



---

# プログラミング演習(11)

## 関数(メソッド)

---

中村, 高橋  
小林, 橋本

# 目標

---

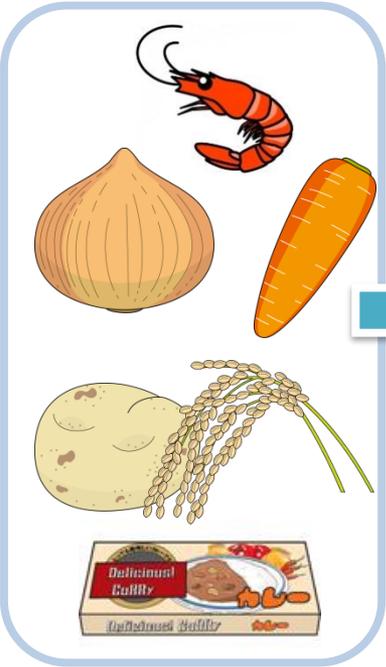


- 関数(メソッド)の理解を深める

# 関数(メソッド)とは



- 何かの処理をしてくれる「処理機械」みたいな物



なんか  
処理機



「なんか処理機」の中は  
良くわからないブラックボックス  
だけど使える！  
具材入れたらカレーができる！

# 例：関数(メソッド)



- 電子レンジ
  - 冷えたお弁当を入れてボタンを押すと、温まったお弁当が手に入る
- ポット
  - ボタンを押すとなんかお湯が出てくる
- 冷蔵庫
  - 食材を冷蔵庫に入れる(保存する)
- ストップウォッチ
  - ボタンを押すとラップタイプが記録されているらしい

# たとえば



- `background( r, g, b );`
- `ellipse( cx, cy, rx, ry );`
- `line( x1, y1, x2, y2 );`
- `dist( x1, y1, x2, y2 );`
- `pow( x, n );`
- `radians( theta );`
- `sin( radians( theta ) );`
- `atan2( mouseY-cy, mouseX-cx );`

とかいろいろありましたよね？

# 関数ってなんで必要なの？



- 同じ処理を何度も書いていませんか？
- drawがやたらめったら長くなっていませんか？
- どこからどこまでがちゃんと動くか分離できていないではないですか？
- 一度関数を作ったら使いまわせるよ！

# メソッドは4種類



何か入力して  
何か出力される

なんか  
処理機

何か入力されるが  
何も出力されない

なんか  
処理機

何も入力してないけど  
何か出力される

なんか  
処理機

何も入力してないし  
何も出力されない

なんか  
処理機

# わかりにくいけれど



- メソッドは何らかの出力をするのでは？
  - 何かを画面に表示する
  - 何か音を鳴らす
  - 何かをファイルに出力する
  - クラス内変数の値を変更する
  - グローバル変数の値を何かに変更する

