Announcements

- After Image Survey – it's part of our commitment to the AP experiment … your help is REALLY appreciated

Writing Programs

- Naturally, programs are given sequentially, the declarations at the top
- Braces { } are statement groupers … they make a sequence of statements into one thing, like the "true clause of an If-statement"
- All statements must end with a semicolon EXCEPT the grouping braces … they don't end with a semicolon (OK, it's a rare inconsistency about computer languages!)
- Generally white space doesn't matter; be neat!

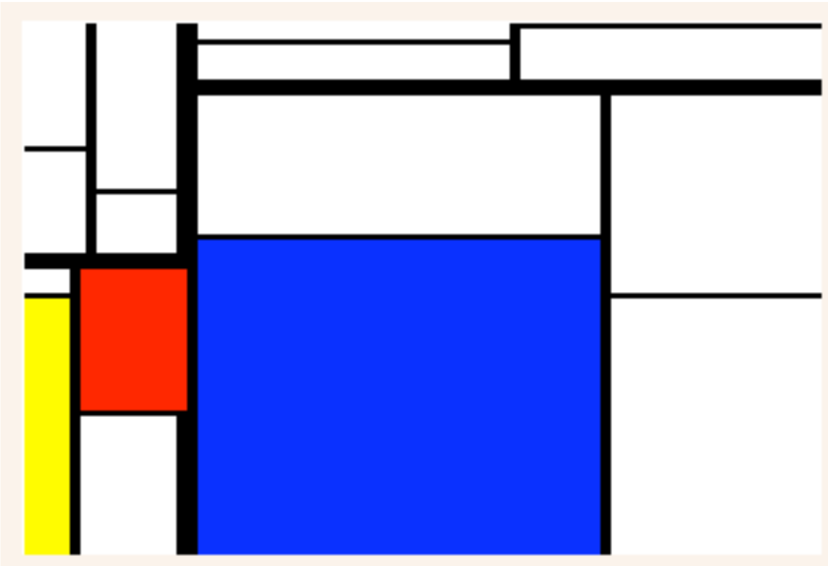
Program Execution

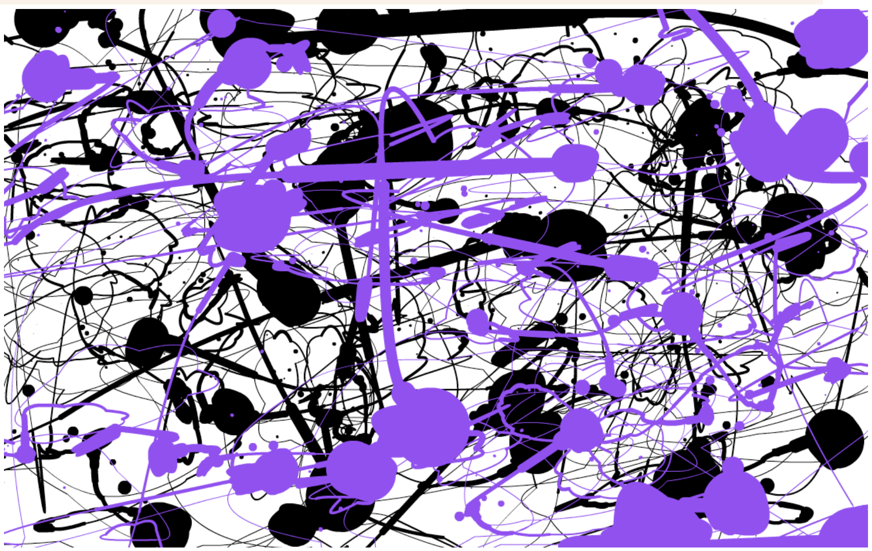
- Keep in mind how a program executes

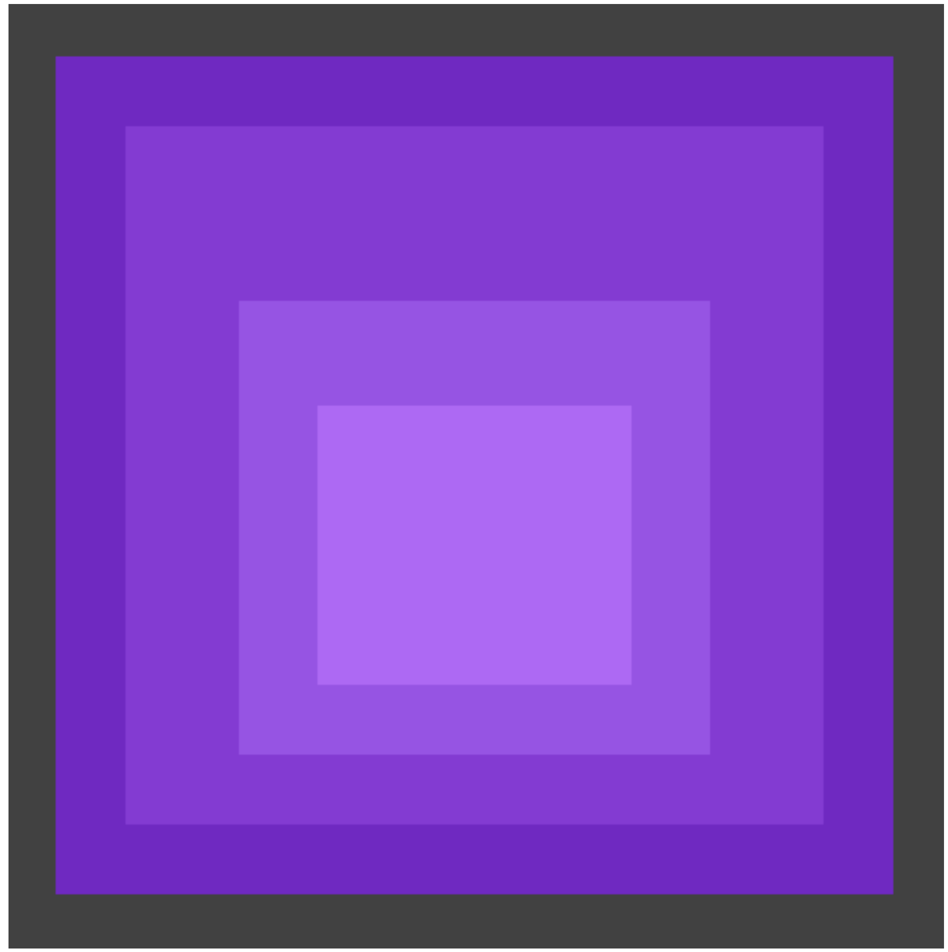
```
int next=1;

void setup( ) {
  size(100,100);
  fill(255, 0,0);
}

void draw( ){
  background(0);
  rect(mouseX, mouseY, 25, 25)
}

void mousePressed( ){
  if (next == 1) {
    fill(0, 0, 255);  // go to blue
  } else {
    fill(255,0,0);   // go to red
  }
  next=1-next;
}
```

Art Programs Raise Deep Questions

Mondrian, Pollack, Albers are stars …







Adding some light to computing ….

