Today

- Close your laptops (only paper and cell phone are allowed)

- Chapter 4
  - Repetitions

CS 101
Repetition

Chapter 4

Remember our Algorithm to clean dishes…

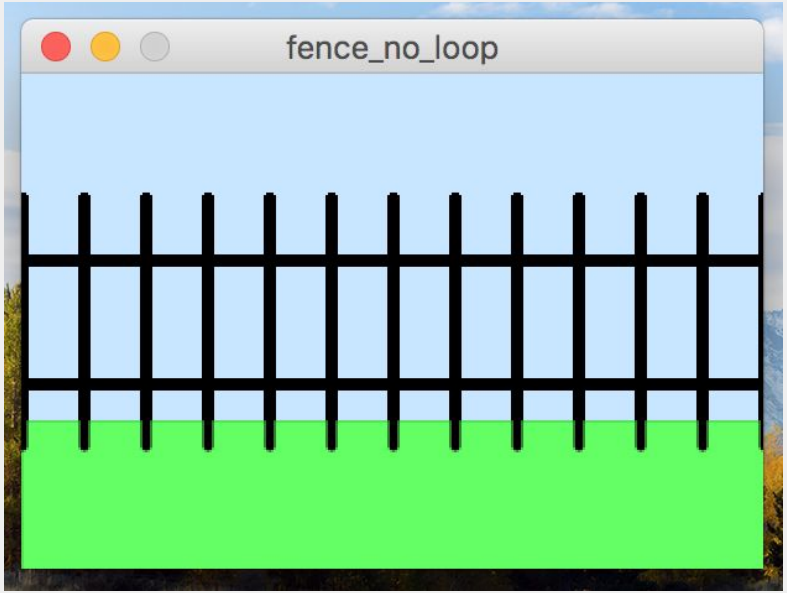
1) Turn on the water to the left sink
2) Grab the sponge
3) Put some soap on the sponge
4) As long as dishes remain in the right sink
    a) Grab a dish from the right sink
    b) Scrub it well
    c) Place it in the left sink & rinse
5) Grab the drying rag
6) As long as dishes remaining in the left sink
    a) Grab dish from the left sink
    b) Dry it well
    c) Put it in the proper cabinet

Which steps are repetitive in this algorithm?

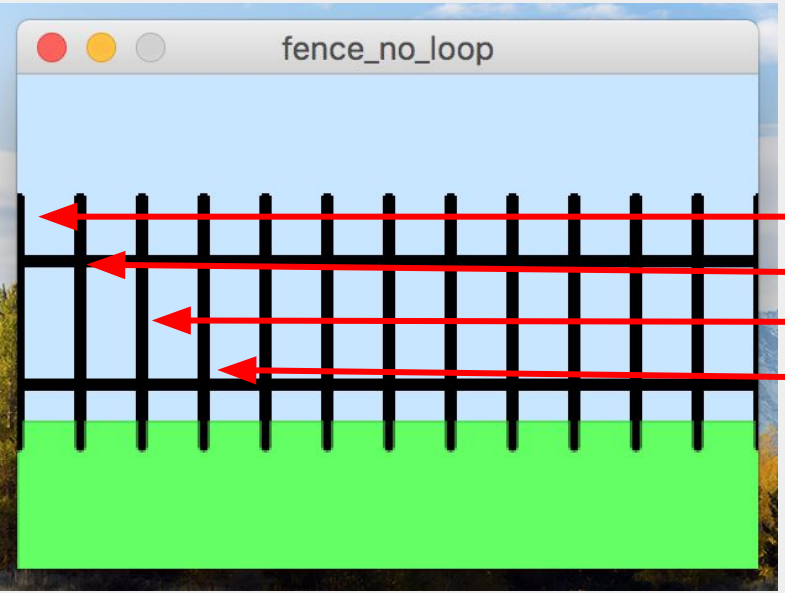
1) Turn on the water to the left sink
2) Grab the sponge
3) Put some soap on the sponge
4) As long as dishes remain in the right sink
    a) Grab a dish from the right sink
    b) Scrub it well
    c) Place it in the left sink & rinse
5) Grab the drying rag
6) As long as dishes remaining in the left sink
    a) Grab dish from the left sink
    b) Dry it well
    c) Put it in the proper cabinet

Drawing a Picture

- Given what we know about processing so far, how could we draw this picture? Specifically
  - What kinds of shapes?
  - How many shapes?
  - What colors?
  - Other functions needed?