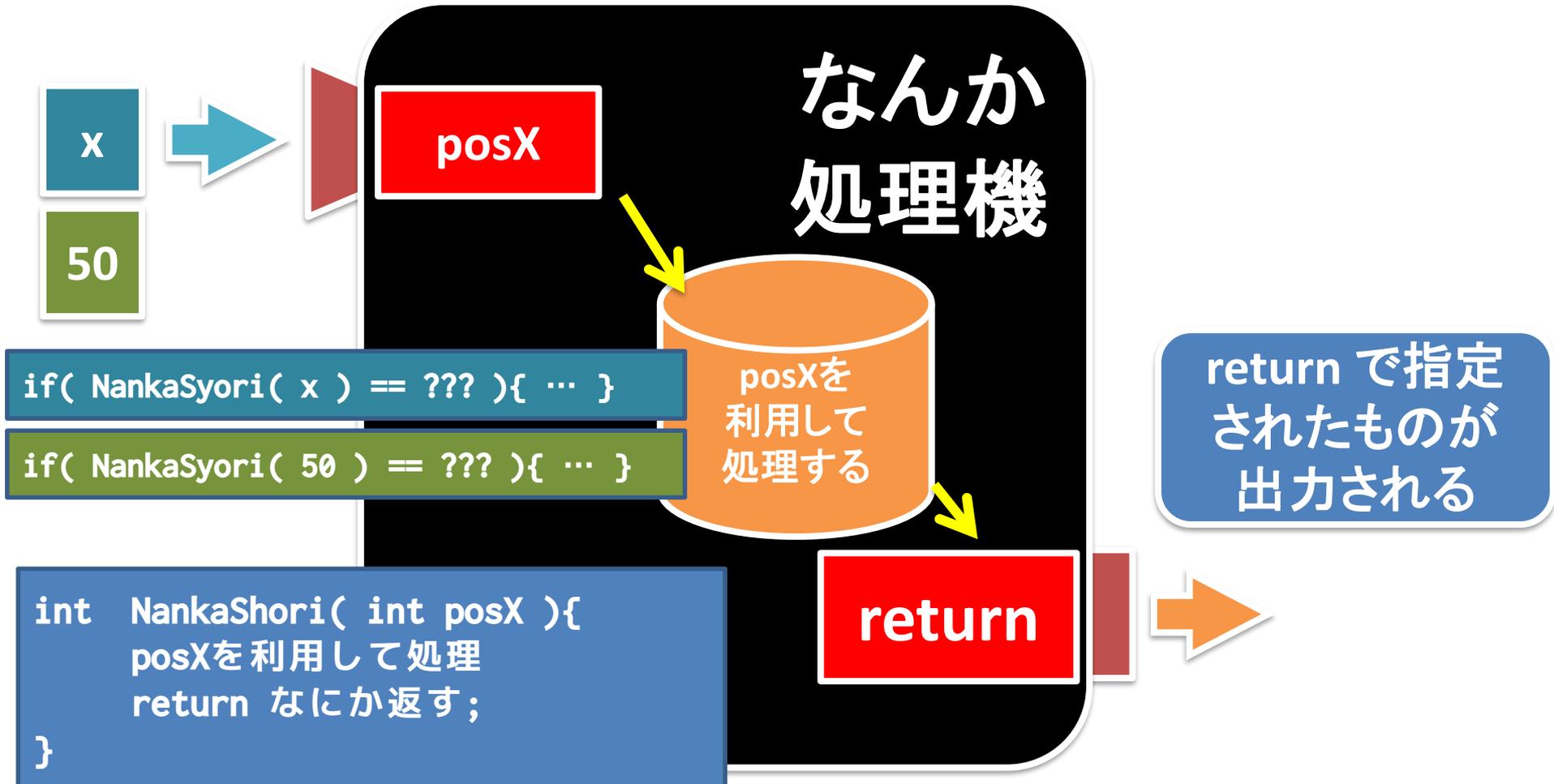
出力してるじゃん！！

- 上記はメソッドの明示的な出力ではなく副産物みたいなもの。メソッドとしての明示的な出力はある場合とない場合がある。

# メソッドの内部処理



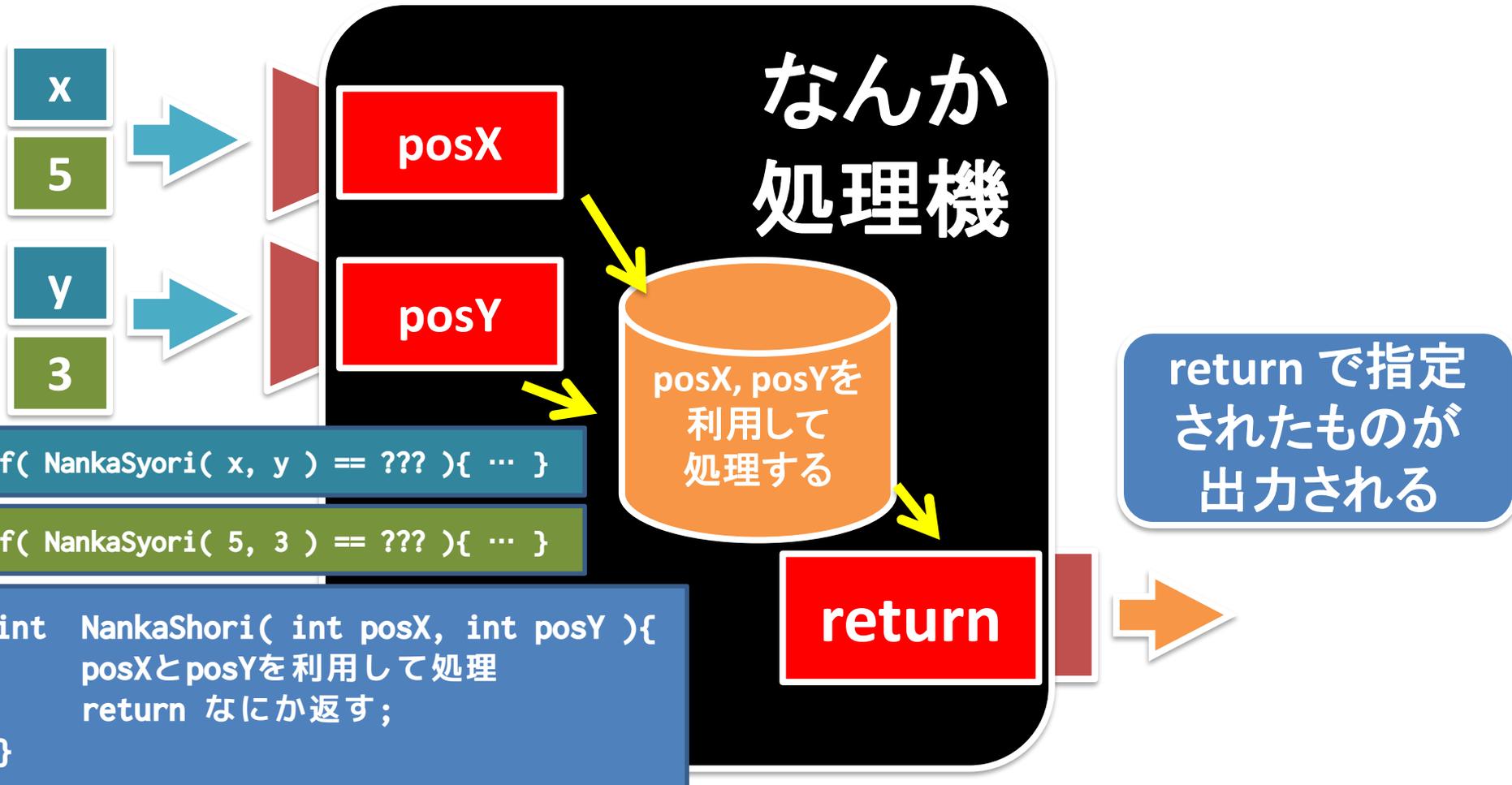
- 引数として取得した値は，引数で指定された変数名を利用して処理. returnで何か返される