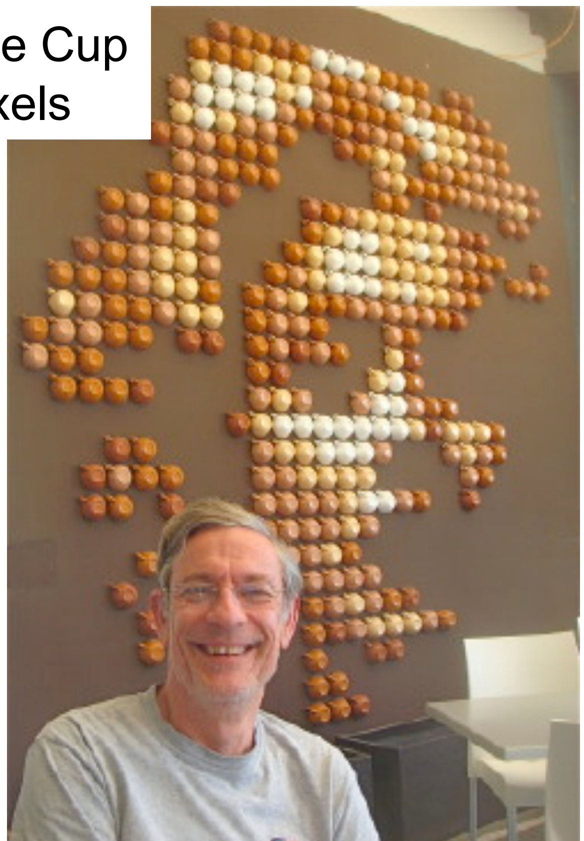
Bits of Color

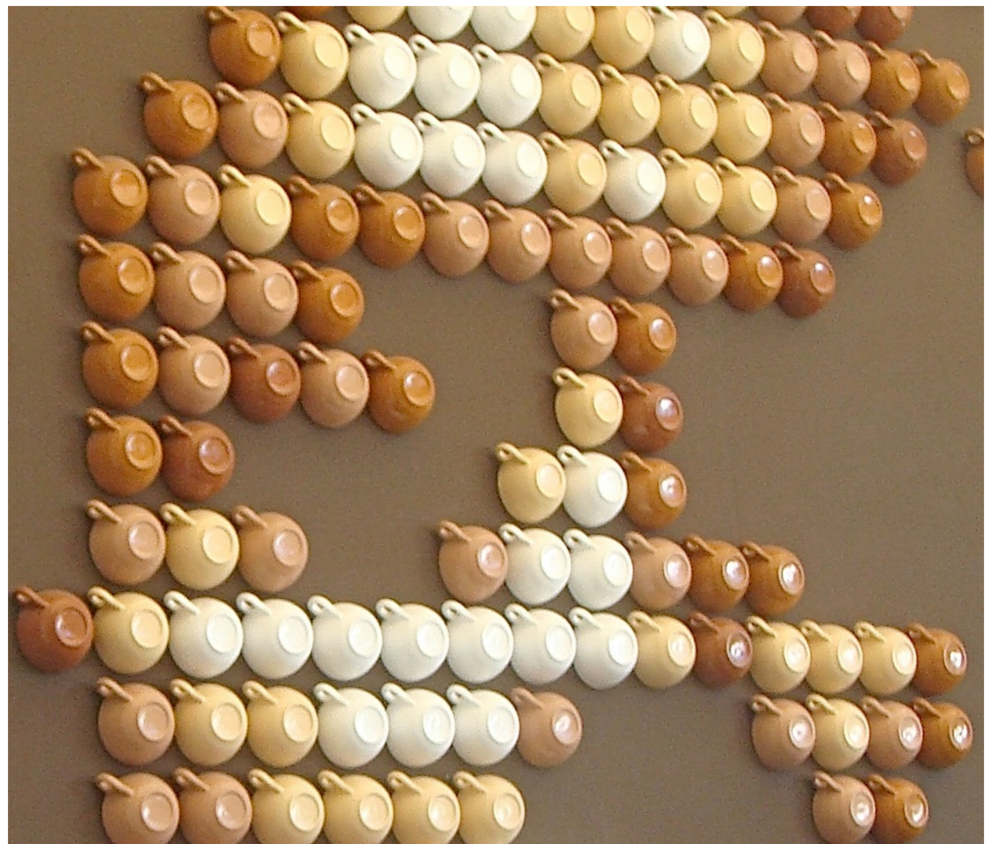
Lawrence Snyder
University of Washington, Seattle

