

プログラミング基礎

関数 (function)

関数 (function)

- データを受け取って、何らかの計算や処理を行って、結果を返す命令

【書式】関数名(引数1, 引数2...)

line(10, 10, 30, 30);

関数名

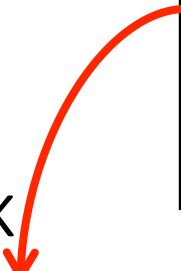
引数(データのこと)

- 数学の関数と見た目は似ている $f(x) = x^2 + 6x + 9$
- メソッド(method)と呼ぶ場合もある

関数の結果(戻り値)

- 結果を返さない関数
 - 例: 表示や描画を行う関数
- 結果を返す関数
 - 例: 数学関数

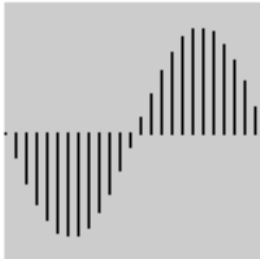
1つの文に
まとめてもOK



```
for(int i = 0; i < 100; i = i + 1) {  
    float x = TWO_PI * i / 100;  
    float h = sin(x) * 50;  
    line(i, 50, i, 50 + h);  
}
```

```
for(int i = 0; i < 100; i = i + 1) {  
    line(i, 50, i, 50 + sin(TWO_PI * i / 100) * 50);  
}
```

関数の使い方を調べる

Name	sin()		関数名
Examples		<pre>float a = 0.0; float inc = TWO_PI/25.0; for(int i=0; i<100; i=i+4) { line(i, 50, i, 50+sin(a)*40.0); a = a + inc; }</pre>	
Description	Calculates the sine of an angle. This function expects the values of the angle parameter to be provided in radians (values from 0 to 6.28). Values are returned in the range -1 to 1.		
Syntax	sin(rad)		
Parameters	rad	float: an angle in radians	引数
Returns	float	戻り値の型(結果)	

型の種類

- 整数型 (int)
- 小数型 (float, double)
 - double の方が表せる桁数が多い
- 論理型 (boolean)
 - true か false が入る型
- void
 - 関数が結果を返さないときに書く

関数を作る

オリジナルの関数を作る(1/2)

```
void setup(){  
  size(400, 400);  
  noStroke();  
}
```

setup関数: 自動的に最初に1回実行される

```
void draw(){  
  background(255);  
  drawKobe(mouseX, mouseY);  
}
```

draw関数: 自動的に繰り返し実行され続ける

オリジナルの関数 “drawKobe” を使う
使い方は他の関数と同じ

```
void drawKobe(int x, int y){  
  fill(40, 58, 137);  
  ellipse(x, y, 100, 100);  
  fill(255);  
  ellipse(x - 6, y - 6, 90, 90);  
  fill(27, 131, 65);  
  ellipse(x - 6, y - 6, 88, 88);  
  fill(255);  
  ellipse(x - 10, y, 80, 80);  
}
```

オリジナルの関数 drawKobe を作る
詳細は次のページ

オリジナルの関数を作る(2/2)

【書式】 結果の型 関数名(引数1の型 名前, 引数2...){ ... }



```
void drawKobe(int x, int y){  
    fill(40, 58, 137);  
    ellipse(x, y, 100, 100);  
    fill(255);  
    ellipse(x - 6, y - 6, 90, 90);  
    fill(27, 131, 65);  
    ellipse(x - 6, y - 6, 88, 88);  
    fill(255);  
    ellipse(x - 10, y, 80, 80);  
}
```

関数の中身は
引数の x や y を
変数のように
使って書く

練習2ー1

- 国旗を描く関数を作って、国旗がマウスカーソルについていくプログラムを作ろう
 - 前回の課題で作ったプログラムを改造
- ヒント1: 引数は drawKobe と同じ int x, int y
- ヒント2: 描画する位置を x と y で表す
 - 例: 元々 (0, 0) に描画 → (x, y) に描画
 - 例: 元々 (20, 40) に描画 → (x+20, y+40) に描画

