Review

In the beginning…

```
int x, y; // integer: no decimal point value.
boolean xIncreasing, yIncreasing; // true or false
int w = 500; // store a number in a variable
int h = 250;
```

In the beginning…

```
int x, y;
boolean xIncreasing, yIncreasing;
int w = 500;
int h = 250;

void setup() { // runs once
    background(100, 200, 300); // R G B
    size(w, h); // make the window a specific size
}

void draw() { // runs again and again in a loop

    // do stuff
}
```

```
int x, y;
boolean xIncreasing, yIncreasing;
int w = 500;
int h = 250;

void setup() {
    background(100, 200, 300);
    size(w, h);
}

void draw() { // runs again and again in a loop

    ellipse(x, y, 10, 10);

}
```

Simplest bouncing:

```
void draw() { // runs again and again

    ellipse(x, y, 10, 10); // draws a circle

    // now we need to change the values stored in ‘x’ and ‘y’
    // in order to make the ellipse move.
    if (xIncreasing) x++; // move right
    else x--; // move left

    if (yIncreasing) y++; // move up
    else y--; // move down

    if (x < 0) xIncreasing = true; // we've hit the left edge!
    if (x > w) xIncreasing = false; // right edge

    if (y < 0) yIncreasing = true; // top
    if (y > h) yIncreasing = false; // bottom
}
```

Simplest bouncing:

```
void draw() { // runs again and again

    ellipse(x, y, 10, 10); // draws a circle

    // now we need to change the values stored in ‘x’ and ‘y’
    // in order to make the ellipse move.
    if (xIncreasing) x++; // move right
    else x--; // move left

    if (yIncreasing) y++; // move up
    else y--; // move down

    if (x < 0) xIncreasing = true; // Let’s take a look
    if (x > w) xIncreasing = false; // at these.

    if (y < 0) yIncreasing = true; // top
    if (y > h) yIncreasing = false; // bottom
}
```

Simplest bouncing:

```
if (x < 0) xIncreasing = true;
if (x > w) xIncreasing = false;
```

Making functions

```
if (x < 0) xIncreasing = true; // we've hit the left edge!
if (x > w) xIncreasing = false; // right edge
```

```
if (shouldBounceX()) {
    bounceX();
}
```

i.e. ("If we should bounce, let's bounce")

```
if (shouldBounceX()) {
    bounceX();
}
```

Making functions

```
if (shouldBounceX()) {
    bounceX();
}
              i.e. ("If we should bounce, let's bounce")
```

Making functions

```
if (shouldBounceX()) {
    bounceX();
}
                i.e. ("If we should bounce, let's bounce")
```

```
_______ shouldBounceX() {

}
```

Making functions

```
if (shouldBounceX()) {
    bounceX();
}
            i.e. (“If we should bounce, let’s bounce”)
```

```
boolean shouldBounceX() {
   // we will return either ‘true’ or ‘false’
}
```

Making functions

```
if (shouldBounceX()) {
    bounceX();
}
             i.e. ("If we should bounce, let's bounce")
```

```
boolean shouldBounceX() {
    return // we will return either 'true' or 'false'
}
        Asking a question, getting a true/false answer
```

Making functions

```
if (shouldBounceX()) {
    bounceX();
}
              i.e. ("If we should bounce, let's bounce")
```

```
boolean shouldBounceX() {
    return true;
}
        Asking a question, getting a true/false answer
```

Making functions

```
if (shouldBounceX()) {
    bounceX();
}
              i.e. ("If we should bounce, let's bounce")
```

```
boolean shouldBounceX() {
    return false;
}
        Asking a question, getting a true/false answer
```

Making functions

```
if (shouldBounceX()) {
    bounceX();
}
               i.e. (“If we should bounce, let’s bounce”)
```

```
boolean shouldBounceX() {
    return ((location.x > width) || (location.x < 0));
}
        Asking a question, getting a true/false answer
```

Making functions

```
if (shouldBounceX()) {
    bounceX();
}
                i.e. ("If we should bounce, let's bounce")
```

Making functions

```
if (shouldBounceX()) {
    bounceX();
}
              i.e. ("If we should bounce, let's bounce")
```

```
____ bounceX() {


}

                                    ...let's bounce
```

Making functions

```
if (shouldBounceX()) {
    bounceX();
}
              i.e. ("If we should bounce, let's bounce")
```

```
void bounceX() {


}
                                   ...let's bounce
```

Making functions

```
if (shouldBounceX()) {
    bounceX();
}
              i.e. (“If we should bounce, let’s bounce”)
```

```
void bounceX() {

    velocity.x = velocity.x * -1;
}
                                  ...let’s bounce
```

Making functions

```
if (shouldBounceX()) {
    bounceX();
}
              i.e. ("If we should bounce, let's bounce")
```

```
boolean shouldBounceX() {
    return ((location.x > width) || (location.x < 0));
}
        Asking a question, getting a true/false answer
```

```
void bounceX() {

    velocity.x = velocity.x * -1;
}
                                    ...let's bounce
```

Arrays

and so on

```
int[] myArray = {0,1,2,3};
```

```
int[] myArray = {0,1,2,3};
float[] myFloatArray = {1.1, 3.14, 2.87, 5.0};
```

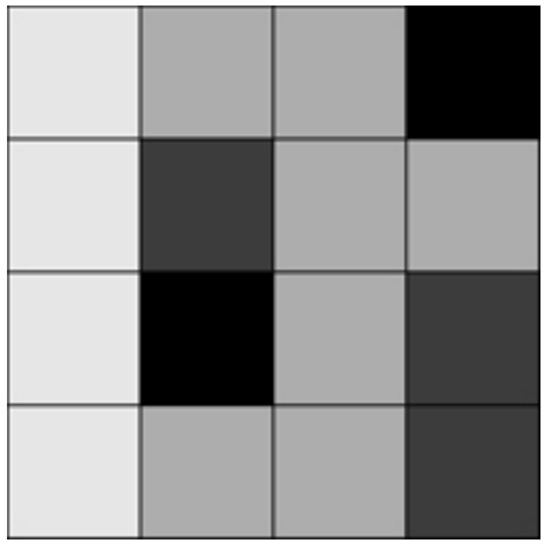
```
int[] myArray = {0,1,2,3};
float[] myFloatArray = {1.1, 3.14, 2.87, 5.0};

Ball[] bouncies = {ball0, ball1, ball2};
```

```
int[][] myArray = { {0,1,2,3}, {3,2,1,0}, {3,5,6,1}, {3,8,3,4} };
```

```java
int[][] myArray = { {0, 1, 2, 3},
                    {3, 2, 1, 0},
                    {3, 5, 6, 1},
                    {3, 8, 3, 4}  };
```

```
int[][] myArray = { {236, 189, 189,   0},
                    {236,  80, 189, 189},
                    {236,   0, 189,  80},
                    {236, 189, 189,  80}  };
```



https://processing.org/reference/Array.html

```java
int[] myArray = new int[10];
for (int i = 0; i < myArray.length; i++) {
    myArray[i] = 0;
}
```

```java
int cols = 10;
int rows = 10;
int[][] myArray = new int[cols][rows];

// Two nested loops allow us to visit every spot in a 2D array.
// For every column I, visit every row J.
for (int i = 0; i < cols; i++) {
    for (int j = 0; j < rows; j++) {
        myArray[i][j] = 0;
    }
}
```

https://processing.org/tutorials/2darray/