MIDTERM REVIEW: THURSDAY

I KNOW WHAT I WANT TO REVIEW. BUT ALSO I WOULD LIKE YOU TO TELL ME WHAT YOU MOST NEED TO GO OVER FOR MIDTERM. BY EMAIL AFTER TODAY’S CLASS.

What can we do with Processing? Let’s check.

Illustrate A Way To Do It

- The Plan ...
  - hide “subversive” protest picture in “calendar art”

Remember how colors work

- Pure red is 255

<table>
<thead>
<tr>
<th>Pure red</th>
<th>Guest</th>
<th>Host</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>1111 0100</td>
<td>1101 0011</td>
<td>1011 1101</td>
<td>1111 0100</td>
</tr>
<tr>
<td>1101 0011</td>
<td>1011 1100</td>
<td>1011 0100</td>
<td>1101 0011</td>
</tr>
</tbody>
</table>

Step 2: Replace Bits In Host

- Put guest bits into right 2 bits of host

What bits matter most?

If you had to throw away some bits from each pixel of a message which should you throw away so that the remaining image was a close to the original as possible?

A. The rightmost (low order bits).  
B. The leftmost (high order bits).  
C. Doesn’t matter, just how many you throw is all that matters.
**What Does the Multiplication *, Mod % Do?**

- Read in the file, and then on mouse click, pull out the bits and make a picture
  - Remove right 2 bits
  - Make them left 2 bits for each color
  - New color

**Data types**

- Primitive data types
  - **int**: numbers without a decimal point.
  - **boolean**: data type for the boolean values `true` or `false`.
  - **color**: storing color values
  - **Function color(R, G, B)**, creates a color that can be stored in variable of the color data type

**Sample Code:**

```cpp
color colorBlinky = color(255, 0, 0);
```

**Multiple operators. Show_mod_mult_onscreen**

```cpp
void setup() {
  //size(136,136);
  textSize(25);
  textAlign(CENTER);
  frameRate(2);
}

void draw() {
  background(0);
  //if (count % 3 == 0) {
  //  println(count);
  //  text(count, 60, 80);
  //}
  count = count + 1;
}
```

**Mod operator %**

- Processing_review_whatisthismodfn
- `int count = 0;
  void setup() {
    size(120, 120);
    textSize(64);
  textAlign(CENTER);
  frameRate(2);
  }
  
  void draw() {
    background(0);
    //if (count % 4 == 0) {
    //  println(count);
    //  text(count, 60, 80);
    //}
    count = count + 1;
  }
```

**Multiplication operator.**

- `int count = 0;`
The **if** statement

```
if (BooleanExpression)
  Statement
```

- **Expression** (test) - true or false (boolean).
- **Statements** - executable: call a function, assign a value, etc.
- **if** expression inside the parentheses is true, execute the statements enclosed within the curly brackets.
- **else**: extends the if structure, when the expression in if() is false, execute the statements enclosed within the curly brackets.

### Sample syntax

```java
if (expression) {
  statements
} else {
  statements
}
```

### Returns a boolean value (true/false)

**Relational Operators**

- Return true/false expressions (or boolean values)
- `==` (equality)
- `>` (greater than)
- `>=` (greater than or equal to)
- `!=` (inequality)
- `<` (less than)
- `<=` (less than or equal to)

**Sample Code**

```java
int n = 0;
if (n == 0) {
  n = n + 1;
} else {
  n = n - 1;
}
```

**Logical Operators**

- Connect two or more true/false expressions (or boolean values)
- `&&` (logical AND)
- `!` (logical NOT)
- `||` (logical OR)

**Sample code using "&&"**

```java
if (x < 100 && y > 100)
```

**Sample code using "!"**

```java
if (!a)
```

**Sample code using "||"**

```java
if (a || b)
```

### HW 7 answer part 2

**This can work too (notice the "else if"):**

```java
if (mouse.x > 130 && x) {
  dir = 1;
  look = 2;
} else if (mouse.x < 110 && y) {
  dir = -1;
  look = 2;
} else if (mouse.x < 110 && y) {
  dir = -1;
  look = 0;
} else {
  dir = 0;
  look = 1;
}
```

**This will not work, Blinky will not move right:**