Return To RGB

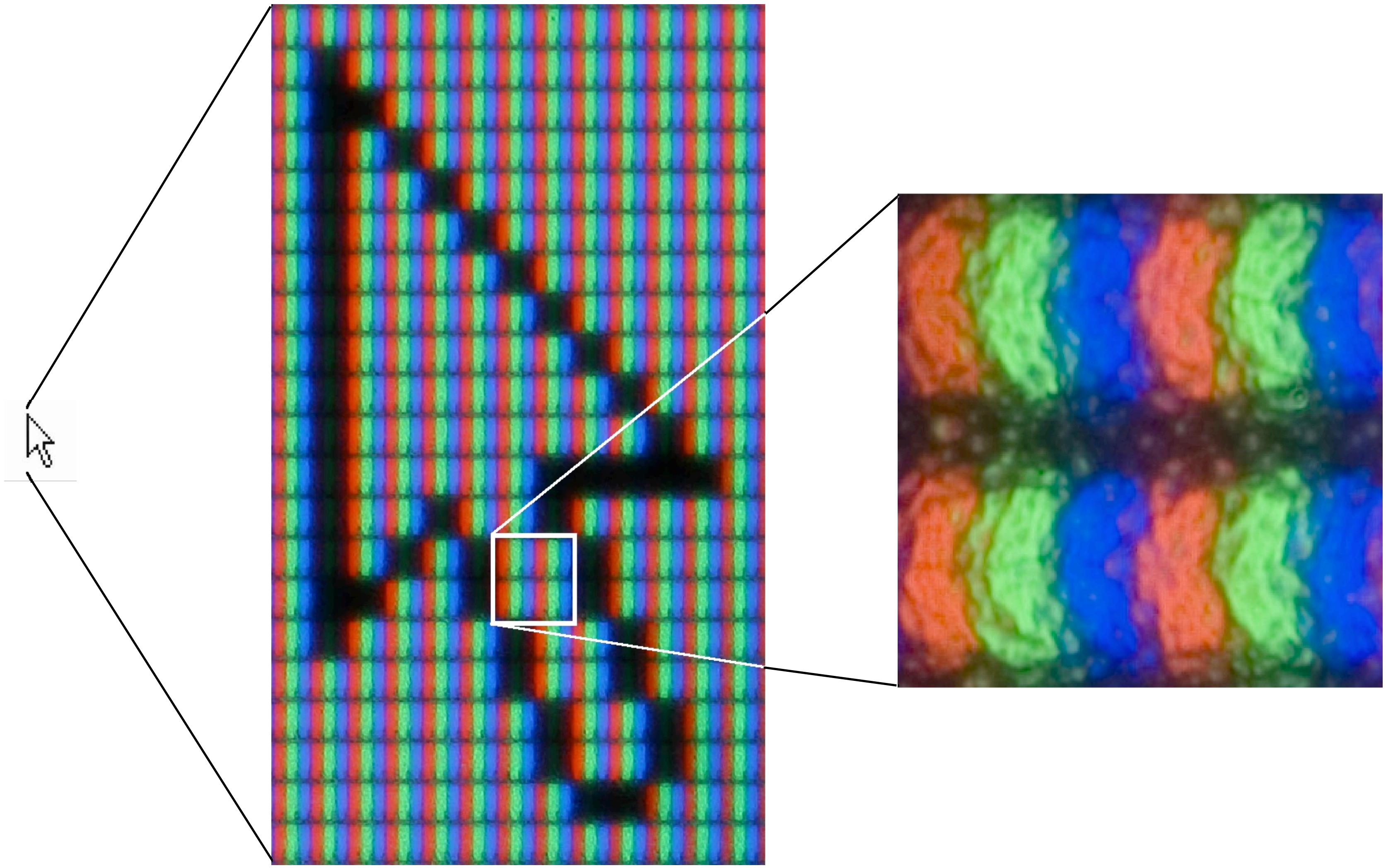
- Recall that the screen (and other video displays) use red-green-blue lights, arranged in an array of picture elements, or *pixels*