# メソッドの内部処理



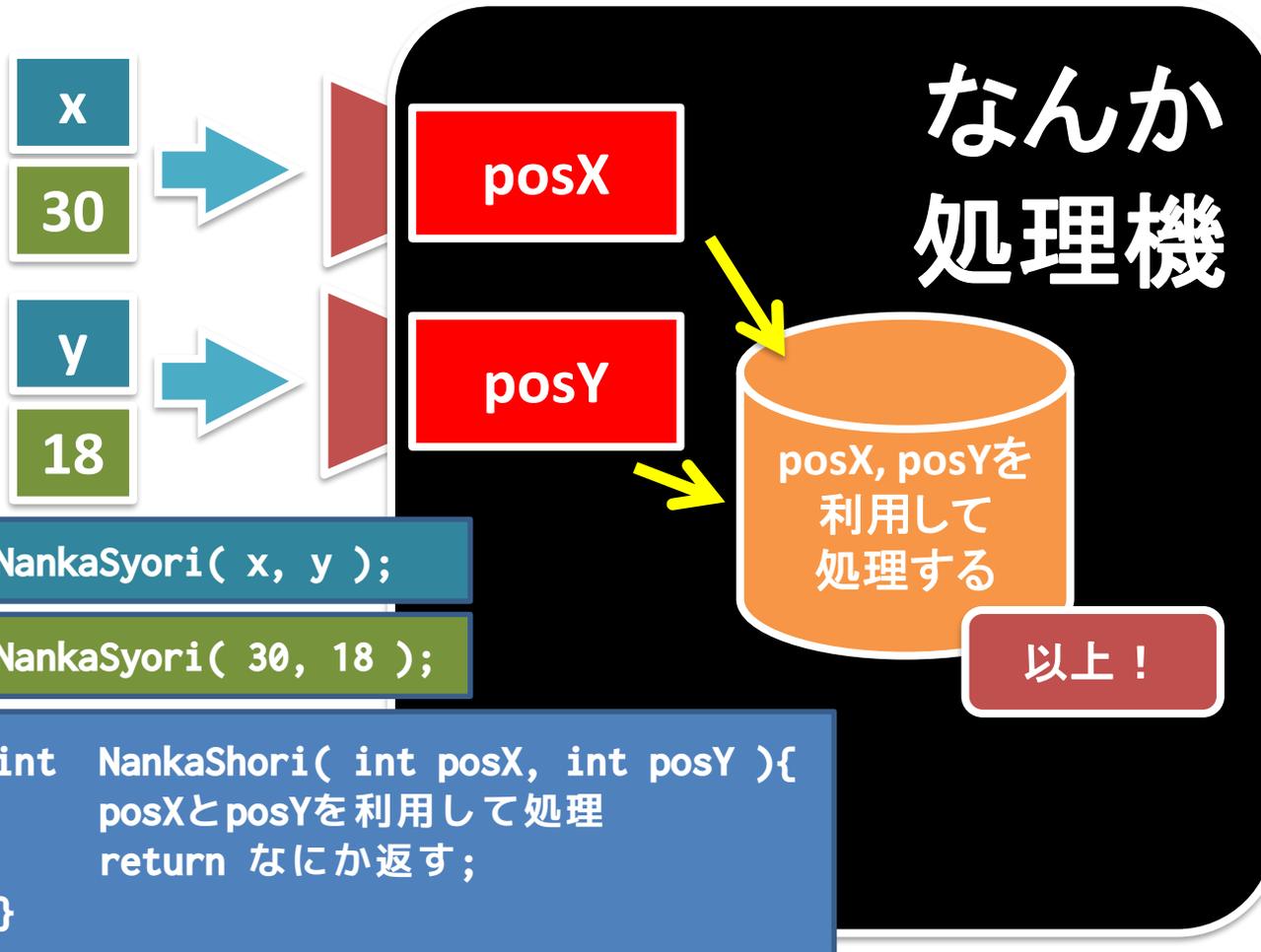
- 引数は増えるけれど返り値 (returnされる値) は増えない



# メソッドの内部処理



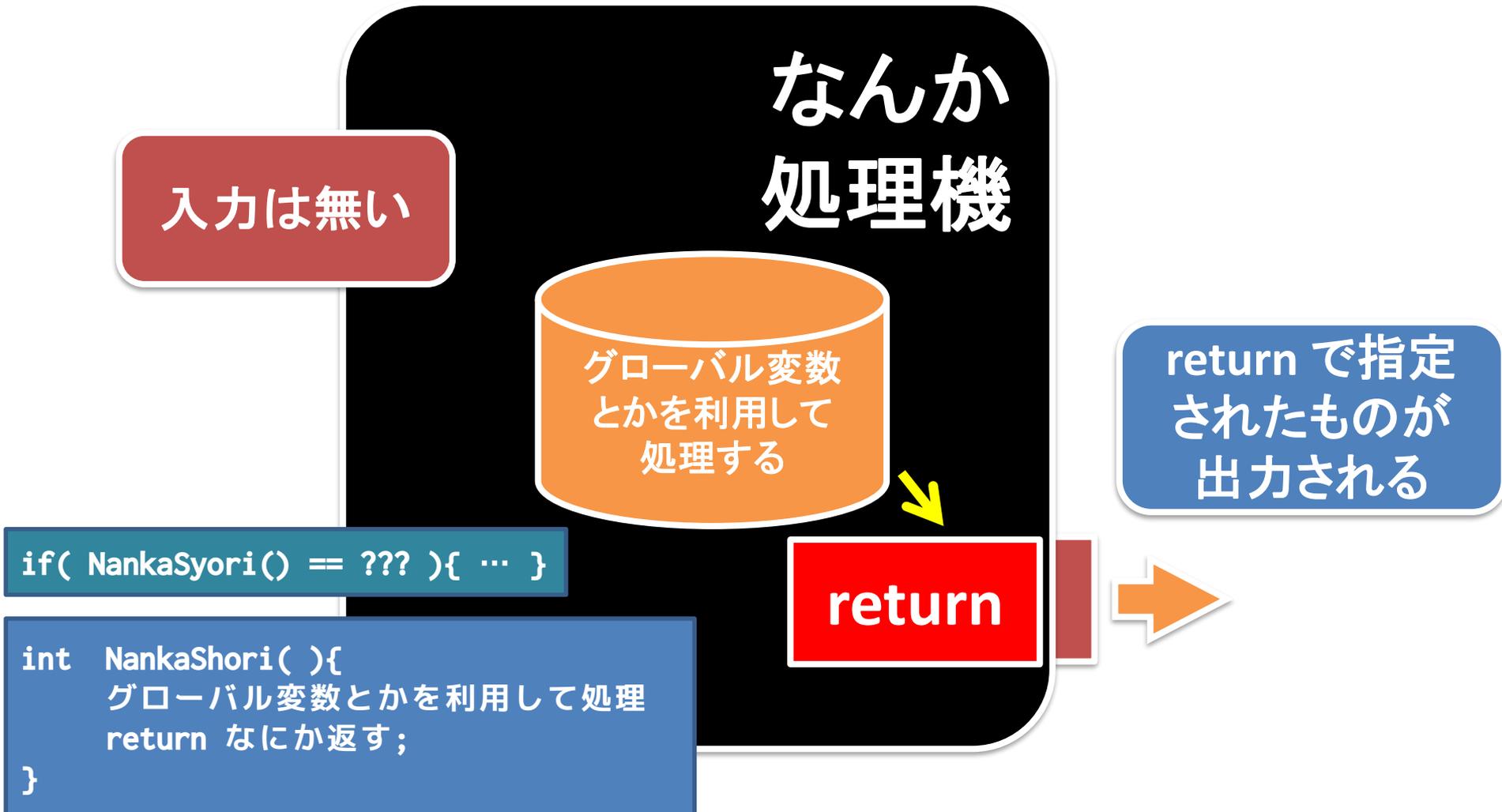
- 引数は増えるけれど返り値 (returnされる値) は増えない



# メソッドの内部処理



- 返り値だけのケース(引数がない場合)



# メソッドの内部処理



- 入力も出力もないメソッド

入力は無い

なんか  
処理機

グローバル変数とか  
を利用して処理する

```
NankaSyori();
```

以上！

```
void NankaShori( ){  
    なんかグローバル変数を利用して処理  
}
```

# もう一度整理



何か入力して  
何か出力される

なんか  
処理機

何か入力されるが  
何も出力されない

なんか  
処理機