```java
if (mouse.x > 130 && x) {
  dir = 1;
  look = 2;
} else if (mouse.x < 110 && y) {
  dir = -1;
  look = 2;
} else if (mouse.x < 110 && y) {
  dir = -1;
  look = 0;
} else {
  dir = 0;
  look = 1;
}
```

Can you figure out why?
mouseX, mousePressed() 

- mouseX: this system variable always contains the current horizontal coordinate of the mouse
- mouseY: this system variable always contains the current VERTICAL coordinate of the mouse
- mousePressed(): this function is called automatically by the system once after every time a mouse button is pressed.

Sample syntax:  
```void mousePressed() {  
  statements  
}  
```

HW 7 answer part 3.

We can find out the color through the variable "colorFlag" 

```void mousePressed() {  
  if (colorFlag==red) {  
    colorBlinky=color(255, 255, 0);  
    mark me as yellow  
  }  
  else {  
    colorBlinky=color(255, 0, 0);  
    Change me to red  
  }  
}  
```

Processing Review: What you should have seen so far.
Kinds of multiple choice questions that might be on midterm

What is a variable?
- a named location in the computer’s memory

Variables
- store/remember values  
- can be changed  
- must be declared to store a particular kind of value (e.g. whole number, fraction, character, color, image, boolean)  
- should have a descriptive name (so you know what it does or is supposed to do)  
- start with letter  
- can also include numbers  
- no spaces

```void setup() {  
  size(500, 500);  
}  
int diameter = 0;  
void draw() {  
  ellipse(width/2, height/2, diameter,diameter);  
  diameter = diameter + 5;  
}  
```

What does this display?
A) many nested circles (like a target) growing in size  
B) a white circle growing in size  
C) a black circle growing in size  
D) a pulsing images of circles growing and shrinking  
E) nothing (a circle with diameter 0)
primitive types (Java)

- boolean - true or false
- char - 'a', 'b', 'c', ...
- byte - small integer -128 to 127
- short - bigger integer -32768 to 32767
- int - even bigger integer +/- 2 billion
- long - really big integer
- float - numbers with fractional parts 3.1415
- double - like float but more precision

declaring vs. initializing vs. assigning

```java
int carFront; // declare
int carFront = 100; // declare and initialize
carFront = carFront - 1; // assign
```

void setup() {
  size(200, 200);
}

void draw() {
  background(255);
  int xPos = 0;
  ellipse(xPos, height/2, 20, 20);
  xPos = xPos + 1;
}

What does this draw?
A) circle moving across the screen left to right
B) circle moving across the screen right to left
C) circle moving across the screen top to bottom
D) circle moving across the screen bottom to top
E) half-circle on the left edge not moving

System Variables (Processing)

- mouseX, mouseY
- pmouseX, pmouseY
- width, height
- frameCount

Processing review placement click 1

```java
void setup() {
  size(400, 400);
}

void draw() {
  fill(255-abs(mouseX-pmouseX));
  rect(pmouseX, pmouseY, mouseX, mouseY);
}
```

Processing review placement click 3

```java
void setup() {
  size(400, 400);
  __________________ // position A
  __________________ // position B
  __________________ // position C
  __________________ // position D
}

void draw() {
  fill(255-abs(mouseX-pmouseX));
  rect(pmouseX, pmouseY, mouseX, mouseY);
  __________________ // position E
}
```

Where should the line "background(255);" be placed so that the sketch shows just a single moving rectangle?
Choose option E if it would work with either C or D.
Making Choices

- If you wish to defrost, press the defrost button; otherwise press the full power button.
- Let the dough rise in a warm place until it has doubled in size.
- If the ball reaches the side of the display change its direction.

Boolean Expressions

- Any expression that evaluates to true or false.
- Relational operators, <, <=, >, >=.
- Equality operators, ==, !=.
- For example:
  ```
  int i = 3, j = 4;
  5 < 6
  i == j
  (j + 2) <= 6
  ```

Expressions and Statements

- Expression statements are formed by adding a semicolon to the end of certain types of expressions.
- An assignment expression is an expression involving an assignment.
  ```area = width * height;```
- A method call expression has no assignment.
  ```rect(•••);```

Non-statements

- Not all expressions can be turned into statements. The following are syntax errors.
  - `x+y;
  - width > 20;
- The above do not make sense as statements. They don’t DO anything. Statements must DO something:
  - assign a new value to a variable
  - cause some output to occur (println(), rect())
  - change some internal “state”
  - `background(255);
  - `noStroke();

Blocks

Several statements can be grouped into a block using `{`:
```
int x = 20, y = 30, size = 40;
ellipseMode(CORNER);
fill(255, 0, 0);
rect(x, y, size, size);
fill(0, 255, 0);
ellipse(x, y, size, size);
}
```

The `if` statement

