



---

# プログラミング演習(6)

## 繰り返し

---

中村, 高橋  
小林, 橋本



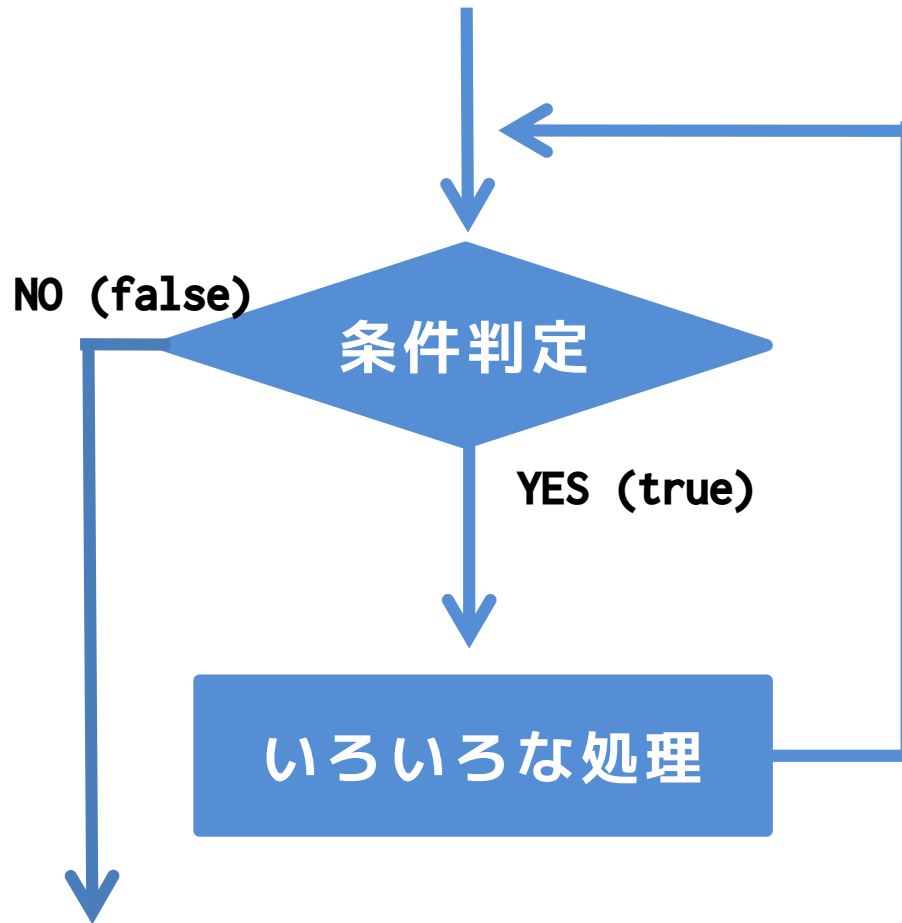
- Processing で繰り返しに挑戦！
  - 1から1000まで足しあわせた値は？
  - $\sum_{i=1}^{100}(3i + 1)$  の計算結果は？
  - 沢山同じ絵を描画してみる
- 課題：
  - Processing で色々な計算をしてみよう
  - Processing で同じ絵をたくさん書いてみよう

# 繰り返しとは

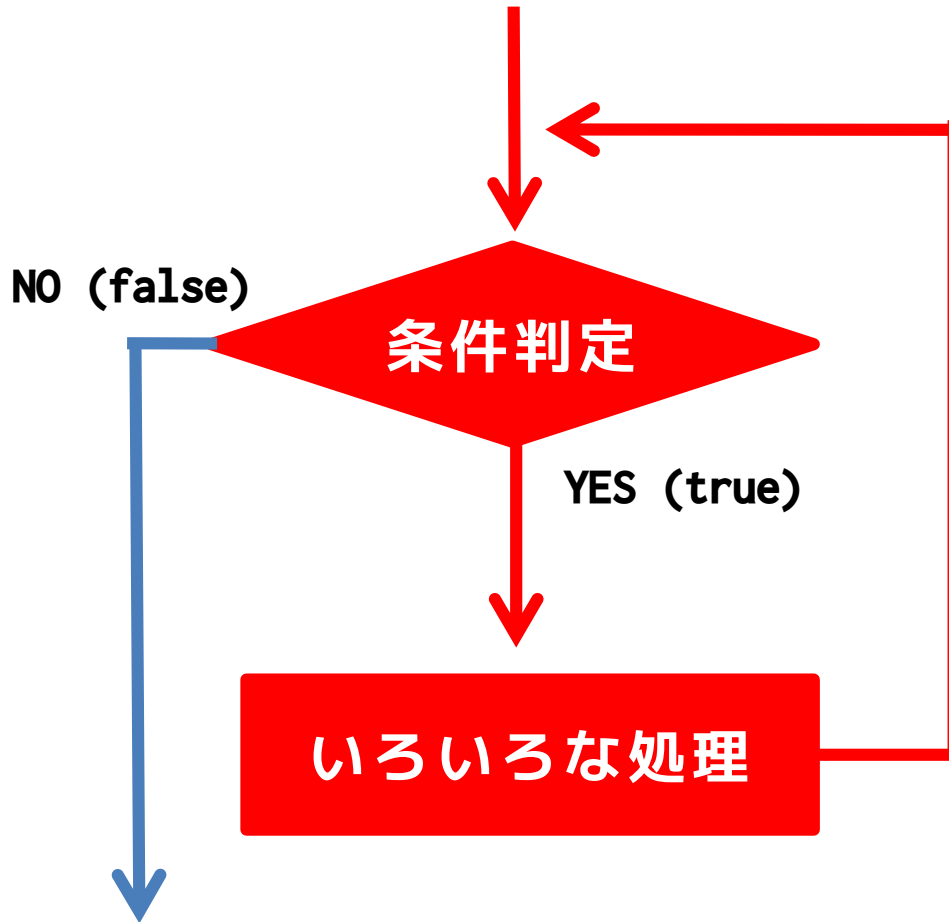


- 「何かの条件」を満たしている間ずっと、その処理をするというもの
- 例
  - 「100まで数えて」
  - 「30からカウントダウンして」
  - 「10回やって」
  - 「ぞろ目が出るまでやって」

