Functions

Defining Functions
Function Parameters
Returning values

- What is a Function?
  - Code that is “packaged” so it can be run “by name”
  - Often has parameters to change what it does
  - Often performs some computation and returns a value

- We use functions all the time in JavaScript, e.g.:
  ```javascript
  noFill();
  point(20, 30);
  let angle = map(i, 0, 10, -2, 2);
  ```

- We've customized special built-in event functions, e.g.:
  ```javascript
  function draw() { ... }
  ```

- We can also define our own custom functions ...
Why Custom Functions?

- Custom functions make code **modular**
- Modular code is:
  - easier to reuse
  - easier to test
  - easier to understand

Credit: www.babble.com

Credit: http://www.cutting-mats.net/

Anatomy of a Function Definition *without* Parameters

means “define a new function”

function name

empty brackets means “no function parameters”

```javascript
function tree() {
    noStroke();
    // trunk
    fill("#5A3709"); // brown
    quad(45, 40, 55, 40, 60, 90, 40, 90);
    // leaves
    fill("#5CBF17"); // green
    ellipse(50, 30, 55, 55);
}
```

function code block
function draw() {
    // call the tree function
tree();
}

// draw tree
// with bottom of trunk at 50, 90
function tree() {
    ...
}

Calling Functions

- **You must call your custom functions** or they won’t execute
- JavaScript p5 calls setup and draw for you
  - setup and draw are “built-in” functions that you customize

Defining Functions

- Define your functions “outside” setup and draw
  - the same place global variables are declared
function setup() {
    frameRate(1);
    print("setup");
}

function draw() {
    print("draw start");
    myFunction();
    print("draw end");
}

function myFunction() {
    print(" myFunction");
}

(function definition order doesn’t matter

function printB() {
    print("B");
}

function setup() {
    printA();
    print("Setup");
    printB();
}

function printA() {
    print("A");
}

https://editor.p5js.org/cs105/sketches/40THSvHlm
What does this print to the console?

```javascript
function setup() {
  print("apple");
sayBanana();
sayOrange();
}

function sayOrange() {
  print("orange");
}

function sayBanana() {
  print("banana");
}
```

Anatomy of a Function Definition *with* Parameters

```javascript
function drawTree(x, y) {
  noStroke();
  // trunk
  fill("#5A3709"); // brown
  quad(x - 5, y - 50, x + 5, y - 50,
       x + 10, y, x - 10, y);
  // leaves
  fill("#5CBF17"); // green
  ellipse(x, y - 60, 55, 55);
}
```
(Review) Calling Functions with Arguments

```javascript
function name arguments
drawTree(40, 150);
```

```javascript
// use the tree function to create a forest
tree(40, 90);
tree(100, 90);
tree(160, 90);
}
```

```javascript
// draw tree
// x, y position (centre bottom of trunk)
function tree(x, y) {
    ...
}
```

https://editor.p5js.org/cs105/sketches/KRLY8p0mS
What does this print to the console?

```javascript
function setup() {
  fruit(-1);
  print("apple");
}

function fruit(a) {
  if (a > 0) {
    print("orange");
  } else {
    print("banana");
  }
}
```
function vehicle(x, y, hue) {
  ...
}

https://editor.p5js.org/cs105/sketches/ylnubokjs
Function attributes are “passed-by-value”

- “passed-by-value” means the function parameter is a copy of the argument variable used in the function call
- changing the value of a function parameter inside the function code block **will not change** the value of the variable used as an argument

```javascript
function setup() {
  let a = 10;
  print(a); // 10
  f(a);
  print(a); // 10
}

function f(b) {
  b = b + 1;
}
```

Functions and Scope

- Custom functions can access global variables, (just like setup and draw)
- Custom functions can have local variables, (just like setup and draw)
function setup() {
    a();
    ellipse(25, 50, 50, 50);
    b();
    ellipse(75, 50, 50, 50);
}

function a() {
    strokeWeight(10);
    fill(128); // grey
}

function b() {
    strokeWeight(2);
    fill(255); // white
}
// draw trees behind with a for loop
for (let x = 50; x < width; x += 100) {
  tree(x, 90);
}

