Lists with Arrays

We already learned how to create and use variables. Sometimes, however, we'd prefer to use a list of variables—like a list of users, or a list of buttons in a UI (User Interface).

Often, if we need to perform similar tasks to many items, it is better to group them together. This will let us use a **loop** to perform repetitive tasks to every item in the list, without having to repeat the code over and over.

Array Example

Let's say we want to animate 10 bouncing balls. We already had similar code for one ball earlier:

```
float circleY = 0;
float circleSpeed = 1;

void draw() {
  background(200);

  circle(50, circleY, 10);

  circleY += circleSpeed;

  if (circleY < 0 || circleY > height) {
    circleSpeed = -circleSpeed;
  }
}
```

It would be a real pain and a mess to repeat nearly identical code 10 times. Instead, we can make a list (**array**) of the values we need, and use a **for loop** to repeat the code for each item.

This code isn't much longer than the code for a single ball, but it animates 10 instead of just 1!
(Example from https://happycoding.io/tutorials/processing/arrays)

```
float circleY[] = {0, 10, 20, 30, 40, 50, 60, 70, 80, 90};
float circleSpeed[] = {1, 1, 1, 1, 1, 1, 1, 1, 1, 1};

void draw() {
  background(200);

  for(int i = 0; i < 10; i++){
    circle(i*10+5, circleY[i], 10);
    circleY[i] += circleSpeed[i];
    if(circleY[i] < 0 || circleY[i] > height){
      circleSpeed[i] *= -1 ;
    }
  }
}
```

How to Use Arrays

Creating an **array** is similar to creating a variable, we need to declare its type (the list has to all be the same type), and we give it a name.

`String userNames[] = {Adam, Boruch, Chaim};`

`int studentGrades[] = {98, 87, 79, 91};`

The square brackets tell Java that we are creating an **array**. The values for the **array** go inside of curly brackets, separated by commas.

Each item in the array gets an **index** number for its position in the list. The index (unfortunately) starts at 0, not at 1.

To access the variables in our list, we use the name, followed by its **index**, for example:

- `userNames[0]` is “Adam”
- `userNames[1]` is “Boruch”
- `userNames[2]` is "Chaim

Adding Values to Array

Like regular variables, we can name the variable (also called **declaring** a variable) and assign its value later (also called **initializing** a variable). The only difference, is that we need to specify how long the **array** will be. This is because **arrays** have a fixed size.

In our example, we can use a **for loop** to initialize the variables, because they follow a regular pattern.

```
float circleY[] = new float[10];
float circleSpeed[] = new float[10];

void setup() {
  for (int i = 0; i < circleY.length; i++) {
    circleY[i] = (i*1) * 10;
    circleSpeed[i] = 1;
  }
}
```

Notice, that after creating the **array**, we specify its size with the command `new float[10]`, which reads "new float array of size 10.

Things to Try

- Modify the bouncing ball program to show 50 balls, instead of 10.
- Make the Y position of the balls random (using the `random()` method), instead of a set pattern.
- Make the X position a variable, and change the program to have both an Xspeed and a Yspeed. Make the balls bounce off all 4 walls.