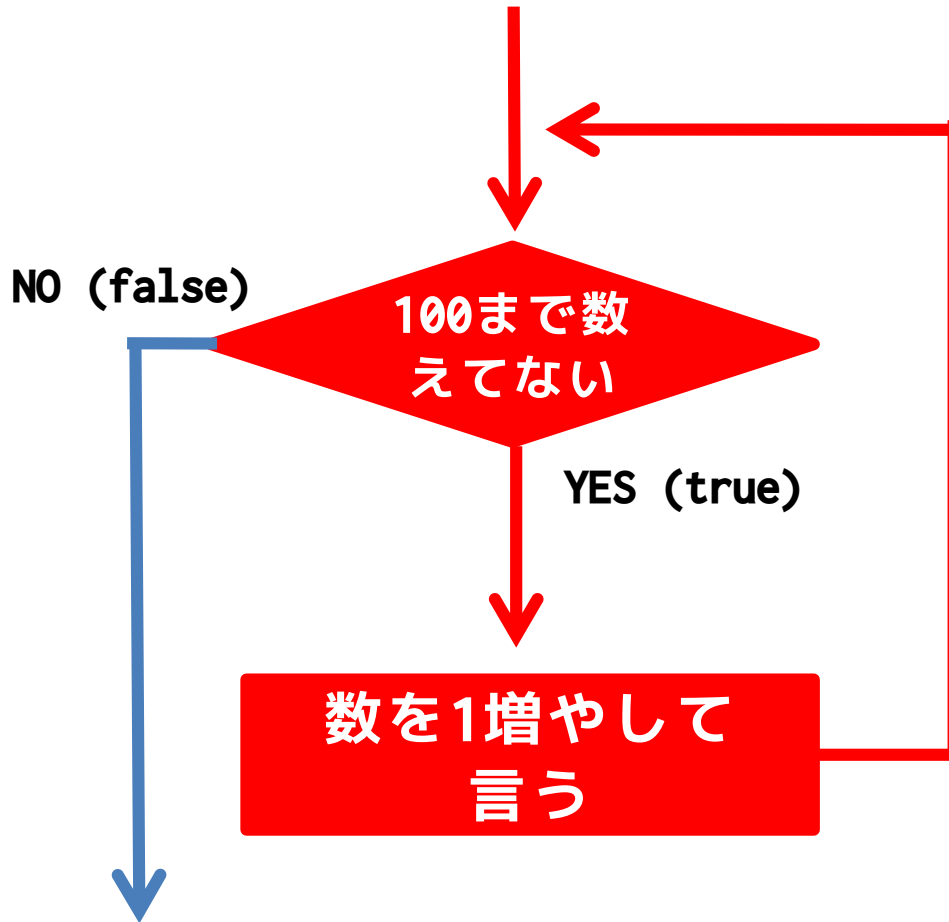
# 繰り返しの仕組み



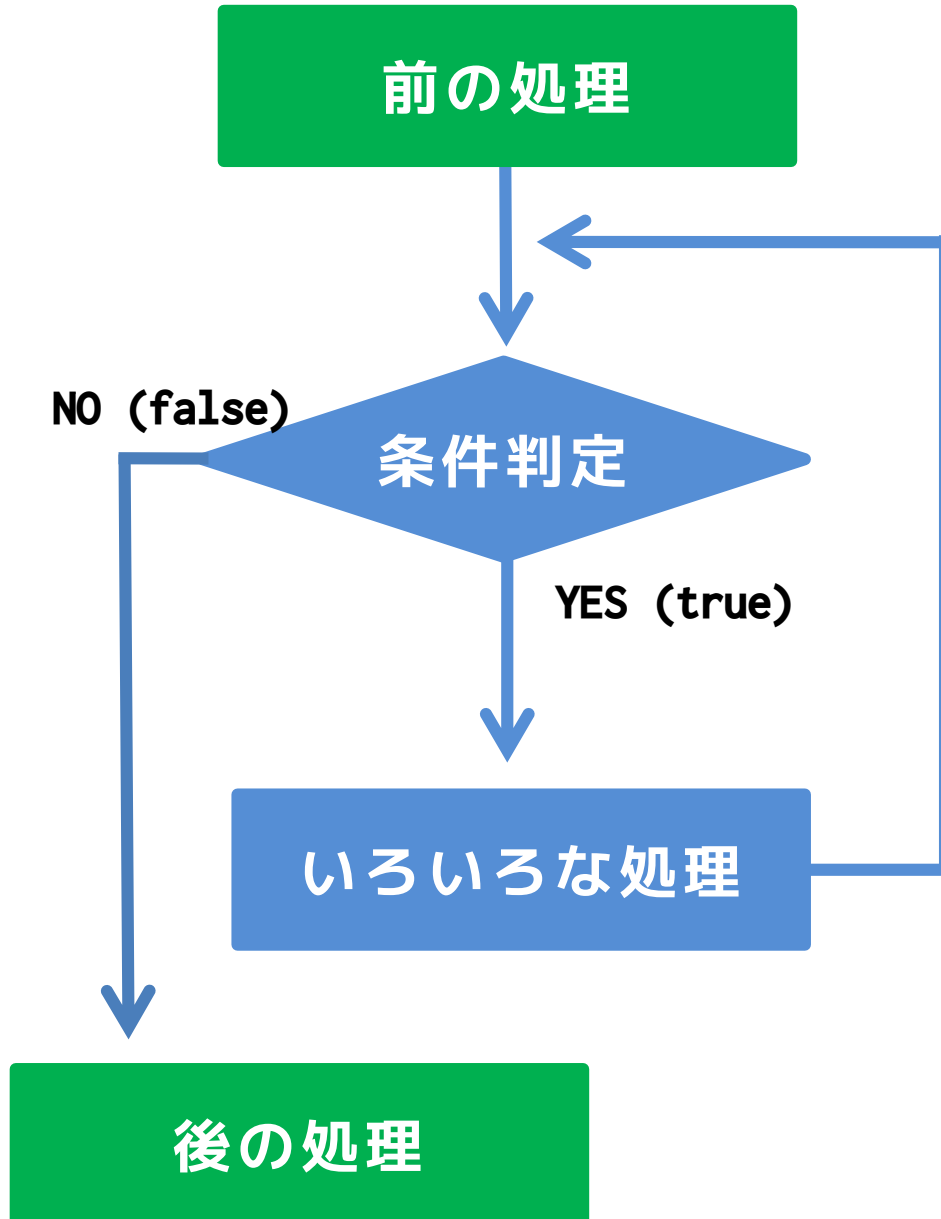
# 繰り返しの仕組み



# 繰り返しの仕組み



# 繰り返しの仕組み



前の処理

while (条件判定)

{

いろいろな処理

}

後の処理

# (Q) 1から100まで表示



1から100までの数字を改行しながら標準出力していきましょう

```
1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18
```