何も入力してないけど  
何か出力される

なんか  
処理機

何も入力してないし  
何も出力されない

なんか  
処理機

# 引数〇, 返り値〇



(Q) ある入力された数字の約数の数を求める関数をどう作るか？ また, その関数を使って数字と約数の数のペアを出力しよう

```
17989: 2
17990: 16
17991: 6
17992: 16
17993: 4
17994: 8
17995: 8
17996: 12
17997: 8
17998: 4
17999: 4
18000: 60
```

# 約数の数を求める関数



## • 考え方

- 引数は整数型の `num` にする
- 約数を数える整数型の変数 `count` を用意
- 整数型の変数 `i` (1から`num`まで1ずつ増やす) を用意し, `num` が `i` で割り切れたら `count` を1追加する
- 最後に `count` の値を返す (`return count;`)
- `println` で数と, 返って来た値を表示する



# 約数の数を求める関数



```
int getNumberOfDivisor( int num ){
    int i=1;
    int count=0;
    while( i<=num ){
        if( num%i == 0 ){
            count++;
        }
        i++;
    }
    return count;
}
```

戻り値(この値が、呼び出し元に返る)

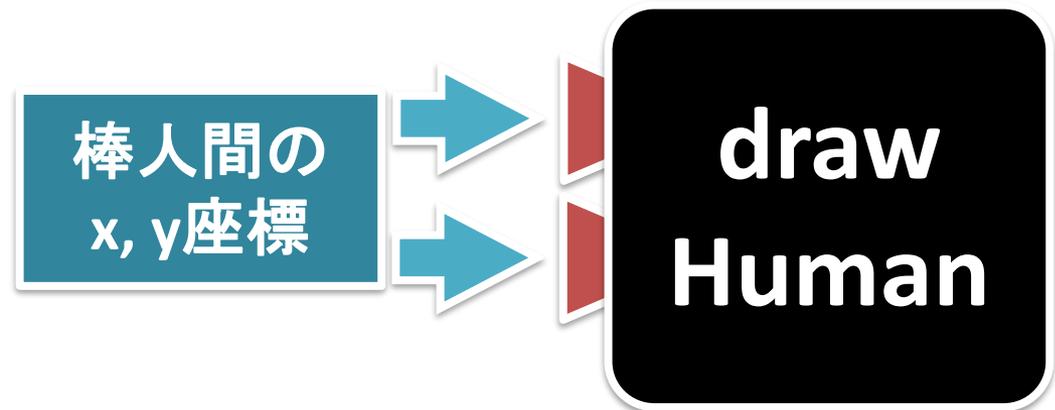
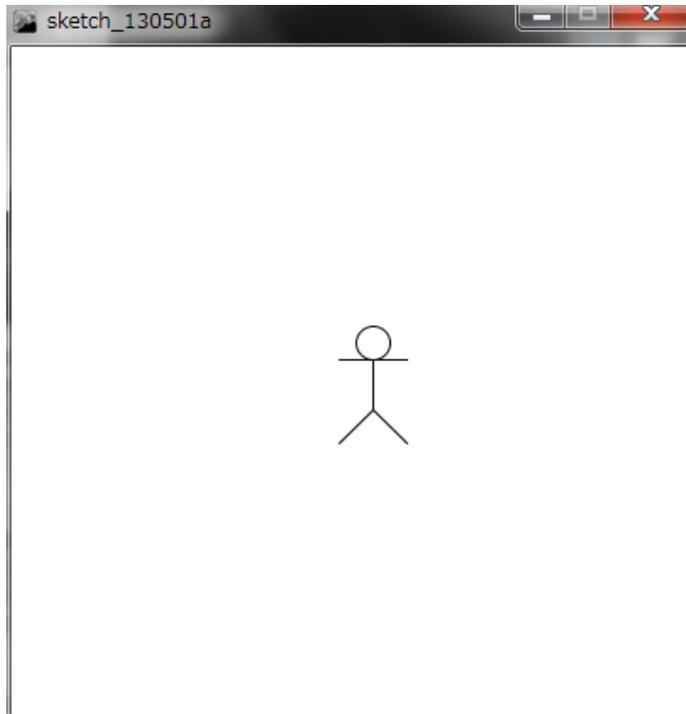
ここに戻る

```
void setup(){
    for( int i=1; i<100000; i++ ){
        println( i+": "+ getNumberOfDivisor(i) );
    }
}
```