```if (BooleanExpression) Statement```

```
if (BooleanExp)
    Statement
```

// x, y, and size above cannot be used // here
int count = 0;
void setup() {
  size(120,120);
  textSize(64);
  textAlign(CENTER);
  frameRate(2);
}
void draw() {
  background(0);
  if (count % 3 == 0) {
    println(count);
    text(count,60,80);
  }
  count = count + 1;
}

What is the effect of the if-clause?
A. The drawing only happens every other frame.
B. The drawing gets bigger and smaller depending on the current value of count.
C. The background changes color when count % 2 equals 0
D. Nothing. The value of count keeps getting set to 0 every frame.

What happens as count changes value?
A. The color of the ellipse changes
B. The color of the corners change
C. Both the ellipse and the corner change color
D. The rectangle with the ellipse blinks
E. All of the above
What can we do with Processing we haven’t done yet?

Announcements

- **Creativity in Processing**
  - Due the week after the midterm. Will need to start right after the midterm.
  - Get a programming partner! Go Crazy!
  - [https://courses.soe.ucsc.edu/courses/cmps10/Winter13/01/pages/syllabus](https://courses.soe.ucsc.edu/courses/cmps10/Winter13/01/pages/syllabus)
  - **Thursday** : Review for Midterm (a week from today on Feb 12th!!!)

HW9: Creativity in Processing

**Assignmt.** Write four programs to do whatever you want (but don’t copy the examples above), and try to make them clever or interesting or cute or have some property that would interest a viewer. You should try to use those you have learned in your former Processing homework, because those are the basics and one goal of this assignment is to practice the basics. But, if you need some other feature of processing that you find in the reference page, go ahead and use it. The goal is creativity … but don’t spend forever on it either.

Why Pair Programming?

- Having a partner gives you a built-in helper
- Many people find computers more fun if they get to work with someone else
- Learning to work with computers is like learning a foreign language, lots of new words and codes
  - Easier to learn a language if you have some to ‘converse with’ in the language
- Working with a partner helps you develop good teamwork skills
- Most programming projects in real life done in teams
- Employers look for people who are good at teamwork

A creativity program from last year

Sierpinski: function call when mouse pressed
Art In A Click

- Computer art, that is, art generated by computers, not art created by people using computers, leads to some fun Web sites.
- Google “piet mondrian”. He was a cubist and created pictures that look like this:
  <http://www.google.com/search?client=safari&rls=en&q=“Piet+Mondrian”&ie=UTF-8&oe=UTF-8>
- Could we generate this in processing?

What would we do in Processing? What does Random do?


How about something a little easier first?

- ideas? More Pictures!
Random Numbers

- Random numbers should be called random number sequences, because the definition requires that no matter how many numbers you already know in the sequence, it’s not possible to predict the next one. A non-random sequence is 2, 4, 6, 8, 10, …
- Computers cannot produce random numbers (because computers are completely predictable), but they can produce a sequence of numbers that passes all of the tests for randomness. These are called pseudo-random numbers, but everyone drops the “pseudo” part.
- To generate a random number in Processing we write:
  - `random(<smallest possible number>, <largest possible number>)`
  - We get back a number – we can’t predict which – between the two limits, including the end points.
  - To generate a random number between 0 and 255, write `random(0, 255)`.
  - To generate a number between 0 and 1, write `random(0, 1)`.

Let's Try it!

```
void setup() {
  size(200, 200);
  background(255);
}
void draw() {
  ellipseMode(RADIUS);
  strokeMode(BLENDED);
  lerpColor(0, 1.0, .33, .66, color(255, 0, 0), color(0, 255, 255));
}
```

How about Jackson Pollock?

- Google “jackson pollock”.
- He was an abstract expressionist and created paintings that look like this!

Computer generated Jackson Pollock?

- I made this on this website.

IerpColor()?

- There is a very interesting function called `lerpColor()`. It uses a mathematical idea (that we don’t need to know about) called linear interpolation to pick intermediate colors.
- Give it two colors, say gold and purple, and it finds a color in between.
- Where in between? We also give it a fraction (.33, .66) between 0 and 1 that tells.
- Program fills in the black region in the middle with two colors between gold and purple.