# (A) 1から100まで表示



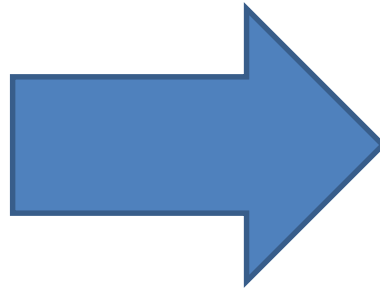
```
println(1);  
println(2);  
println(3);  
println(4);  
println(5);  
println(6);  
:  
println(95);  
println(96);  
println(97);  
println(98);  
println(99);  
println(100);
```

これでできるけど無駄が多い

# 繰り返しを使うと . . .



```
println(1);  
println(2);  
println(3);  
println(4);  
println(5);  
println(6);  
:  
println(95);  
println(96);  
println(97);  
println(98);  
println(99);  
println(100);
```



```
int i = 1;  
while(i <= 100)  
{  
    println(i);  
    i = i + 1;  
}
```

100行

短くなった！！

6行

# 繰り返しのパターン



- 2つの種類の繰り返し
  - 既定の回数繰り返す
    - 「100まで数えて」
    - 「30からカウントダウンして」
    - 「10回やって」
  - 何らかの条件を満たすまで繰り返す
    - 「ぞろ目が出るまでやって」

# 既定の回数繰り返しの場合



- 100まで数える場合

```
int i = 1; // 初期化する

// 繰り返す
while(i <= 100)
{
    // 色々な処理
    println(i); // 数字を出力する
    i = i + 1;   // 数を増やす
}
```

「初期化」と「数を増やす」を忘れがち

# 既定の回数繰り返しの場合



- 100まで数える場合（forが便利）

## 前の処理

```
for (初期化; 条件判定; 数を変化させる処理)
{
    いろいろな処理
}
```

## 後の処理

```
int i = 1;
while(i <= 100)
{
    println(i);
    i = i + 1;
}
```



```
for(int i=1; i <= 100; i = i + 1)
{
    println(i);
}
```

ちょっと不思議な構文だけど間違えにくい

# 繰り返しを使うと . . .



```
println(1);  
println(2);  
println(3);  
println(4);  
println(5);  
println(6);  
:  
println(95);  
println(96);  
println(97);  
println(98);  
println(99);  
println(100);
```



```
for(int i = 1; i <= 100; i++)  
{  
    println(i);  
}
```

100行

4行

もっと短くなった！！

# 条件の記述方法



演算子	意味	プログラム上
$x > y$	x が y より大きい	左記の時に true それ以外で false
$x < y$	x が y より小さい	同上
$x \geq y$	x が y 以上	同上
$x \leq y$	x が y 以下	同上
$x == y$	x と y が等しい	同上
$x != y$	x と y が等しくない	同上
$!x$	x は false?	同上

# 省略記法



式	意味
$x++$	$x = x + 1$
$x--$	$x = x - 1$
$x *= 10$	$x = x * 10$
$x /= 10$	$x = x / 10$
$x \% = 10$	$x = x \% 10$
$x += 10$	$x = x + 10$
$x -= 10$	$x = x - 10$



# (Q) 10からカウントダウン

明治大学総合数理学部  
先端メディアサイエンス学科  
中村研究室



10から0までカウントダウンしてみよう

10  
9  
8  
7  
6  
5  
4  
3  
2  
1  
0

# 繰り返しの仕組み



i を 10 にする

NO (false)

$i \geq 0$

YES (true)

i を出力

特になし

前の処理

while (条件判定)

{  
}

いろいろな処理

後の処理

# 繰り返しの仕組み



i を 10 にする

NO (false)

$i \geq 0$

YES (true)

i を出力

特になし

```
i = 10;
```

```
while (i >= 0)
```

```
{
```

```
    println(i);
```

```
    i--; // i=i-1;
```

```
}
```

# (A) 10からカウントダウン



$i$  という変数を利用して初期化し, その変数で繰り返しの処理を行う

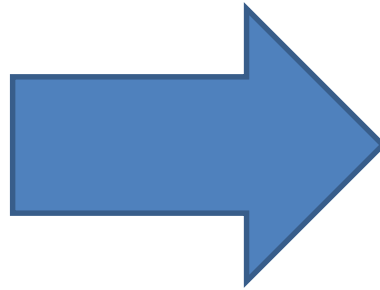
```
int i = 10;
while(i >= 0)
{
    println(i);
    i = i - 1;
    // i--; でもok
}
```

```
for(int i = 10; i >= 0; i--)
{
    println(i);
}
```

# 繰り返しを使うと . . .



```
println(1);  
println(2);  
println(3);  
println(4);  
println(5);  
println(6);  
:  
println(95);  
println(96);  
println(97);  
println(98);  
println(99);  
println(100);
```



```
int i = 1;  
while(i <= 100)  
{  
    println(i);  
    i = i + 1;  
}
```

**100行**

**短くなった！！**

**6行**

# (Q) 1から100まで表示



1から100までの数字を順に標準出力していき  
ましょう。ただし10個ずつで改行しよう

```
1 2 3 4 5 6 7 8 9 10
11 12 13 14 15 16 17 18 19 20
21 22 23 24 25 26 27 28 29 30
31 32 33 34 35 36 37 38 39 40
41 42 43 44 45 46 47 48 49 50
51 52 53 54 55 56 57 58 59 60
61 62 63 64 65 66 67 68 69 70
71 72 73 74 75 76 77 78 79 80
81 82 83 84 85 86 87 88 89 90
91 92 93 94 95 96 97 98 99 100
```

# (A) 1から100まで表示



- ある回数ごとに改行するなどの条件を入れてください！

```
for(int i = 1; i <= 100; i++)  
{  
    print(i);           // i を出力  
    print(" ");        // 空白を出力  
    if(i % 10 == 0)    // 10個ずつ  
    {  
        println();    // 改行する  
    }  
}
```

# 演習



1から100までの数字を順に標準出力していきましょう。10個ずつで改行し、下記のように整形せよ

ヒント：1桁と2桁でスペースの入れ方を工夫しよう

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100



# (Q) ぞろ目がでるまで



2つのサイコロをふり，そのときのサイコロの目を表示し，それをぞろ目が出るまで繰り返してみましよう

## • 考え方

- 2つのサイコロを振るのはこれまで通り
- サイコロの目を表示する
- 2つのサイコロの値が一致したら終わり

# 繰り返しの仕組み



さいころ2つの  
値を初期化

NO (false)

目が一致  
しない

YES (true)

さいころ2つ振る

さいころの目を表示

ぞろ目ですと出力

さいころ2つ初期化

while (目が一致しない?)

