情報メディア基盤ユニット

関数その2

情報メディア学科佐藤尚

組み込み関数 → Processing言語側で用意している関数

組み込み関数の大まかの分類

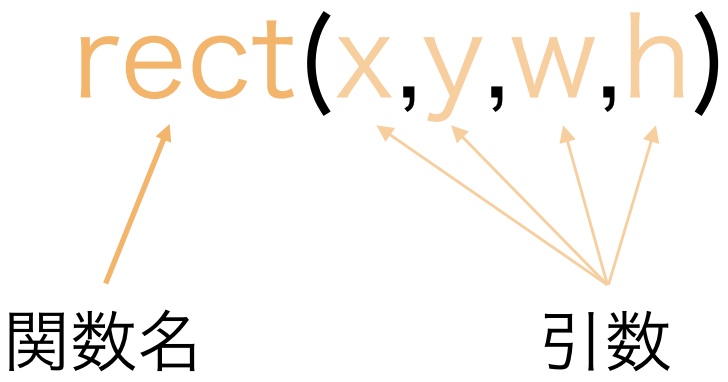
- 図形描画の関数
- ファイル関連の関数
- 時刻に関する関数
- 文字列関連の関数
- 数学関連の関数
- その他の関数

詳しくは、リファレンスを参照のこと

関数の分類

	引数がない	引数がある
戻り値がある	randomなど	mapなど
戻り値がない void型	smoothなど	rectなど

手続き



関数の分類

戻り値がない

何らかの処理を
ひとまとめにしたもの

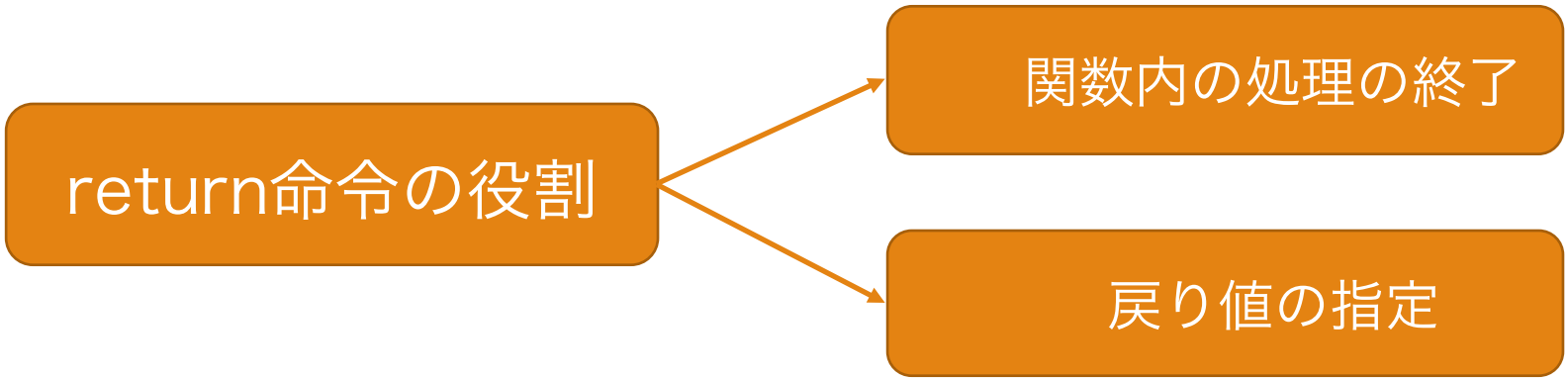
戻り値がある

何らかの処理の結果、
何かの答え（戻り値）が
求まるもの

戻り値：リターンバリュー

関数の宣言
（引数無し、戻り値あり）

```
戻り値のデータ型 関数名(){
        関数処理の内容を書きます。
        どこかに、return命令が必要です。
        変数なども使うことができます。
}
```



関数の宣言
（引数無し、戻り値あり）

戻り値はString型の値　　引数がないので空欄

```
String today(){
  String msg = year() +”/”+month() + “/”+day();
  return msg;
}
```

文字列
処理の基礎

文字列+文字列–＞文字列の連結
“kait”+”DeNA” -> “kaitDeNA”
数字+文字列、文字列+数字->文字列の連結

関数の宣言
（引数無し、戻り値あり）

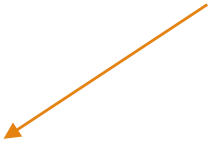
```
PFont font;
String today(){
  String msg = year() + "/" + month() + "/" + day();
  return msg;
}
void setup(){
  size(400,200);
  font = loadFont("SansSerif-48.vlw");
  textFont(font,48);
}
void draw(){
  background(255);
  fill(0);
  String str = today();
  text(str,100,height/2);
}
```

②ここに来るともとに戻る
戻り値は変数strに保存される

①ここに来ると関数todayに
処理が移動する

関数の宣言
（引数あり、戻り値あり）

引数（ひきすう）



戻り値のデータ型 関数名(データ型名1　引数名1,
　　データ型名2　引数名2…){

　　関数処理の内容を書きます。
　　どこかに、return命令が必要です。
　　変数なども使うことができます。
　　引数は自由に使うことができる。

}

引数：関数の処理に利用するデータ（値）を与える

仮引数

①ここに来ると処理がこの関数に移る、実引数の値が仮引数にコピーされる

```
boolean doesCollide(int x,int y,int xTip,int yTip){
  float d = dist(x,y,xTip,yTip);
  if(d <= 20){
    return true;
  }else if((abs(x - xTip) <= 20) && (y <= yTip)){
    return true;
  }else{
    return false;
  }
}
void setup(){
  size(400,400);
}
```

