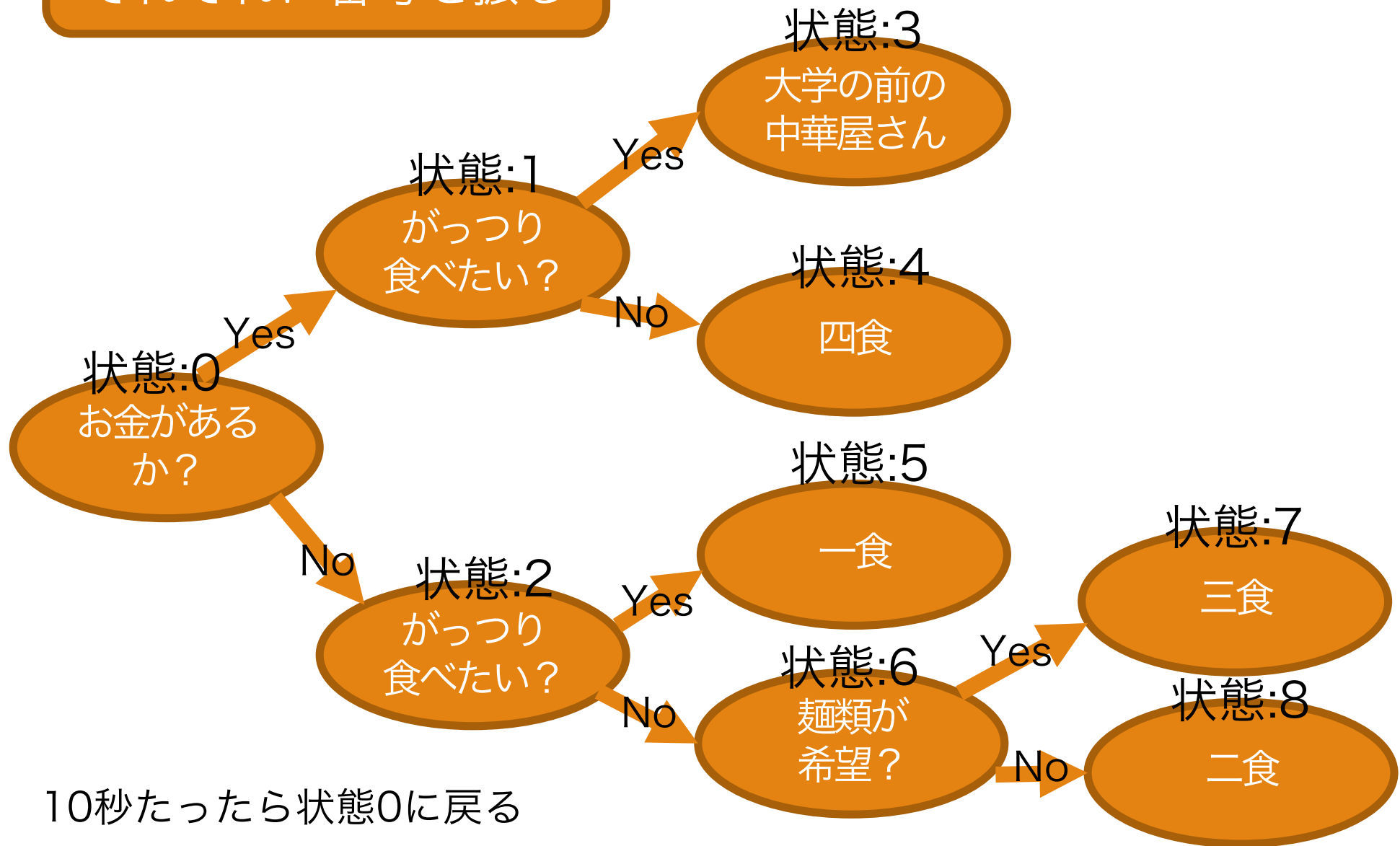
```
Pvector v = new PVector(cos(angle),sin(angle));  
v.mult(speed);
```

```
ball.add(v);
```

状態遷移図の例



状態遷移図の例
それぞれに番号を振る



10秒たったら状態0に戻る

プログラムのポイント

現在の状態を
int型変数stateにいれる

if文を組み合わせて、
現在の状態を調べる

状態iでの表示メッセージは
message[i]に入っている

switch~case文を使うと
もう少し簡潔に書ける

```
if(state==0){  
  // 何かをする0  
}else if(state==1){  
  // 何かをする1  
}else if(state == 2){  
  // 何かをする2  
}else if…{  
}else{  
  // 何かをする  
}
```

```
switch(state){  
  case 0:  
    // 何かをする0  
    break;  
  case 1:  
    // 何かをする1  
    break;  
  case 2:  
    // 何かをする2  
    break;  
  …  
  default:  
    // 何かをする0  
}
```

変数stateはint型かchar型とする

データ型の分類

基本型

変数名

値が入っている

```
int a;  
a = 1;
```

a

1

参照型

変数名

値がどこに入っているかを示している

```
PImage a;  
a = loadImage(...);
```

a

画像情報の
本体

参照型の単純な代入

```
PImage a,b;  
a = loadImage(...);  
b = a;
```



授業時に配布した資料

<http://www.sato-lab.jp/imfu/index.html>

においてあります。