// draw the car
vehicle(vehicleX, 92, vehicleHue);

// draw trees in front with a while loop
let x = 100;
while (x < width) {
  tree(x, 95);
  x = x + 100;
}

Starter: https://editor.p5js.org/cs105/sketches/KRLY8p0mS
https://editor.p5js.org/cs105/sketches/MnlQU8-zt

Functions that Return Values

- We use functions that return values all the time:

  let x = random(1, 100);
  let a = map(x, 1, 200, 20, 30);
  fill(random(0, 255));
A Function with a *Return Value*

function sum (a, b, c) {
    ... 
    return __________;
}

this is where the value is returned

**Example**

// calculate the sum of three numbers
function sum(a, b, c) {
    let total = a + b + c;
    return total;
}

required return statement
function setup() {
    let answer = sum(1, 2, 3);
}

function sum(a, b, c) {
    let total = a + b + c;
    return total;
}
function setup() {
    let answer = sum(1, 2, 3);
}

function sum(a, b, c) {
    let total = a + b + c;
    return total;
}
function setup() {
    let answer = sum(1, 2, 3);
}

function sum(a, b, c) {
    let total = a + b + c;
    return total;
}

total = 6
function setup() {
    let answer = sum(1, 2, 3);
}

function sum(a, b, c) {
    let total = a + b + c;
    return total;
}
function setup() {
    let answer = sum(1, 2, 3);
}

function sum(a, b, c) {
    let total = a + b + c;
    return total;
}
Analogy

setup() $\rightarrow$ sum(…)

----------

 Analogy

setup() $\rightarrow$ sum(…)

1, 2, 3

a = 1
b = 2
c = 3
Analogy

setup() sum(…)

Analogy

setup() sum(…)

CS 105 - Functions
setup() sum(...)  

function setup() {
  let x = sum(2, 4, 6);
  let y = sum(23, 45, 71) / 2;
  let z = sum(x, y, 10);
  line(0, 0, width, sum(x, y, z));
}

// calculate the sum of three numbers  
function sum(a, b, c) {
  let total = a + b + c;
  return total;
}

https://editor.p5js.org/cs105/sketches/kyhFK5MA2
Using Functions with Return Values

- Can use a function that returns values *anywhere* you use the same type of value, e.g.

```javascript
// a = 12
let a = sum(2, 4, 6);

function setup() {
  // b = 12 * 100
  b = sum(2, 4, 6) * 100;
  // c = 2 + 4 + 12
  c = sum(2, 4, sum(2, 4, 6));

  // point(12, 10)
  point(sum(2, 4, 6), 10);

  // if (12 > 3) { ... } 
  if (sum(2, 4, 6) > 3) { ... }
```

What does this print to the console?

- A. twice
- B. 1
- C. 2
- D. r
- E. nothing, it's a syntax error

```javascript
function twice(a) {
  let r = a + a;
  return r;
}

function setup() {
  print(twice(1));
}
```
What does this print to the console?

function setup() {
    print(twice(twice(1)));
}

function twice(a) {
    let r = a + a;
    return r;
}

What does this print to the console?

function setup() {
    print(twice(1) * twice(1));
}

function twice(a) {
    let r = a + a;
    return r;
}
What does this print to the console?