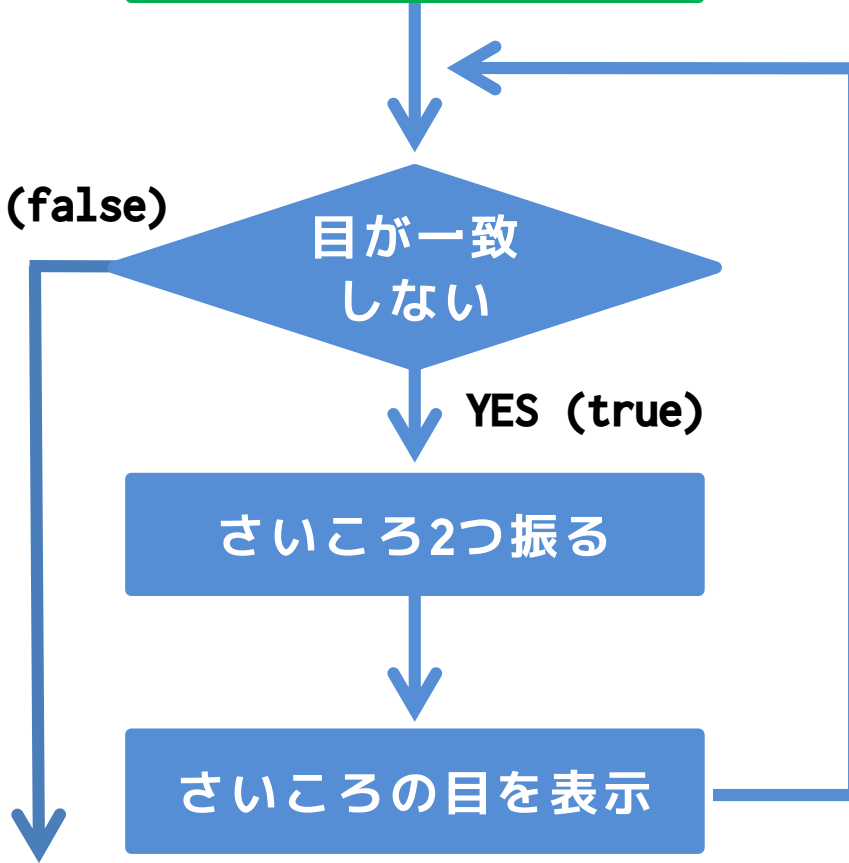
{

さいころ2つ振る

さいころの目を表示

}

ぞろ目ですと出力



# (A) ぞろ目がでるまで



2つさいころを振って，同じ値なら終了する  
同じ値でなければ終了しない

```
// 初期値はあえて変な値を入れておく
int diceA = -1;
int diceB = -2;

while(diceA != diceB)
{
    diceA = (int)random(1, 7);
    diceB = (int)random(1, 7);
    println(diceA + ", " + diceB);
}

println("ぞろ目です");
```

# (A) ぞろ目がでるまで



繰り返しを抜ける別の方法として break というものがある（breakがあると繰り返しから脱出！）

```
// 無限ループ
while(true)
{
    int diceA = (int)random(1, 7);
    int diceB = (int)random(1, 7);
    println(diceA + ", " + diceB);
    if(diceA == diceB)
    {
        // ループからの脱出！
        break;
    }
}

println("ぞろ目です");
```

# 次に計算



- $\sum_{i=1}^{1000} i$  の計算結果は？
- つまり、1から1000までの和は？
  - `println(1+2+3+4+5+...+998+999+1000);` と書くのはしんどい（というか長すぎる）
  - じゃあ、どうやって計算するのか？
  - まずは、1から5までの和で考えてみる

# 少し考えてみる



- 合計を格納する変数を `int` (整数) の `total` とする
- `total` の初期値を `0` とする (`total = 0;`)
- 1から5までの和は `total = 1 + 2 + 3 + 4 + 5;`
- これを分解すると  
`total = total + 1;`  
`total = total + 2;`  
`total = total + 3;`  
`total = total + 4;`  
`total = total + 5;`  
となる。

# さらに考えてみる



- 分解したものの赤枠内に注目してみる

```
total = total + 1;
```

```
total = total + 2;
```

```
total = total + 3;
```

```
total = total + 4;
```

```
total = total + 5;
```

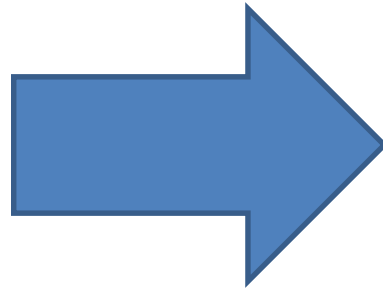
- 1から順に増えている！つまり例えば、整数の変数  $i$  を用意して、 $i$  を毎回 `total` に加算しては？

# こんなかんじになる？



```
int total = 0;
total = total + 1;
total = total + 2;
total = total + 3;
total = total + 4;
total = total + 5;
println(total);
```

7行



```
int total = 0;
int i = 1;
total = total + i;
i = i + 1;
total = total + i;
i = i + 1;
total = total + i;
i = i + 1;
total = total + i;
i = i + 1;
total = total + i;
println(total);
```

12行

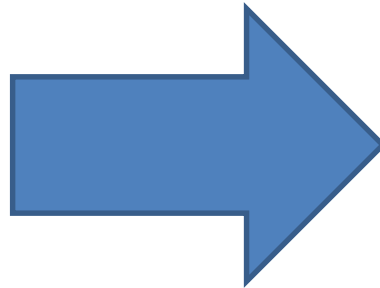
長くなった！！



# 繰り返しを使うと . . .



```
int total = 0;
int i = 1;
total = total + i;
i = i + 1;
total = total + i;
i = i + 1;
total = total + i;
i = i + 1;
total = total + i;
i = i + 1;
total = total + i;
println(total);
```



```
int total = 0;
int i = 1;
while(i <= 5)
{
    total = total + i;
    i = i + 1;
}
println(total);
```

12行

短くなった！！

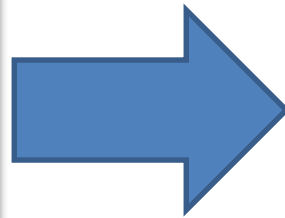
8行

# じゃあ、1000までの和は？



```
int total = 0;
total = total + 1;
total = total + 2;
total = total + 3;
total = total + 4;
total = total + 5;
:
total = total + 998;
total = total + 999;
total = total + 1000;
println(total);
```

**1002行**



```
int total = 0;
int i = 1;
while(i <= 1000)
{
    total = total + i;
    i = i + 1;
}
println(total);
```

**8行**

**かなり短くなった！！**

# じゃあ、1000までの和は？



```
int total = 0;
total = total + 1;
total = total + 2;
total = total + 3;
total = total + 4;
total = total + 5;
:
total = total + 998;
total = total + 999;
total = total + 1000;
println(total);
```