Coffee Cup Pixels



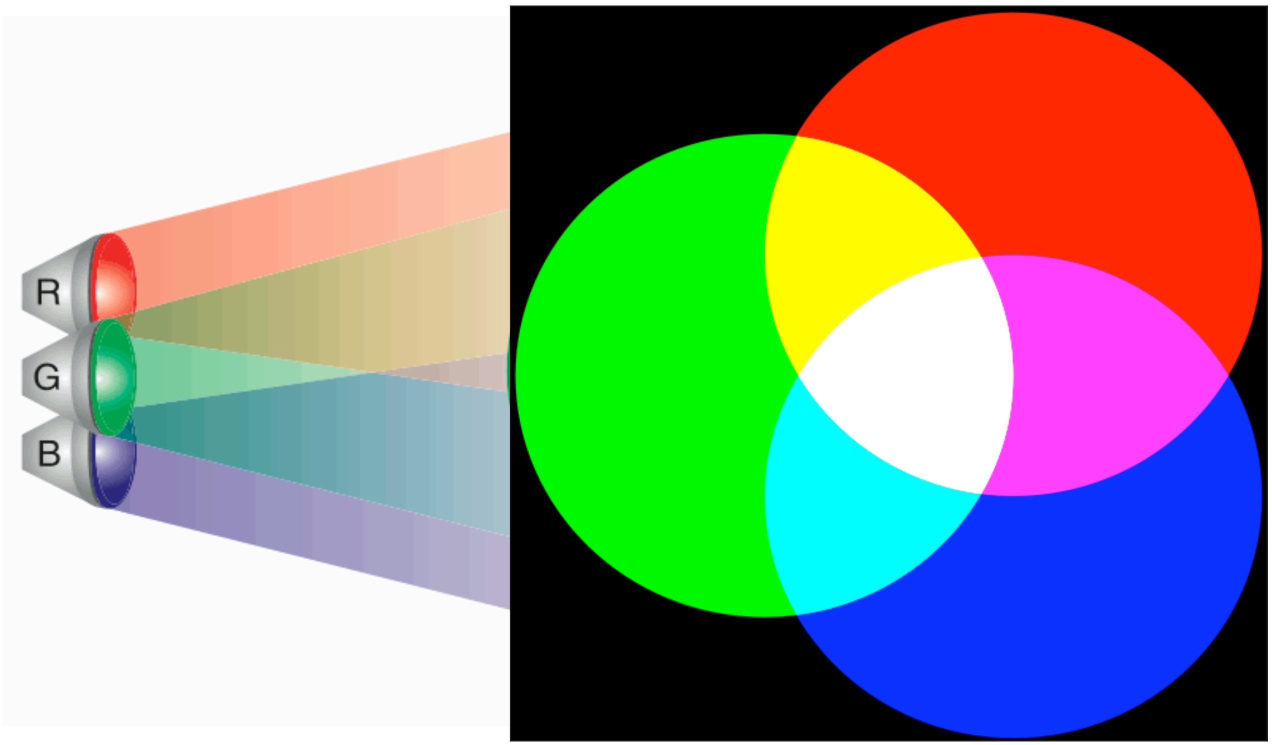


Actual Pixels From TFT LCD Display



Combining Colored Light

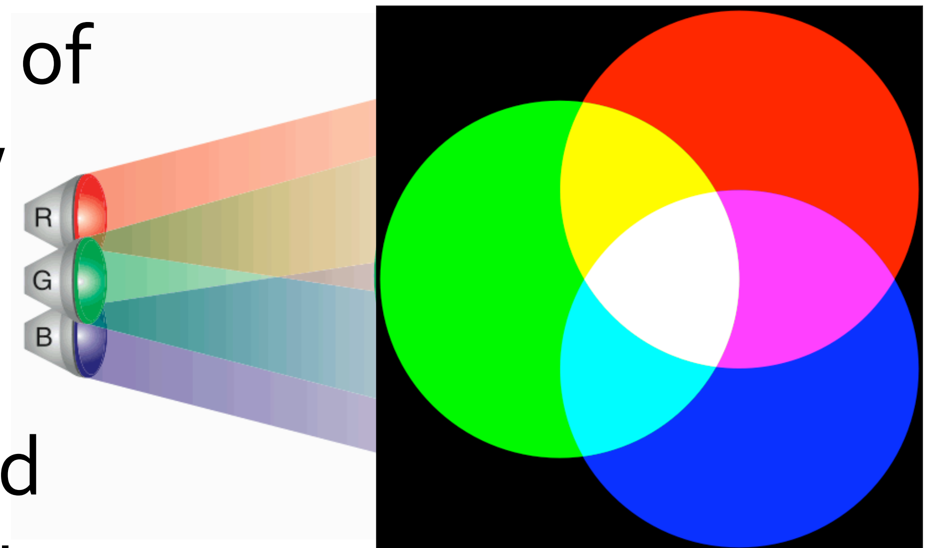
- The Amazing Properties of Colored Light!

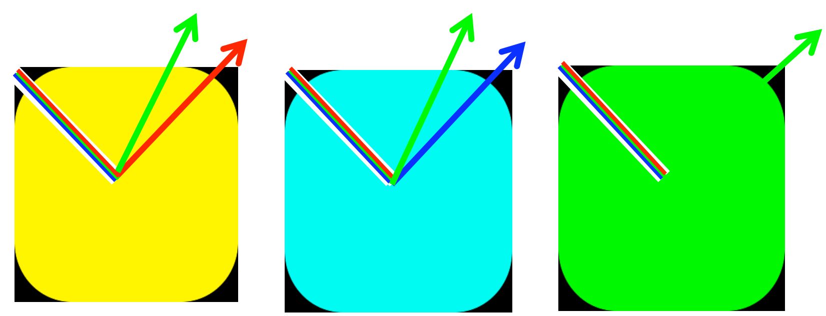


- Caution: It doesn't work like pigment

Green + Red = Yellow?

