Lesson 10: Variables, and Animation

Lesson aim:

We are going to be introduced to a foundation of modern programming: the variable. Variables are bits of named data. We need to get familiar with variables as we will use them all the time.

Why:

Apart from being the foundation of most popular modern programming languages (Java etc.) variables are needed for most of the interesting things we want to do - e.g. animation, storing scores etc.

Animation Example:

```
int ballHozPosition;   //← make new variable called ballHozPosition
void setup(){
 size(600,600);
 ballHozPosition = 50;  //← give the variable an initial value (50)
}
void draw(){
  ellipse(ballHozPosition,300,100,100);
  ballHozPosition = ballHozPosition + 1;   //← update the value in
}                                          // ballHozPosition.
```

Run this, what happens? _____

Refresh tasks, change the above program so that:

1. Change the colour of the ball.
2. Change it so that there is a rectangle moving, rather than a ball.
3. Speed up the ball. (Try changing the 1 in "`ballHozPosition + 1;`")
4. Make the ball go left rather than right by:
    a. Changing its start position to start on the right side of the window
    b. Make the ball move left instead of right, (**hint:** how do you move a shape's position from left to right? What happens to the horizontal position number?)
5. Change the program so that we can only see one ball at a time.

How the examples works with variables

A variable is a bit of stored data, with a name that you have given it. Eg: `ballHozPosition`

So instead of being the same number all of the time - you can change the data in a variable.
In the above example (the first one, written on the page):

- We make a new variable with a name of ballHozPosition
    - `int ballHozPosition;`
- Give ballHozPosition a starting value of 50.
    - `ballHozPosition = 50;`

- Then each time the draw code block is run (30 times a second) it's given a new value that is its old value plus 1. So the number in `ballHozPosition` keeps getting bigger and therefore the ball with keep moving right.
    - `ballHozPosition = ballHozPosition + 1;`

Anatomy of a variable - what makes a variable?

```
int aNumber = 10;
```

- **a name:** eg "`aNumber`" you can call it almost anything you like - but no spaces!
- **a type:** eg `int`. Tell the computer what data it is: `int` means it is a whole number
- **a value**: eg "10". Depends on the type. For an `int`: a number, could be 1, -6, 1000...

Tasks:

1. Save the old program - start the program name with "**Lesson10**".
2. Make a new program with a new variable in it, name the variable whatever you want. The variable name should tell you what the variable is for ("ballHozPosition" stores the horizontal position of the ball, for example).
3. Use that variable to make a ball move **down** the screen. So the vertical position will be getting bigger as time goes on.
4. Change the above program so that the ball starts at the bottom of the screen and then move up the screen.

Having multiple variables:

You can have as many variables as you want, but they have to have **different names.**

Example:

```
int ballHozPosition;   //← make new variable called ballHozPosition
int ballVertPosition; //← make new variable called ballVertPosition
void setup(){
 size(600,600);
 ballHozPosition = 50;  //← give the variable an initial value (50)
 ballVertPosition = 550;  //← give variable an initial value (550)
}
void draw(){
  ellipse(ballHozPosition,ballVertPosition ,100,100);
  ballHozPosition = ballHozPosition + 1;    //← update the value
  ballVertPosition = ballVertPosition - 1;    //← update the value
}
```

Tasks:

1. Change the above program so both the balls go faster. Try and keep the speeds the same.

2. Change the above program so that the ball goes from top left to bottom right.
3. Make another change so a rectangle goes from bottom right to top left.
4. Make the ball change vertical direction twice as fast as the horizontal direction.
5. Speed up both the horizontal and vertical speeds.