Drawing a Picture

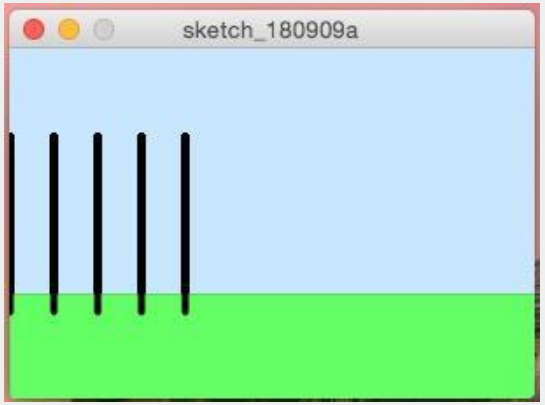



```
void setup() {
  size(300, 200);
  background(200, 230, 255);
}

void draw() {
  strokeWeight(0);
  fill(100, 255, 100);
  rect(0, 140, 300, 100);
  strokeWeight(4);
  line(0, 50, 0, 150);
  line(25, 50, 25, 150);
  line(50, 50, 50, 150);
  line(75, 50, 75, 150);
  line(100, 50, 100, 150);
  line(125, 50, 125, 150);
  line(150, 50, 150, 150);
  line(175, 50, 175, 150);
  line(200, 50, 200, 150);
  line(225, 50, 225, 150);
  line(250, 50, 250, 150);
  line(275, 50, 275, 150);
  line(300, 50, 300, 150);
  line(0, 75, 300, 75);
  line(0, 125, 300, 125);
}
```

Drawing a Picture

- Let's just draw 5 of the vertical lines.
- Do you see a pattern?

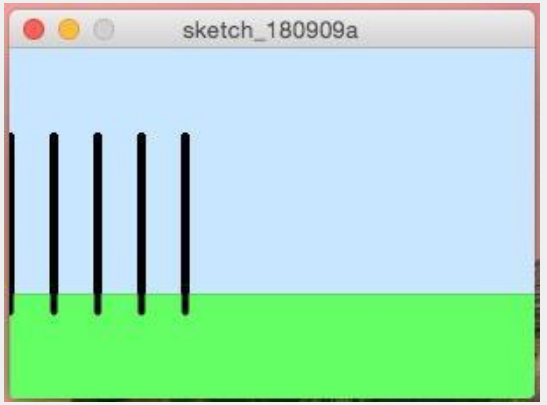
```
void setup() {
  size(300, 200);
  background(200, 230, 255);
}

void draw() {
  strokeWeight(0);
  fill(100, 255, 100);
  rect(0, 140, 300, 100);
  strokeWeight(4);
  line(0,   50, 0,   150);
  line(25,  50, 25,  150);
  line(50,  50, 50,  150);
  line(75,  50, 75,  150);
  line(100, 50, 100, 150);
}
```



Drawing a Picture

- Let's just draw 5 of the vertical lines.
- Do you see a pattern?
- We'd like to say,
  - draw the line 5 times
  - each time, add 25 more to the x-coordinate than last time

```
void setup() {
  size(300, 200);
  background(200, 230, 255);
}

void draw() {
  strokeWeight(0);
  fill(100, 255, 100);
  rect(0, 140, 300, 100);
  strokeWeight(4);
  line(0,   50, 0,   150);
  line(25,  50, 25,  150);
  line(50,  50, 50,  150);
  line(75,  50, 75,  150);
  line(100, 50, 100, 150);
}
```



Repetitions

Repetitions in programming are loops

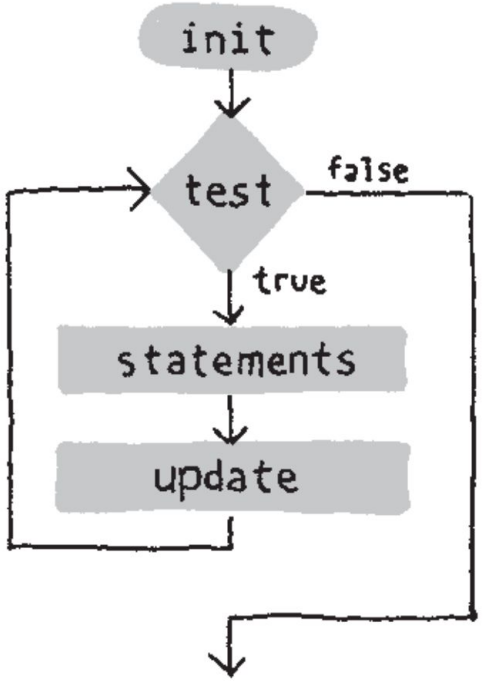
We have:

- **For** loops
- **While** loops
- **Do while** loops
- **For each** loops
- ….

In Processing -> For Loops

- **For Loops** are perfect to use when a step needs to be repeated in code
- A **for-loop** is a special feature in processing (referred to as a **code-construct**) that allows you to repeat lines of code a specified number of times
- This can be used to repeatedly draw the fence-posts
- But first, let’s look at the basics