- Colored light seems to violate our grade school rule of green = blue + yellow What gives?
- In pigment, the color we see is the reflected color from white light; the other colors are absorbed





White, Gray, Black

- You know that gray is just different degrees of white as the "light is turned down" till we get to black

Black	= [0,	0,	0]	0000 0000	0000 0000	0000 0000
Gray	= [128,	128,	128]	1000 0000	1000 0000	1000 0000
White	= [255,	255,	255]	1111 1111	1111 1111	1111 1111

White-gray-black all have same values for RGB

Colors

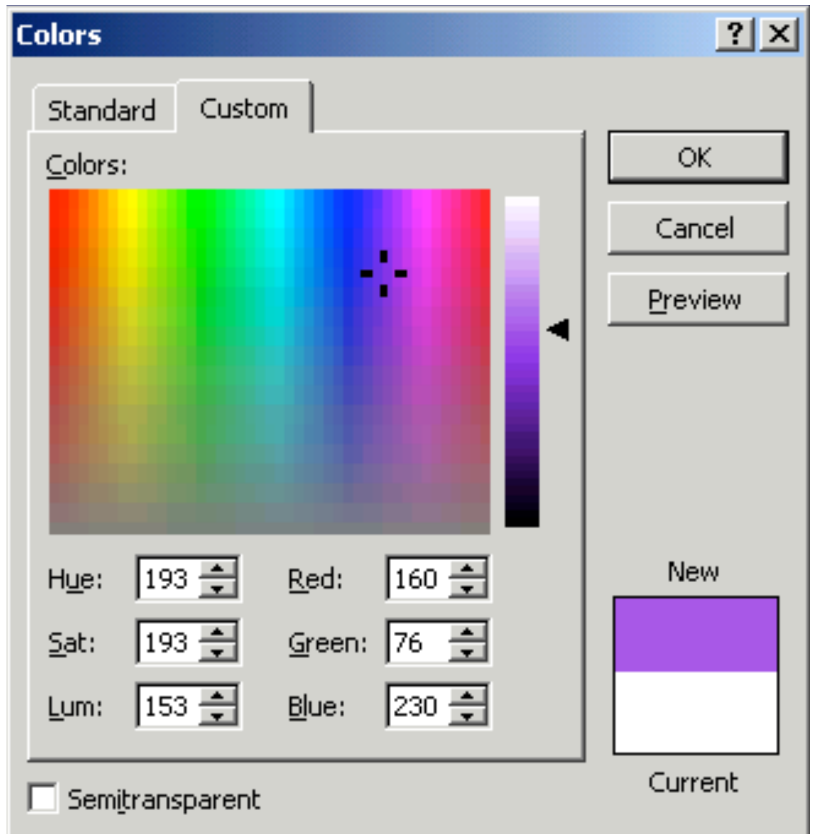
Colors use different combinations of RGB

Husky Purple

Red=160

Green=76

Blue=230



Positional Notation

- The RGB intensities are binary numbers
- Binary numbers, like decimal numbers, use *place notation*

