Homework 5: The Simpson's Are Coming Home

Goal: In this assignment we practice with variables, declarations and assignment statements. These ideas help make computations dynamic in interesting ways.

Idea 1: A Variable

A *variable* is the name of a quantity that *varies* … get it? *Variable = able* to *vary*! Isn't computing obvious? The thing about a variable that varies is the *value* or number that it names. For example, if the variable `score` names the value 21, which happens because we wrote

```
score = 21;
```

and then later in the program, we write

```
score = 23;
```

the value of `score` – the number it names – has changed from what it was (21) to 23, that is, it has varied. Variables are important to dynamic programs, because values change all the time.

Idea 2: Declaring Variables

To understand our programs the Processing engine needs to know what variables we use in our program, so we *declare* them. Declarations come at the start of our program (before setup( )) by writing code such as

```
int score = 0;
```

This line of code tells Processing several things: (a) `score` is a variable that will be used in the program, (b) `score` will have integer values (whole numbers) because we used the abbreviation `int` for integer (there are other types of values), and (c) `score` starts with the value 0.

Idea 3: Changing Variables

We can use normal "calculator operations" to change the values of variables. For example, if score kept track of the score of a basketball game, we can record a basket with

```
score = score + 2;
```

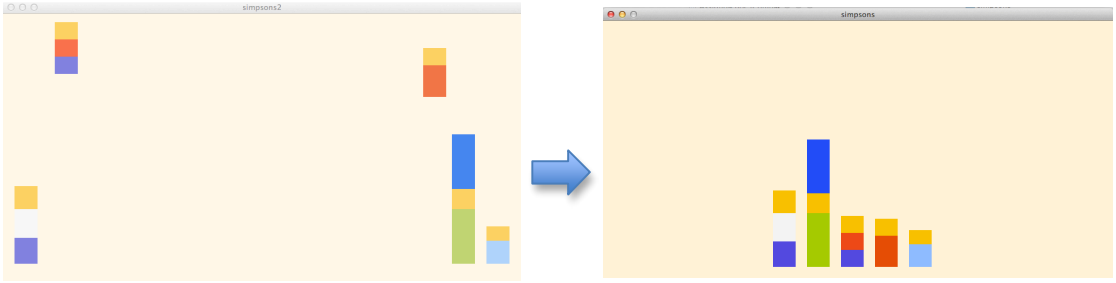
since a basket is worth two points. A free throw can be recorded with

```
score = score + 1;
```

because they are worth just one point each. We read that line of code as "the variable score becomes score plus 1," meaning its value increases by 1.
Using the three ideas we have just reviewed, we can make the Simpsons move into place.

Step 1: Declare for the Simpsons

Grab a copy of the posted code on the Calendar page for this assignment, placing it into a new Processing window. (You must use this code.) Our program will begin by placing the Simpsons in their daily positions (Homer at the Nuclear Power Plant, etc.) and then bring them together. To make that happen we need some variables, and that means declarations.



Begin by declaring six variables. Recall that variables are declared as the very first thing in the program. Also, these variables will all be integers, and their initial values are shown in the table. So, the first declaration is

```
int hom = -280;
```

This declares that hom will be an integer variable in the program and its value begins at -280.

Notice that Maggie is four letters long, because mag is a special (keyword) for Processing and we avoid using those. Also, Lisa, being especially talented, gets two variables, lis1 and lis2. Run the program (see no visible change) to be sure all is well.

Person	Variable	First Value
Homer	hom	-280
Marge	mar	420
Bart	bar	-330
Maggie	magg	300
Lisa	lis1	250
	lis2	-290

Step 2: Move To 5:00 PM Positions

The Simpsons are in different places in Springfield before coming home, so we will place them in those positions. We do this by adding the variable we just declared to the X or the Y (or maybe both) positions in the rectangles.

So, to position Homer left, which is horizontal motion, we add hom to the X coordinate of each rectangle that makes Homer. That would be

```
// Homer
  fill( 250, 193, 35);              //yellow
```

```
  rect( 300+hom, 300, 40, 40);
  fill( 245, 245, 245);
  rect( 300+hom, 340, 40, 50); //white
  fill( 89, 79, 217);
  rect( 300+hom, 390, 40, 45);  //blue
```

as shown in the highlights. Make these changes and run the program. Homer should be at the left side of the screen.

Next, change Marge and Maggie. They're like Homer, because their offset is also horizontal. That is, it applies to the X coordinate only. Run that program and see that they are on the right and all is good.

Bart and Lisa are a little different because their variables apply to both coordinates, X and Y. Bart adds his variable `bar` to both. Lisa adds `lis1` to X and `lis2` to Y. Run the result to see that the Simpsons are at "5:00."

Step 3: Going Towards Home

Now to move the Simpsons toward home. We do this in a clever way. Remember that a dynamic program like ours keeps redrawing over and over again. We will use that fact to move the characters: In each case, we change the variable so it goes closer to 0, which is their home. In Homer's case, for example, `hom` starts out at -280, so we just add 1 to it before we draw it.

```
…
//Homer
hom = hom + 1;
fill( 250, 193, 35);  //yellow
rect( 300+hom, 300, 40, 40);
…
```

Make this change, and run the program. Homer should move right. (He won't stop; we'll do that next.)

Marge and Maggie are different, because their variables start out as positive numbers, and so for them to get closer to 0, we must subtract 1 before they are drawn, as in

```
…
// Marge
mar = mar - 1;
fill(167,203,60);      // Olive
rect(360+mar,340,40,95);
…
```

Change Maggie similarly. They will both move left without stopping, so run the program and watch it happen. Bart and Lisa use the same ideas. Please change their variables before drawing them.

Step 4: STOP!

All of the Simpsons go towards home now, but they don't stop to sit on the couch. To make them stop, we use one of two special functions: min(a, b) and max(a, b). These functions get two numbers, a and b, and pick the smaller (min) or larger (max) as a result.

So, to stop Homer, we use the following logic. When hom has reached 0, we can prevent it from going further by always correcting it to the min(0, hom), as in

```
//Homer
hom = hom + 1;
hom = min(0, hom);
```

Notice how this works. Suppose hom was 0 the last time Homer was drawn, then adding 1 will make hom have the value 1. This means that the next line is equivalent to min(0,1), which will be 0. So we reset hom back to 0 again. That stops him, and makes his figure be redrawn in the exact same place. That is, at home.

We use min( ) because Homer's variable counted up. Marge and Maggie's variables are counting down, so we need to do the opposite, as in

```
//Marge
mar = mar - 1;
mar = max(0, mar);
```

This has the opposite result: When mar is already 0, subtracting 1 makes it negative, and the next line is equivalent to max(0, -1) which is 0. Clever, no?

Finish Maggie, Bart and Lisa. Be careful to note how the values are changing. Run the program and watch them come home.

Wrap Up.

You've programmed Processing using variables. You used the fact that the draw( ) routine refreshes the image several times per second. Each time it runs draw( ) it repeats the "increasing" operation such as hom = hom + 1; This causes the variable's value to grow. By using that variable to increase the horizontal position of shapes, you could make Homer move across the screen; you stopped him by resetting hom whenever it passed 0 using min( ).

Submit your .pde file into the class drop box.