②ここに来ると戻り値が呼び出し元に戻される

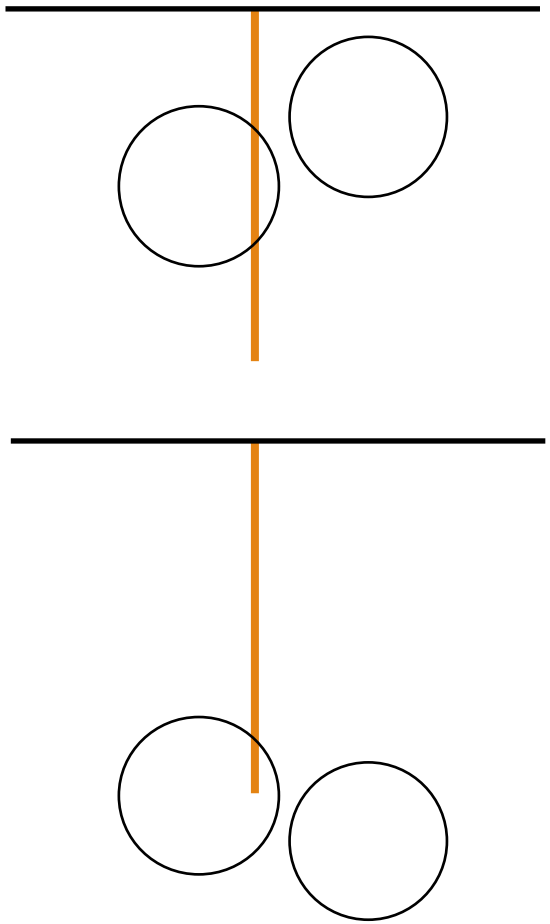
実引数

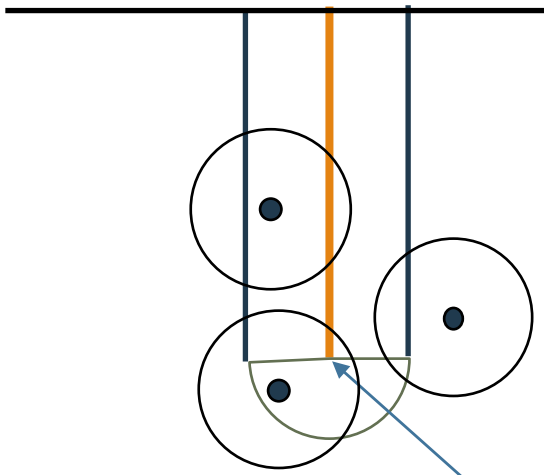
```
void draw(){
  background(255);
  stroke(0);
if(doesCollide(mouseX,mouseY,width/2,height/3)){
    fill(255,10,10);
  }else{
    fill(100);
  }
  line(width/2,0,width/2,height/3);
  ellipse(mouseX,mouseY,2*20,2*20);
}
```

仮引数

```
boolean doesCollide(int x,int y,int xTip,int yTip){
  float d = dist(x,y,xTip,yTip);
  if(d <= 20){
    return true;
  }else if((abs(x - xTip) <= 20) && (y <= yTip)){
    return true;
  }else{
    return false;
  }
}
```

線分と円の接触





円の中心がこの中に
入ると、線分と円が
接触する

ここの座標は
(xTip,yTip)

このプログラムでは、円の半径は20。

仮引数の有効範囲

仮引数の有効範囲は定義した関数の中だけ

関数が異なれば同じ名前の仮引数を使っても良い

```
戻り値のデータ型 関数名(データ型名1　引数名1,
                        データ型名2　引数名2…){
    関数処理の内容を書きます。
    どこかに、return命令が必要です。
    変数なども使うことができます。
    引数は自由に使うことができる。
}
```

組み込み関数

- 関数名と処理内容が既に決まっている
- 呼び出されるタイミングはプログラマが決める

ユーザ定義関数

- 関数名と処理内容はプログラマが決める
- 呼び出されるタイミングはプログラマが決める

コールバック関数

- 関数名はProcessing側で決めている
- 処理内容はプログラマが決める
- 呼び出されるタイミングはProcessing側が決める

コールバック関数の例

関数名	役割・呼び出されるタイミング
mousePressed	マウスボタンが押されたら
mouseClicked	クリックされたら
mouseMoved	マウスが移動したら
mouseDragged	マウスをドラッグしたら
mouseReleased	マウスボタンが離された
mouseWheel	マウスのホイールが回されたら
keyPressed	キーが押された（Shiftキーなどでも OK）
keyReleased	キーが押されなくなったら
keyTyped	キーが押されたら（Shiftキーなどはダメ）

基本的に戻り値は無し

押されたマウスボタンの種類：mouseButton変数でわかる
LEFT,CENTER,RIGHT

押されたキーの種類： key変数とkeyCode変数でわかる
key変数：どのキーが押されたかを保持している
key == ‘k’ ←小文字のkキーが押されたか？
key == ‘1’ ←1キーが押されたか？
key変数の値がCODEDの時は、特殊キーが押されたら、
どのキーが押されたかの情報がkeyCode変数に入っている

```
void mouseWheel(MouseEvent  event) {
  float e = event.getCount();
  println(e);
}
void mousePressed(){
  println("mouse pressed "+mouseButton);
}
void mouseDragged(){
  println("mouse dragged");
}
void mouseReleased(){
  println("mouse released");
}
void mouseMoved(){
  println("mouse moved");
}void keyPressed() {
  println("pressed " + int(key) + " " + keyCode);
}
```

```
void keyTyped() {
  println("typed " + int(key) + " " + keyCode);
}
void keyReleased() {
  println("released " + int(key) + " " + keyCode);
}
void setup(){
  size(200,200);
}
void draw(){
  background(255);
}
```

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。