$$1101 = 1\times1000 + 1\times100 + 0\times10 + 1\times1$$

$$= 1\times10^3 + 1\times10^2 + 0\times10^1 + 1\times10^0$$

except that the base is 2 not 10

$$1101 = 1\times8 + 1\times4 + 0\times2 + 1\times1$$

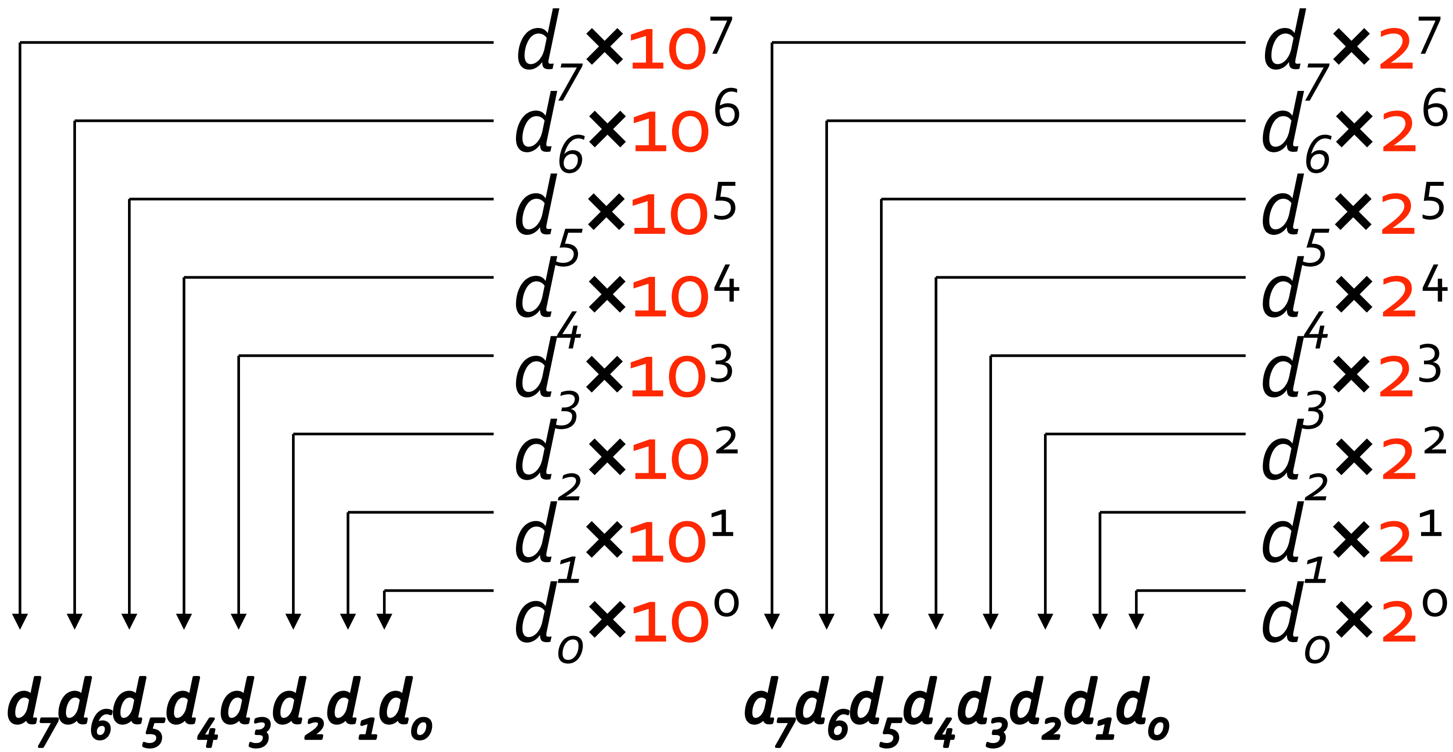
$$= 1\times2^3 + 1\times2^2 + 0\times2^1 + 1\times2^0$$

Base or radix

1101 in binary is 13 in decimal

Positional Notation Logic

Recall that the place represents a power of the base value



The Red of HP As A Binary Number

Given a binary number, add up the powers of 2 corresponding to 1s

1×2^7	$= 1 \times 128$	$= 128$
0×2^6	$= 0 \times 64$	$= 0$
1×2^5	$= 1 \times 32$	$= 32$
0×2^4	$= 0 \times 16$	$= 0$
0×2^3	$= 0 \times 8$	$= 0$
0×2^2	$= 0 \times 4$	$= 0$
0×2^1	$= 0 \times 2$	$= 0$
0×2^0	$= 0 \times 1$	$= 0$

1 0 1 0 0 0 0 0 $\quad$ **=160**

Green of HP As A Binary Number

Given a binary number, add up the powers of 2 corresponding to 1s

0×2^7	$= 1 \times 128$	$= 0$
1×2^6	$= 0 \times 64$	$= 64$
0×2^5	$= 1 \times 32$	$= 0$
0×2^4	$= 0 \times 16$	$= 0$
1×2^3	$= 0 \times 8$	$= 8$
1×2^2	$= 0 \times 4$	$= 4$
0×2^1	$= 0 \times 2$	$= 0$
0×2^0	$= 0 \times 1$	$= 0$

0 1 0 0 1 1 0 0

$= 76$

Is It Really Husky Purple?

