Lab 06: A "Pint's a Pound The World Around"

Goal: Write a few Processing functions to return values, then listen to Michelle describe what is happening in your code.

Measuring Wine

Thanks to BurtonMacKenzie.com for the following nice explanation from a conjecture of a famous computer scientist, Donald Knuth:

Knuth [1] states the first known appearance of pure binary notation was about 1605 with Thomas Harriot [2], but he suggests another intriguing idea. He lists the common units of liquid measure used by English Wine Merchants as late as the 13th century. Specifically,

2 gills = 1 chopin
2 chopins = 1 pint
2 pints = 1 quart
2 quarts = 1 pottle
2 pottles = 1 gallon
2 gallons = 1 peck
2 pecks = 1 demibushel
2 demibushels = 1 bushel (or firkin)
2 firkins = 1 kilderkin
2 kilderkins = 1 barrel
2 barrels = 1 hogshead
2 hogsheads = 1 pipe
2 pipes = 1 tun

This is effectively a *binary* counting system using a 14 bit word. Each measure, twice as large as the last, has its own name. If you represent each named unit as a single bit and combine them all, you get a binary number. For instance, if I had 1 hogshead, a firkin, a gallon, and a pint of wine, I could write the measured amount of wine as **00100100100100** gills, where "gills" is the LSB (least significant bit) and "tuns" is the MSB (most significant bit).

Pint Size Programs

Our only use of the text above is to introduce odd words like gill and pottle; we won't be doing binary with these. The first task is to write result-returning functions. The first is:

```
float gill ( ) {
   return 118.294;          //weight of a gill in metric (grams)
}
```

This function returns the number of grams in a gill.

Write similar functions to return the weight of each unit, but do it so you calculate the result by adding the next smaller measure to itself, as in `return gill() + gill();` to compute a chopin. We want functions for:

```
chopin( )
pint( )
quart( )
pottle( )
gallon( )
```

Check your work after writing each function by using a `draw( )` block that prints out the result with the "print line" function `println( )`. So, after the first program, write

```
void draw( ) {
  println( gill() );
}
```

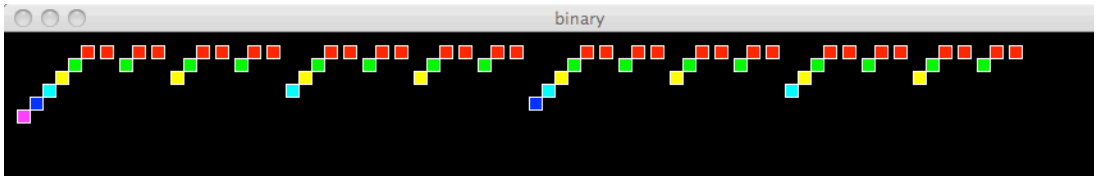
You will change "`gill()`" in the `println()` function after completing each function.

How Are The Functions Executed? Michelle Explains

Functions like yours have been modified so that they print out a little square of color:

gill is red and displayed 10 units down
chopin is green and displayed 20 units down
pint is yellow and displayed 30 units down
quart is aqua and displayed 40 units down
pottle is blue and displayed 50 units down
gallon is purple and displayed 60 units down

With these assignments, and each function changed to display their color to the right of the function that calls them, calling `gallon( )` we have



Time →

Time increases to the right. So, when `gallon( )` is called it displays it's purple square; then it computes its weight by calling `pottle( )` for the first time. That suspends `gallon( )` for the moment, so that `pottle( )` can be computed. `pottle( )` prints it's blue square; then it computes its weight by calling `quart( )` for the first time, which suspends `pottle( )` for the moment. `quart( )` prints it's aqua square, and then computes its weight by calling … you get the idea. We end up calling `gill( )` 32 times. After each one, `gill( )` returns to `chopin( )`, the function that called it. When it has called `gill( )` a second time and has its result, it returns its value to `pint( )` which has been suspended waiting for the result. And so forth all the way back to `gallon( )`. It is a terrific exercise to print squares and reproduce this diagram.