**1002行**

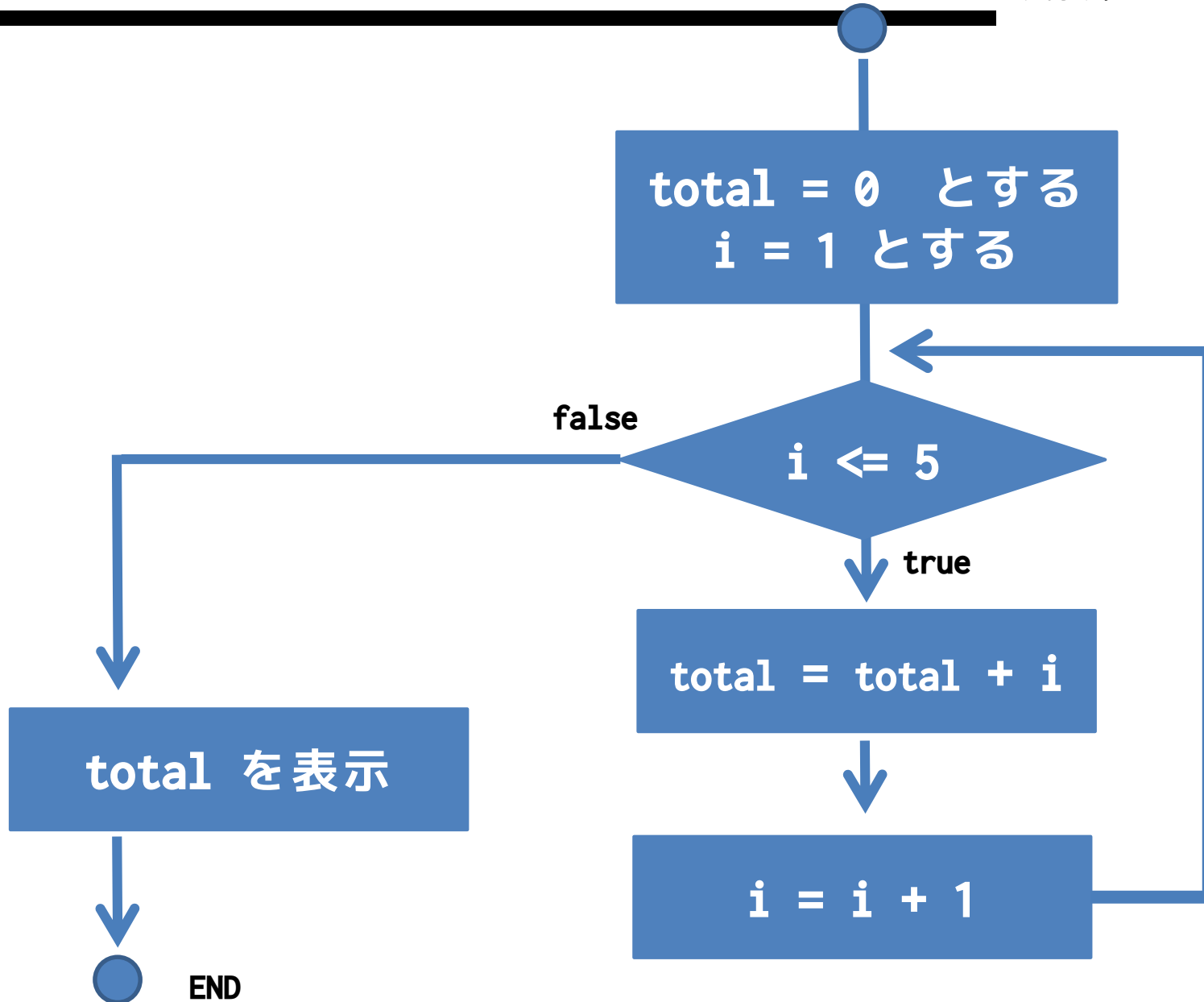


```
int total = 0;
for(int i = 1; i <= 1000; i++)
{
    total = total + i;
}
println(total);
```