# 引数〇, 返り値 ×



(Q)  $x, y$  座標を指定すると棒人間を描いてくれる関数を作成せよ!

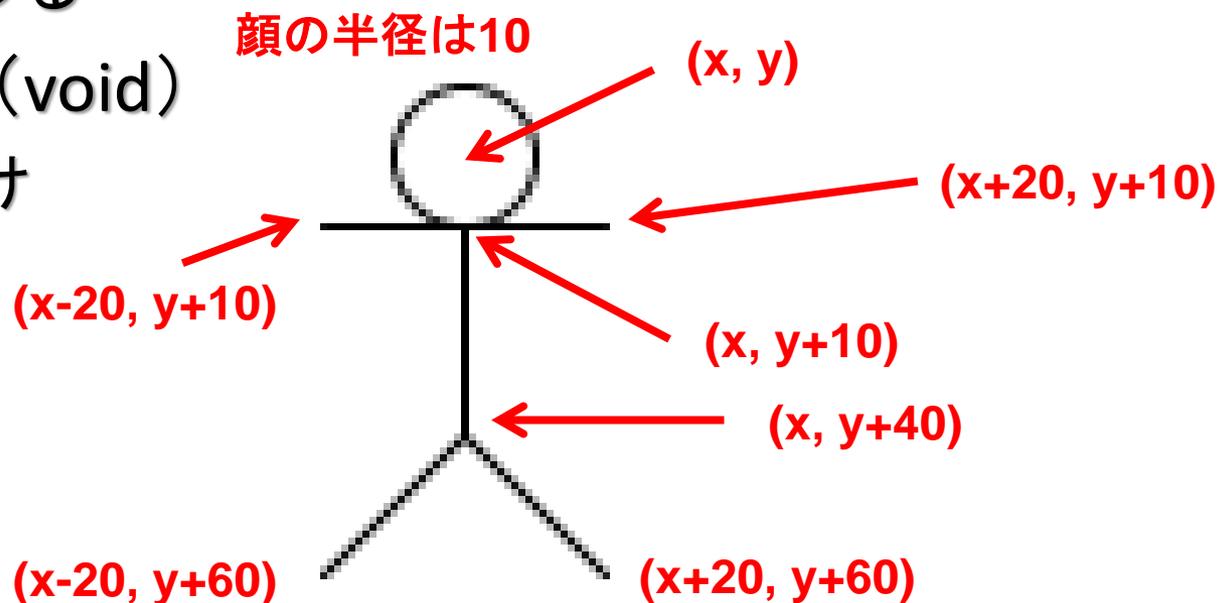


# 棒人間を描く



## • 考え方

- 棒人間は, 顔の中心の座標  $(x, y)$  を与えると, 勝手に体と手と足を描くものにする
- 棒人間の中心の座標を  $(x, y)$  としたときのそれぞれの座標を決める
- 返り値はなし (void)
  - 描画するだけ





