COMP 125:
Basics of Computer Organization
Week 04, Lecture 07

Agenda

1. Big Review!
    a. Structure
    b. Interface
    c. Functions
    d. Variables
    e. Conditionals
    f. Loops
    g. Top-Down Problem Solving
2. Practice!

Big Review

Why?

- We have covered a lot in the margins
- Let's get it all in one place - and then apply it!
- Everything in the following slides **is fair game** on the midterm

Structure

- JavaScript runs code one line at a time
- Variables and functions keep code reusable and readable
- Function contents are wrapped in curly brackets ({ and }) - resulting in *code blocks*
- Comments are preceded by two forward-slashes (“//”) and are not executed as code

Interface

- P5.js has the concept of a canvas, created at the start of the setup function
  - x goes left-right
  - y goes from top-bottom
  - (x, y) == (0, 0) is the upper *left* corner
- Setup runs once, draw runs repeatedly

Variables

- Types:
  - numbers
  - strings
  - booleans
- Scope:
  - Where does the program recognize the variable?

```
1  var x = 0;
2
3  function setup() {
4    createCanvas(200, 150);
5  }
6
7  function draw() {
8    background(200);
9    drawSquare();
10   drawCircle();
11 }
12
13 function drawSquare(){
14   var x = 10;
15   var y = 10;
16   var d = 50;
17   rect(x, y, d, d);
18 }
19
20 function drawCircle(){
21   var x = 100;
22   var y = 70;
23   var d = 40;
24   ellipse(x, y, d, d);
25 }
26
```

Functions

- Like a variable to store code
- Repeatability and readability
- Elements:
  - “`function`”
  - The name cannot contain spaces (use underscores)
  - Curly brackets outline the code in the function
  - Parenthesis outline the arguments (inputs)
  - “`return`”
- Make sure you call the function!

```
1  function setup() {
2    createCanvas(200, 150);
3  }
4
5  function draw() {
6    background(200);
7    var result = add_two(1,50)
8    text(result, 10, 10)
9  }
10
11 function add_two(a, b){
12   var result = a + b
13   return result
14 }
```

Conditionals

- True/False
- Operators include:
  - Compariators
  - AND (&&)
  - OR (||)
  - NOT (!)
- Control code using conditionals and if/else if/else statements

```
1  function setup() {
2    createCanvas(200, 150);
3  }
4
5  function draw() {
6    background(200);
7    var x = 10
8    var y = 100
9    compare(x, y)
10 }
11
12 function compare(a, b){
13   if (a > b) {
14     text(a + " is larger than " + b, 10, 10)
15   }
16   else if (a < b){
17     text(a + " is smaller than " + b, 10, 10)
18   }
19   else {
20     text(a + " and " + b + " are equal", 10, 10)
21   }
22 }
```

Loops

- while
  - Track the conditional
  - Run the loop until the conditional is False
  - Note - something has to change!
- for
  - Sets the initial condition and the change
  - Run the loop until the conditional is False

Loops

- while
  - Track the conditional
  - Run the loop until the conditional is False
  - Note - something has to change!
- for
  - Sets the initial condition and the change
  - Run the loop until the conditional is False

```
function setup() {
  createCanvas(400, 400);
}

function draw() {
  background(220);

  var i = 1

  while (i <= 10){
    text(i, 10, i * 10)
    i = i + 1
  }
}
```

```
function setup() {
  createCanvas(400, 400);
}

function draw() {
  background(220);
  for (i = 1; i <= 10; i = i + 1){
    text(i, 10, i * 10)
  }
}
```

Top-Down

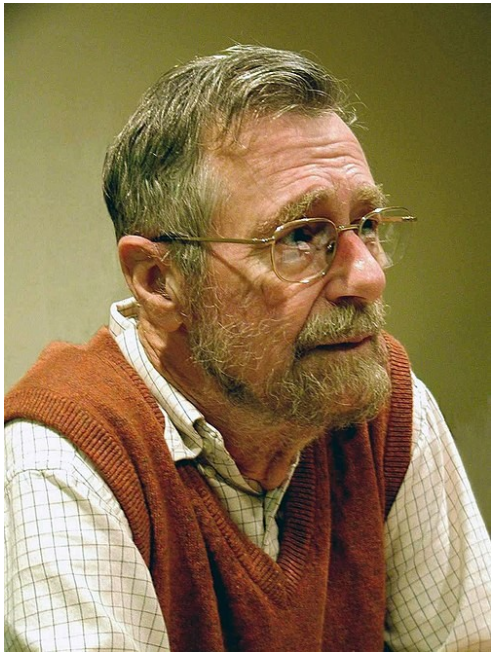
In essence:

Programs should solve problems.

We can’t write decent programs if we are mired in details.

Solution?

Start generic, and specify until it works.



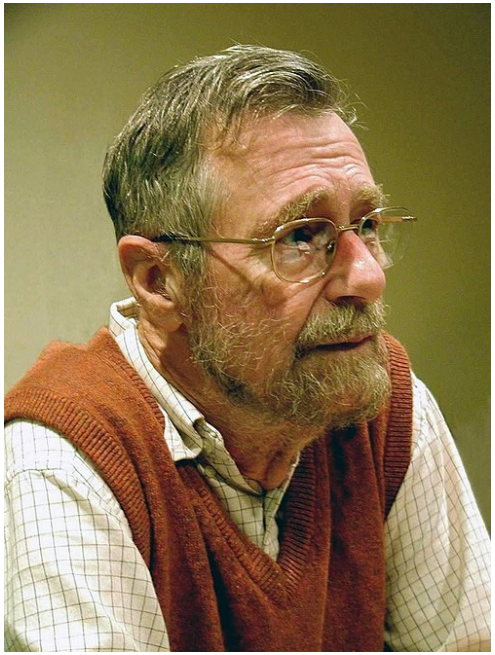
Edsger W. Dijkstra

Father of computer programming

Top-Down

1. Solve the problem - feel free to be generic!
2. Pick some part, and make it more specific
    a. Smaller steps?
    b. Store something in a variable?
    c. Repetition?
    d. Call a function?
3. If you aren't sure what to do, search for one of your steps
4. Do this until the problem is accurately solved by the program

When you're done, if it's a mess, or if it's too slow, you now have a new problem to solve!



Edsger W. Dijkstra

Father of computer programming

Let’s try it!

Practice

Example 0:

Show the letters A - M along the left side of the canvas.

Practice

Example 1:

Put the x-location of the cursor (mouse) in the upper left corner and the y-location of the mouse in the lower left corner

Practice

Example 2:

Draw a circle whose diameter is half the size of the canvas centered in the canvas

Practice

Example 3:

Draw a circle whose diameter is half the size of the canvas centered in the canvas, and make it blue

Practice

Example 4:

Draw a circle whose diameter is half the size of the canvas centered in the canvas, and make it blue or red depending on if the cursor is on the right half of the screen (blue) or the left (red)

Practice

Example 5:

Draw five circles on the canvas that do not touch each other and are all the same size

Practice

Example 6:

Draw five circles on the canvas that do not touch each other and are all the same size but different colors