**6行**

**かなり短くなった！！**

# 繰り返しの仕組み



# 予習問題

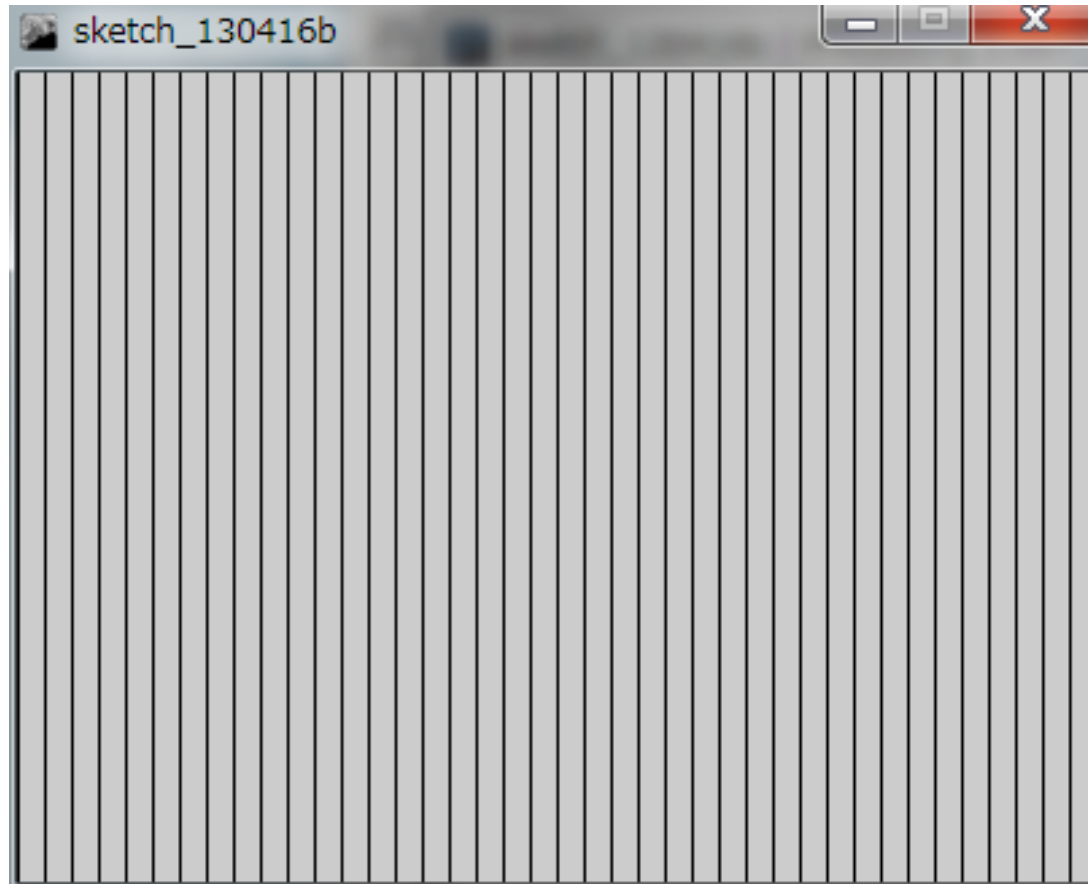


- 1から500までの和を計算してみましょう
  - $\sum_{i=1}^{500} i$
- 1から10000までの和を計算してみましょう
- 1から12345までの和を計算してみましょう
- 1から1000までの偶数の和を計算してみましょう
  - 偶数の場合は,  $i$ を2から始め, 1回毎に2足す
  - $\sum_{i=1}^{500} 2i$  としてもよい
- 1から1000までの奇数の和を計算してみましょう
  - 奇数の場合は,  $i$ を1から始め, 1回毎に2足す
  - $\sum_{i=1}^{500} (2i - 1)$  としてもよい

# 繰り返して線を描こう



(Q)  $400 \times 300$ のウィンドウに10ピクセルごとに、下記のような線を描くにはどうするか？



# 繰り返して線を描こう



- 10ピクセルごとの線は

- line(0, 0, 0, 300);
- line(10, 0, 10, 300);
- line(20, 0, 20, 300);
- line(30, 0, 30, 300);
- line(40, 0, 40, 300);
- line(50, 0, 50, 300);

```
line( 0, 0, 0, 300);  
line(10, 0, 10, 300);  
line(20, 0, 20, 300);  
line(30, 0, 30, 300);  
line(40, 0, 40, 300);  
line(50, 0, 50, 300);
```

とはいえ、全部書くのはあまりにしんどい

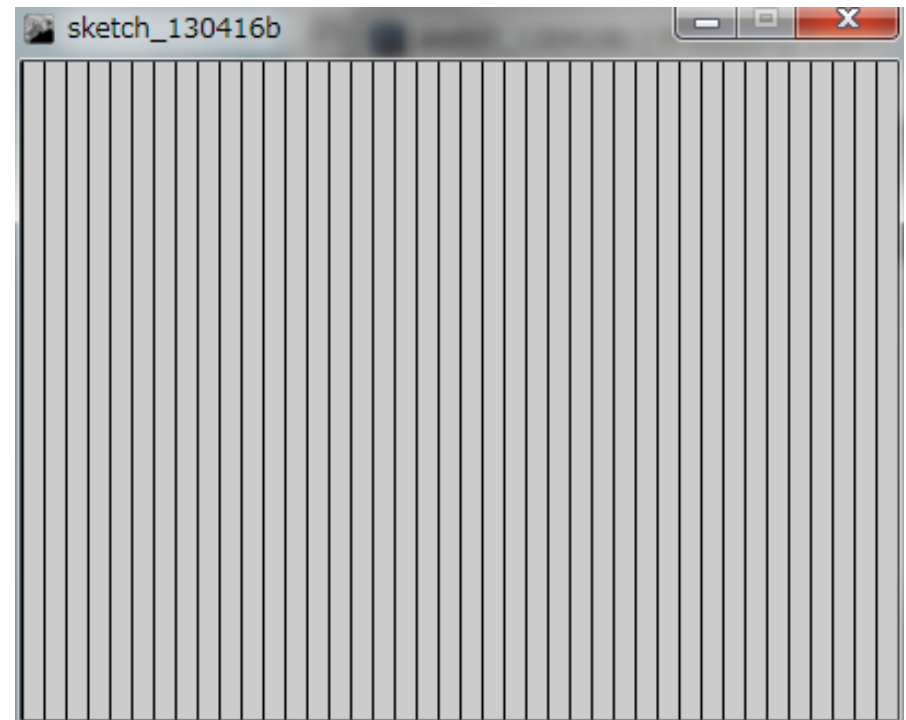
# 繰り返して線を描こう



- 変数  $x$  に  $0$  を設定
- $(x, 0)$  から  $(x, 300)$  まで線を描く
- $x$  を  $400$  まで  $10$  ずつ増やす

```
int x = 0;
while(x <= 400)
{
    line(x, 0, x, 300);
    x = x + 10;
}
```

```
for(int x = 0; x <= 400; x+=10)
{
    line(x, 0, x, 300);
}
```