- マウ斯卡ーソルの場所に棒人間を描く

```
void setup(){
    size( 400, 400 );
}
void drawHuman( int x, int y ){
    ellipse( x, y, 20, 20 );
    line( x, y+10, x, y+40 );
    line( x-20, y+10, x+20, y+10 );
    line( x, y+40, x-20, y+60 );
    line( x, y+40, x+20, y+60 );
}
void draw(){
    background( 255 );
    drawHuman( mouseX, mouseY );
}
```

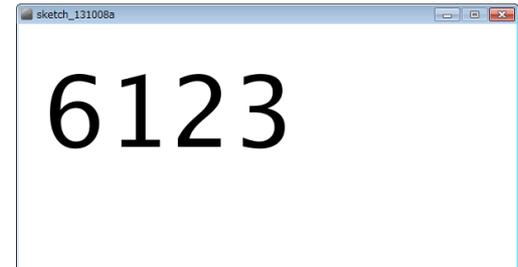
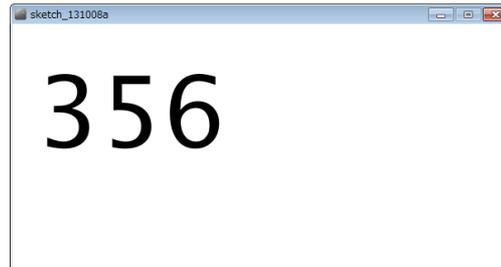
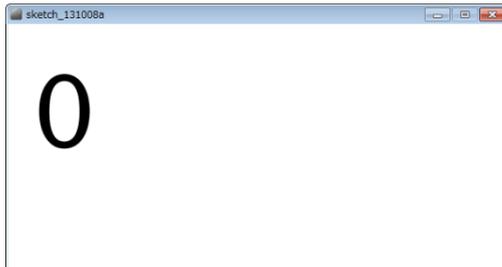
# 引数×, 返り値○



(Q) マウスをクリックしてから現在までの経過時間を表示するプログラムを作成し, 再度クリックすると経過時間を表示するプログラムを作成せよ

## • 考え方

- 現在の時間を, 0時0分0秒から換算して, 何秒目かを変換する関数(メソッド)を作成する
  - 時間は `hour()`, 分は `minute()`, 秒は `second()` で取得可能
  - メソッドは `int getNow(){ ... }` という形にしよう!

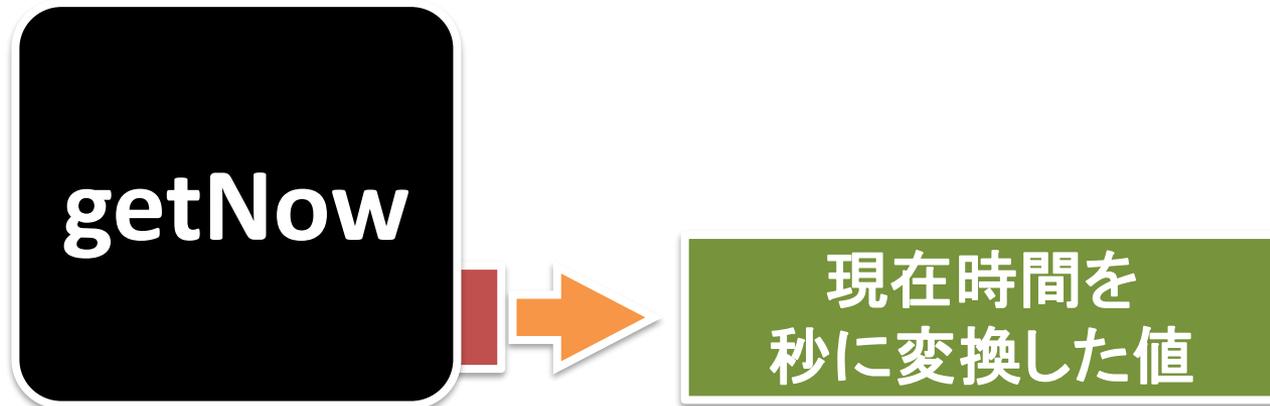


# 何秒目かを求める関数



## • 考え方

- boolean型 (true/falseのみ) のstartingという変数を用意
  - starting==falseなら動かない, starting==trueなら動く
- int型のstartTimeを用意し, マウスクリックされるとgetNow() を startTime に代入
- int型のendTimeを用意し, マウスクリックされるとgetNow() を endTime に代入
- starting==trueの間は, drawでgetNow()-startTime の値を表示する!





```
int getNow()
{
    return (hour()*60+minute())*60+second();
}
```

```
boolean start = false;
int startTime = 0;
int endTime = 0;
void setup() {
    size( 600, 300 );
    textSize( 120 );
}
void draw() {
    background(255);
    fill(0);
    if ( start == true ) {
        text( (getNow()-startTime), 30, 150 );
    } else {
        text( (endTime-startTime), 30, 150 );
    }
}
```

```
void mousePressed() {
    if ( start == false ) {
        startTime = getNow();
        start = true;
    }
    else if ( start == true ) {
        endTime = getNow();
        start = false;
    }
}
```