結果を返す関数を作る

- 例: 5月の日付を受け取って曜日を返す関数
dayOfWeek
 - 日曜=0、月曜=1、...土曜=6と数字で表す

```
void setup(){  
  for(int i = 1; i <= 31; i++){  
    println("5月" + i + "日:" + dayOfWeek(i));  
  }  
}  
  
int dayOfWeek(int date){  
  return (date + 6) % 7;  
}
```

関数の中で return 文を使う

【書式】return 結果;

練習2-2

- うるう年かどうか判定する関数を作る
 - 引数は判定したい年 (int)
 - 結果の型は boolean (true か falseが入る型)
 - 西暦年が4で割り切れる年は閏年
 - ただし、西暦年が100で割り切れる年は平年
 - ただし、西暦年が400で割り切れる年は閏年
- ヒントは次のスライド

ヒント

```
void setup(){
  for(int year = 2000; year <= 2100; year = year + 1){
    if(isLeapYear(year)){
      println(year + "年はうるう年です");
    }
    else{
      println(year + "年はうるう年ではありません");
    }
  }
}

[redacted] isLeapYear([redacted] year){
  if([redacted]){
    return true;
  }
  else{
    return false;
  }
}
```

課題2: カレンダーの描画

- 要件

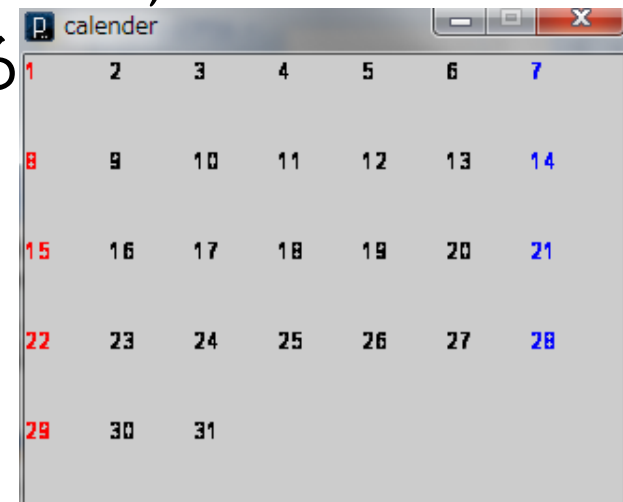
- 関数 `drawCalendar` を作る

- レベル1: 今年の5月のカレンダー `drawCalendar()`
 - レベル2: 今年の好きな月のカレンダー
`drawCalendar(int month)`
 - レベル3: 2000年以降の好きな年の好きな月のカレンダー
`drawCalendar(int year, int month)`

- 関数 `setColor(int day)` を作る

- 日曜は赤、土曜は青、平日は黒

- ヒントは次のスライド



ヒント(レベル1)

- 字を描画する関数は `text`
 - 使い方を調べてみよう
- まずはすべて同じ色で描画してみよう

ヒント(レベル2 & 3)

1. ある月が何日まであるかを返す関数を作る
 - 2月は年によって違う！
2. 1で作った関数を使って、2000年1月1日から数えて何日かを返す関数を作る
 - 例: 2011年5月12日は4150日目
3. 2で作った関数を使って、ある日付の曜日を返す関数を作る

付録: ☆を描く関数 drawStar

```
void setup(){  
  size(400, 400);  
  noStroke();  
  fill(0, 255, 0);  
}
```

```
void draw(){  
  background(255);  
  drawStar(mouseX, mouseY, 20);  
}
```

```
void drawStar(int x, int y, int r){  
  beginShape();  
  vertex(x, y - r);  
  vertex(x + r * sin(PI/5), y + r * cos(PI/5));  
  vertex(x - r * cos(PI/10), y - r * sin(PI/10));  
  vertex(x + r * cos(PI/10), y - r * sin(PI/10));  
  vertex(x - r * sin(PI/5), y + r * cos(PI/5));  
  endShape(CLOSE);  
}
```