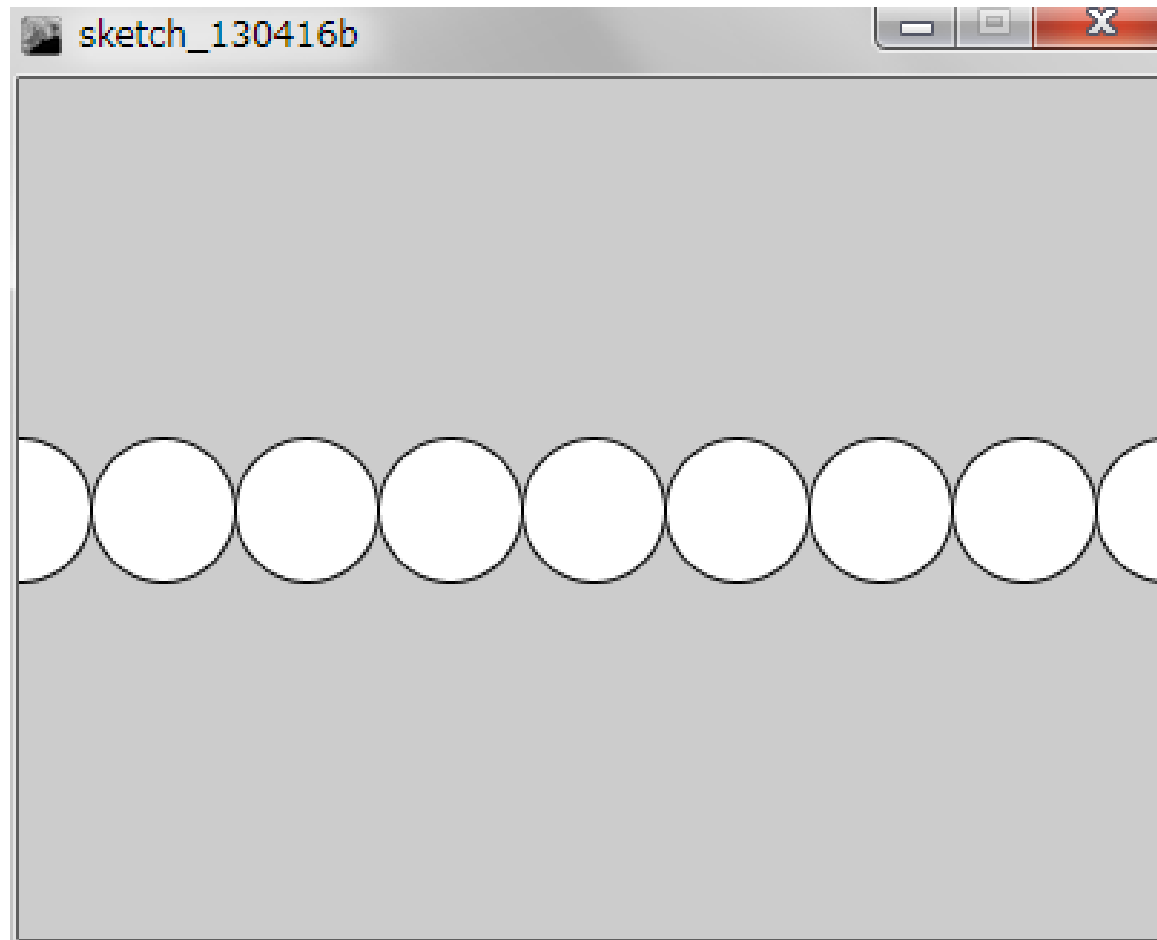




# 繰り返して円で描こう



(Q) 400x300のウィンドウに50ピクセルごとに  
下記のような直径50の円を描く



# 繰り返して円で描こう



- 考え方

- 中心のY座標はずっと中心の150
- 中心のX座標がどんどん変化する
  - 0, 50, 100, 150, 200, 250, 300, 350, 400
- X座標を変数にして, 0から400まで50ずつ増やす
- `ellipse(x, 150, 50, 50);` で円を描く

# 繰り返して円を描こう



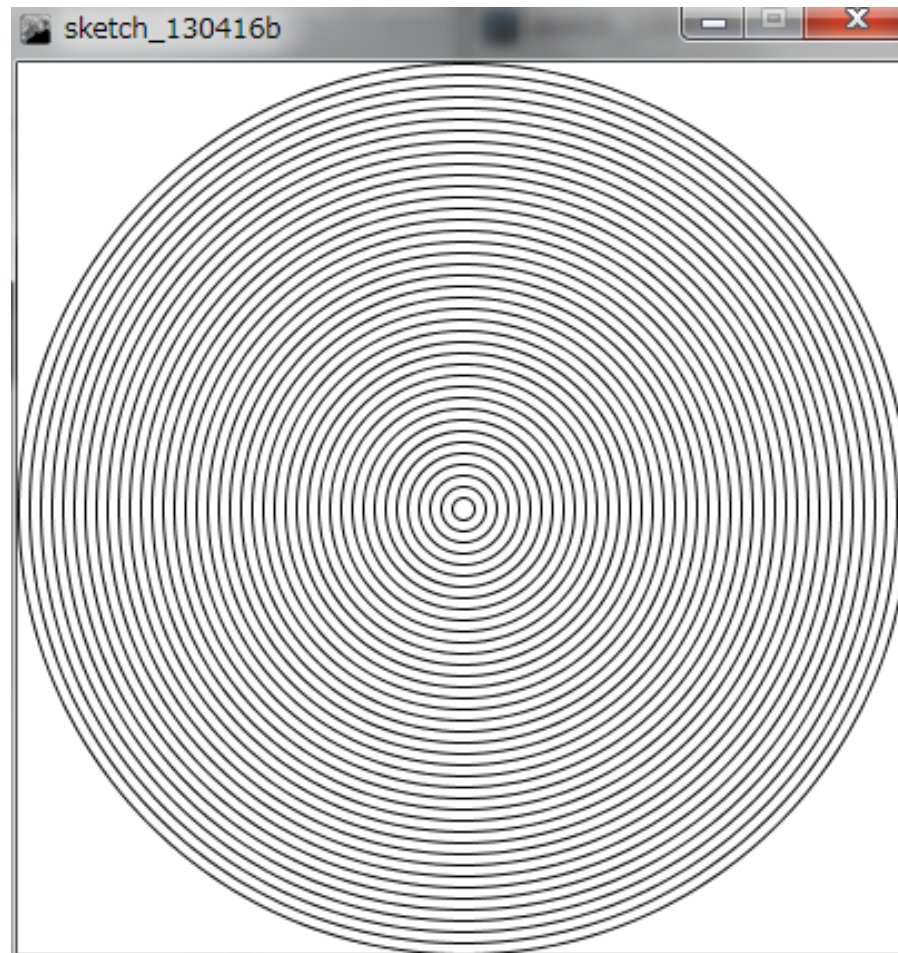
```
void setup()
{
  size(400, 300);
}

void draw()
{
  int x = 0;
  while(x <= width)
  {
    ellipse(x, 150, 50, 50);
    x = x + 50;
  }
}
```

# 円の中に円を描く



(Q) 400x400のウィンドウに直径の差が10ずつ  
変化するたくさんの円を描くにはどうする？



# 円の中に円を描く



- 考え方

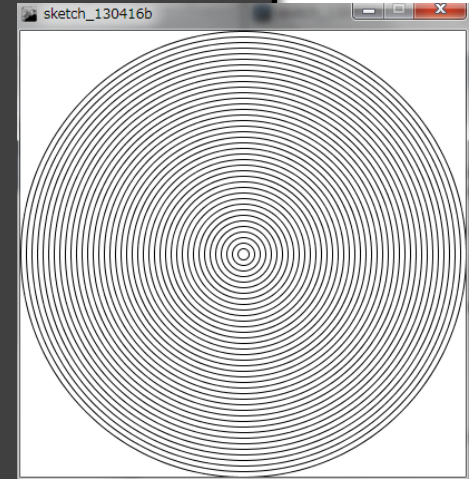
- 中心のX, Y座標はずっと中心の(200, 200)
- 直径は, 400, 390, 380, 370, ..., 30, 20, 10と変化
- 直径の長さの変数を diameter とする
- diameter = 400 とする
- `ellipse(200, 200, diameter, diameter);` で円を描く
- diameter を1回毎に10ずつ減らす