# getNow() 必要？



- 秒は  $(\text{hour()} * 60 + \text{minute()} * 60 + \text{second}());$  で計算できるので `getNow()` はいらないのでは？
- 毎回書くのは面倒だけど、コピペしたらいいし

本当にそれで良いですか？

# どっちが見やすい？



```
void draw() {  
  background(255);  
  fill(0);  
  if ( start == true ) {  
    text( ((hour()*60+minute())*60+second()-startTime), 30, 150 );  
  } else {  
    text( (endTime-startTime), 30, 150 );  
  }  
}
```

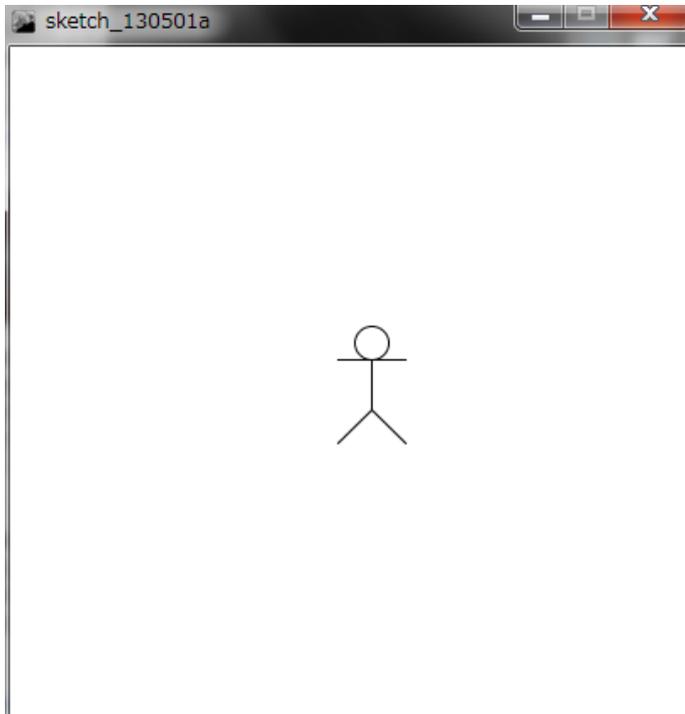
```
void draw() {  
  background(255);  
  fill(0);  
  if ( start == true ) {  
    text( (getNow()-startTime), 30, 150 );  
  } else {  
    text( (endTime-startTime), 30, 150 );  
  }  
}
```

こちらなら  
メソッドにした  
方が良い！

# 引数 × , 返回值 ×



(Q) マウスカーソルの位置に棒人間を描いてくれる関数を作成せよ！



# 引数 × , 返り値 ×



- mouseX, mouseY はグローバル変数だし...

```
void setup(){
  size( 400, 400 );
}
void drawHuman2(){
  ellipse( mouseX, mouseY, 20, 20 );
  line( mouseX, mouseY+10, mouseX, mouseY+40 );
  line( mouseX-20, mouseY+10, mouseX+20, mouseY+10 );
  line( mouseX, mouseY+40, mouseX-20, mouseY+60 );
  line( mouseX, mouseY+40, mouseX+20, mouseY+60 );
}
void draw(){
  background( 255 );
  drawHuman2();
}
```

グローバル変数を使  
えば何もなくても良い  
が、改良しにくい

# 演習1



- 500x500のウィンドウに, 50ピクセルずつあけて九九の表を作成せよ
  - 九九の表の数字が平方数の場合は, 赤色の文字で, それ以外の文字は黒色の文字で描くようにせよ
  - 平方数かどうかを判定するメソッド(引数を判定する数字, 戻り値をboolean型で, 引数で指定した数字が平方数の場合はtrueを, 平方数でない場合はfalseを返すものとせよ)も作成し, 利用せよ.
  - なお, 平方数の判定は単純に, 入力された数字について1から順に2乗の値と一致するかどうかで判定せよ