- So Husky purple is (160,76,230) which is

  1010 0000 0100 1100 1110 0110

  160 76 230

  Suppose you decide it's not "red" enough

  - **Increase the red by 16 = 1 0000**

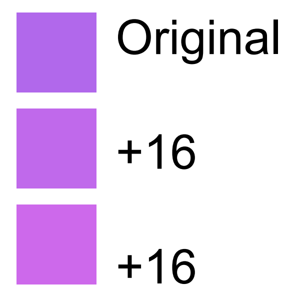
```
  1010 0000
+    1 0000
-----------
  1011 0000
```

Adding in binary is pretty much like adding in decimal

A Redder Purple

- Increase by 16 more

```
   00110 000   ←——  Carries

    1011 0000
+      1 0000
    ---------
    1100 0000
      ↑↑
```



The rule: When the “place sum” equals the radix or more, subtract radix & carry

Check it out online: searching binary addition *hits 19M times, and all of the p.1 hits are good explanations*

Find Binary From Decimal

What is 230 (the Blue of HP)? Fill in the Table:

Num Being Converted	230	230	102	38	6	6	6	2	0
Place Value	256	128	64	32	16	8	4	2	1
Subtract		102	38	6			2	0	
Binary Num	0	1	1	1	0	0	1	1	0

Find Binary From Decimal

Place number to be converted into the table; fill place value row with decimal powers of 2

Num Being Converted	230								
Place Value	256	128	64	32	16	8	4	2	1
Subtract									
Binary Num									

Find Binary From Decimal

Rule: Subtract PV from the number; a positive result gives new number and "1"; otherwise, "0"

Num Being Converted	230	230							
Place Value	256	128	64	32	16	8	4	2	1
Subtract									
Binary Num	0								

Find Binary From Decimal

Rule: Subtract PV from the number; a positive result gives new number and "1"; otherwise, "0"

Num Being Converted	230	230	102						
Place Value	256	128	64	32	16	8	4	2	1
Subtract		102							
Binary Num	0	1							

Find Binary From Decimal

Rule: Subtract PV from the number; a positive result gives new number and "1"; otherwise, "0"

Num Being Converted	230	230	102	38					
Place Value	256	128	64	32	16	8	4	2	1
Subtract		102	38						
Binary Num	0	1	1						

Find Binary From Decimal

Rule: Subtract PV from the number; a positive result gives new number and "1"; otherwise, "0"

Num Being Converted	230	230	102	38	6				
Place Value	256	128	64	32	16	8	4	2	1
Subtract		102	38	6					
Binary Num	0	1	1	1					

Find Binary From Decimal

Rule: Subtract PV from the number; a positive result gives new number and "1"; otherwise, "0"

Num Being Converted	230	230	102	38	6	6			
Place Value	256	128	64	32	16	8	4	2	1
Subtract		102	38	6					
Binary Num	0	1	1	1	0				

Find Binary From Decimal

Rule: Subtract PV from the number; a positive result gives new number and "1"; otherwise, "0"

Num Being Converted	230	230	102	38	6	6	6		
Place Value	256	128	64	32	16	8	4	2	1
Subtract		102	38	6					
Binary Num	0	1	1	1	0	0			

Find Binary From Decimal

Rule: Subtract PV from the number; a positive result gives new number and “1”; otherwise, “0”

Num Being Converted	230	230	102	38	6	6	6	2	
Place Value	256	128	64	32	16	8	4	2	1
Subtract		102	38	6			2		
Binary Num	0	1	1	1	0	0	1		

Find Binary From Decimal

Rule: Subtract PV from the number; a positive result gives new number and "1"; otherwise, "0"

Num Being Converted	230	230	102	38	6	6	6	2	0
Place Value	256	128	64	32	16	8	4	2	1
Subtract		102	38	6			2	0	
Binary Num	0	1	1	1	0	0	1	1	

Find Binary From Decimal

Rule: Subtract PV from the number; a positive result gives new number and "1"; otherwise, "0"

Num Being Converted	230 →	230	102	38	6 →	6 →	6	2	0
Place Value	256	128	64	32	16	8	4	2	1
Subtract		102	38	6			2	0	
Binary Num	0	1	1	1	0	0	1	1	0

Read off the result: 0 1110 0110