# 円の中に円を描く



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

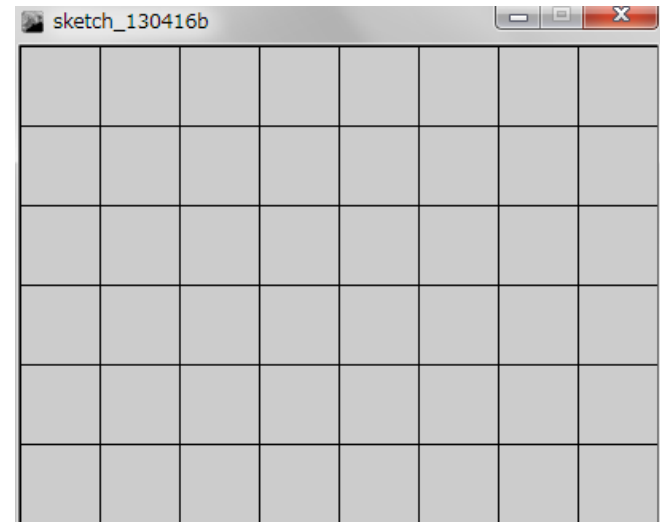
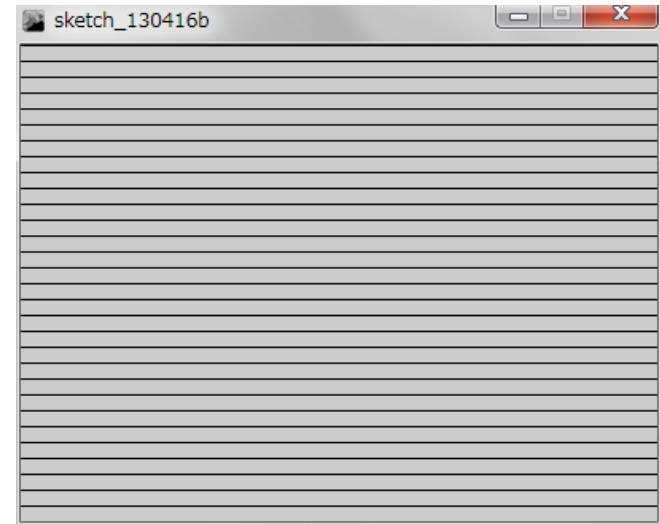
void draw()
{
  background(255, 255, 255);
  noFill();
  int diameter = 400;
  while(diameter > 0)
  {
    ellipse(200, 200, diameter, diameter);
    diameter = diameter - 10;
  }
}
```



# 予習問題



- 400x300のウィンドウに、  
繰り返しで横に10ピクセル  
ずつの線を描いてみま  
しょう
- 400x300のウィンドウに、  
繰り返しで横と縦に50ピ  
クセルのグリッドを書い  
てみましょう
  - 1度にまとめて描くか、2つ  
ループ使うかどちらか



# 沢山の線をアニメーション

明治大学総合数理学部  
先端メディアサイエンス学科  
中村研究室



(Q) 400x300のウィンドウで50ピクセルごとに描画した縦線を右方向にアニメーションする





# 沢山の線をアニメーション

明治大学総合数理学部  
先端メディアサイエンス学科  
中村研究室



- 考え方
  - 線を  $(x, 0)$  から  $(x, 300)$  まで描画
  - 次は  $(x + 50, 0)$  から  $(x + 50, 300)$  まで描画
  - 次は  $(x + 100, 0)$  から  $(x + 100, 300)$  まで描画
  - 最初の  $x$  の値を変化させることでアニメーション



```
int startX = 0;
```

**描画開始場所設定用の startX**

```
void setup()
{
  size(400, 300);
}
```

```
void draw()
{
  background(255, 255, 255);
```

**線の描画開始場所を startX で設定**

```
int x = startX;
while(x <= width)
```

```
{
  line(x, 0, x, height);
  x = x + 50;
```

**startX から始めて 50 毎に描画**

```
}
startX++;
if(startX == 50)
```

**startX が 50 になったら戻る**

```
{
  startX = 0;
```

```
}
```

# 正の約数を求める



(Q) 12345の正の約数を求める。どうする？

- 1, 3, 5, 15, 823, 2469, 4115, 12345 と計算できたらOK!
- 正の約数とは, ある1以上の自然数に対して, 割り切ることができる1以上の自然数のこと

# 正の約数を求めてみる



## • 考え方

- 12345 をある整数の変数  $num$  で割った時の余りが  $0$  の時, その変数  $num$  は12345の正の約数である
- 余りが  $0$  でない場合, その変数  $num$  は12345の正の約数ではない
- 余りを計算する方法は  $12345 \% num$
- $num$  を  $1$  から  $12345$  まで1ずつ増やしながらか変化させ,  $12345 \% num$  の計算結果を調べ, 結果が $0$ のときはその値を約数として表示する!



```
int num = 1;
if((12345 % num) == 0)
{
    println(num);
}
num++;
if((12345 % num) == 0)
{
    println(num);
}
num++;
if((12345 % num) == 0)
{
    println(num);
}
num++;
if((12345 % num) == 0)
{
    println(num);
}
num++;
```



```
int num = 1;
while(num <= 12345)
{
    if((12345 % num) == 0)
    {
        println(num);
    }
    num++;
}
```

# 正の約数の数を求める



(Q) 12345の正の約数の数を求めるには？

- 正の約数とは、ある1以上の自然数に対して割り切ることができる1以上の自然数
- 1, 3, 5, 15, 823, 2469, 4115, 12345 なので8個と計算できたらOK!
- 約数が表示できた時の数を数えればOK!

# 正の約数の数を求める



```
int num = 1;
int count = 0;
while(num <= 12345)
{
    if((12345 % num) == 0)
    {
        // 12345をnumで割った余りが
        // 0だったらcountを増やす
        count++;
    }
    num++;
}
println("正の約数の数は" + count);
```

# 繰り返しの他の書き方



```
for(初期化; 条件; 繰り返しの際の処理)
{
    繰り返し時に実行される色々な処理
}
```

```
int x = 0;
while(x <= 400)
{
    line(x, 0, x, 300);
    x = x + 10;
}
```

```
for(int x = 0; x <= 400; x += 10)
{
    line(x, 0, x, 300);
}
```

慣れると for が便利です



# コインを1万回投げる



(Q) コインを1万回投げた時に、表と裏になる回数はそれぞれ何回か？ その差は？

- random を利用して 0 か 1 を出力
- その 0 または 1 の回数を数える



# コイン

```
int countHead = 0;
int countTail = 0;

for(int i = 0; i < 10000; i++)
{
    if((int)random(2) == 0)
    {
        countHead++;
    }
    else
    {
        countTail++;
    }
}

println("表の数は" + countHead + "回");
println("裏の数は" + countTail + "回");
```

# 予習問題



- 1234567 のすべての正の約数を表示する
- 1234567 の正の約数の数を表示する