The “Flow” of a for-loop

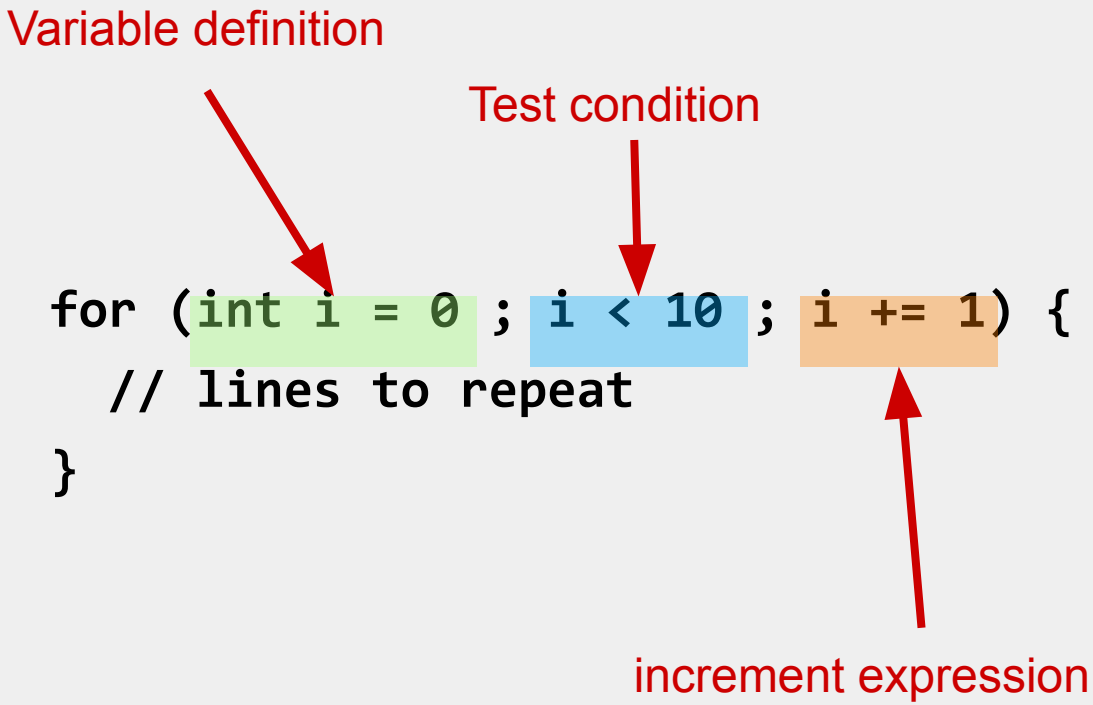



```
for (init; test; update) {
  statements
}
```

For loop Template

```
for (          ;          ;          ) {


}
```

Anything written between curly brackets will be **repeated** as many times as you want!


Variable definition
Test condition
for (int i = 0 ; i < 10 ; i += 1) {
// lines to repeat
}
increment expression

Equivalent to plain English logic

```
for ( each dish, and as long as dishes remain in the right sink ) {
```

a) *Grab a dish from the right sink*
b) *Scrub it well*
c) *Place it in the left sink & rinse*

```
}
```

Template for counting (by one)

```
for (int i = START; i <= END ; i += 1) {

// use variable i


}
```

for loop - template for counting

- Let’s count from **1** to **5**

```
for (int i = 1 ; i <= 5 ; i += 1) {

    println(i);

}
```

Loop table

```
for (int i = 1 ; i <= 5 ; i += 1) {
        println(i);
}
```

iteration	Value i	Condition	Output

equivalent code

```
for (int i = 1 ; i <= 5 ; i += 1) {

    println(i);

}
```

```
println(1);
println(2);
println(3);
println(4);
println(5);
```

Question 1. Write the loop table for the following loop (~ 3')

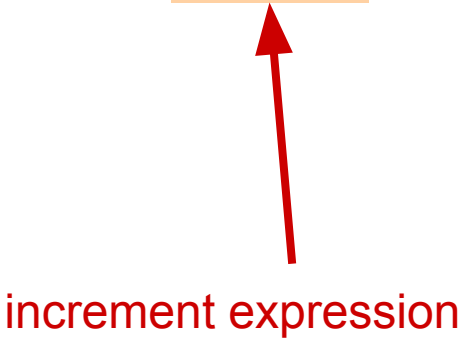
```
for (int i = 3 ; i <= 8 ; i += 1) {

    println(i);

}
```

Question 2.

Write a for loop that prints the numbers from 13 to 584

Template for counting (by one)

```
for (int i = START; i <= END ; i += 1) {

// use variable i


}
```



Template for counting (by anything)

```
for (int i = START; i <= END; i += INCREMENT)
{

// use variable i



}
```

for loop - template for counting

- Let’s count starting from **0** to **100** in increments of 5

```
for (int i = 0 ; i <= 100 ; i += 5) {

    println(i);

}
```

Question 3. Write the loop table for the following loop

```
for (int i = 0 ; i <= 24 ; i += 3) {

    println(i);

}
```

Question 4.

Write a for loop that prints the numbers from 5 to 600 in increments of 8

Question 5.

Write a program that prints the multiplication table

The loop test

- There is actually a lot of flexibility in the condition (or ***test***) in a for-loop
- It can compare any two values, not just the loop variable
- You can also use many types of comparisons
  - `>` greater-than
  - `<` less-than
  - `>=` greater-than or equal-to
  - `<=` less-than or equal-to
  - `==` exactly equal-to
  - `!=` not equal-to

Question 5.

Write a program that prints the numbers from 80 to 2 in decrements of 3