A. 1
B. 2
C. 3
D. Nothing, it's a syntax error
E. Not sure what will happen since thrice doesn't return anything

```javascript
function setup() {
  print(thrice(1));
}

function thrice(a) {
  let r = a + a + a;
}
```

Possible Logic Error: Forgetting to Return a Value

- If you don’t return a value, the function returns “undefined”

```javascript
function setup() {
  let a = thrice(1);
}
```

```javascript
function thrice(a) {
  let r = a + a + a;
}
```
Possible Logic Error: Forgetting to Assign

- You usually want to use a function return value
  - it may be a logic error if you don’t assign it to a variable

```javascript
function setup() {
  sum(3, 6, 9); // nothing assigned

  random(1, 1000); // nothing assigned
}

myMax

// returns the largest number (a or b)
function myMax(a, b) {
  let largest;
  if (a > b) {
    largest = a;
  } else {
    largest = b;
  }
  return largest;
}
```

[Link to editor.p5js.org sketch](https://editor.p5js.org/cs105/sketches/ITVfqQ-Ml)
myMax (unit tests)

```javascript
function setup() {

  // unit tests
  print(myMax(5, 10), 10); // answer is 10
  print(myMax(15, 10), 15); // answer is 15
  print(myMax(-500, 10), 10); // answer is 10
  print(myMax(0.5, 0.1), 0.5); // answer is 0.5
  print(myMax(10, 10), 10); // answer is 10
}
```

https://editor.p5js.org/cs105/sketches/ITVfqQ-Ml

myMax (early return)

```javascript
// returns the largest number (a or b)
function myMax2(a, b) {
  if (a > b) {
    return a;
  } else {
    return b;
  }
}
```

https://editor.p5js.org/cs105/sketches/ITVfqQ-Ml
P5 has built-in functions: `max`, `min`

- return the largest of two numbers
  ```javascript
  max(a, b)
  ```

- return smallest of two numbers
  ```javascript
  min(a, b)
  ```

`myConstrain`

Constrain a value be inside a certain range.

If value is in between low and high, return the value.
If value is less than low, return low
If value is greater than high, return high.

```javascript
// constrain n to be between low and high value
function myConstrain(n, low, high) {
  ...
}
```
P5 has a built-in function: constrain

- constrains n to be between low and high
  constrain(n, low, high)

What does this draw after 1,000,000 frames?
What does this draw after 1,000,000 frames?

```javascript
function draw() {
  point(min(50, random(0, width)), max(random(0, height), 50));
}
```

Circle Hit Testing (Bounds Detection)

- Rectangle hit test won’t work for a circle
- If distance from mouse to circle center is less than radius, then circle is “hit” (e.g. rollover, clicked on, etc.)
  - radius is the distance between circle centre to circle edge

Rectangle Hit Test: https://editor.p5js.org/cs105/sketches/hh943x5vL
Circle Hit Testing (Bounds Detection)

Hit

Miss

Distance Between Two Points

strokeWeight(10);
point(77, 81); // P
point(13, 45); // Q

// calculate distance
// between point P and Q
let d = [how to do this?]
Pythagorean Theorem

\[ c = \sqrt{a^2 + b^2} \]
\[ c^2 = a^2 + b^2 \]

**myDist**

function to calculate distance between two points

use built-in function `sqrt()` for square root

// returns the distance between point P at (pX, pY) // and point Q at (qX, qY)
function myDist(pX, pY, qX, qY) {
    let a = pY - qY; // y difference
    let b = pX - qX; // x difference
    let c = sqrt((a * a) + (b * b));
    return c;
}
circleHitTest

// returns true if pX, pY is inside a circle
// that is centred at cX, cY with radius r
function circleHitTest(pX, pY, cX, cY, r) {
  let d = myDist(pX, pY, cX, cY);
  if (d <= r) {
    return true;
  } else {
    return false;
  }
}

Processing has a built-in function for distance

- returns a Number

dist(x1, y1, x2, y2)

Use this built-in dist function to calculate the distance between two points in your labs and assignments!
Mouse Speed

- The distance between the current mouse position and the previous mouse position is the speed of the mouse
  - pmouseX and pmouseY are built-in global variables for the previous mouse position

let speed = dist(mouseX, mouseY, pmouseX, pmouseY);

- since we calculate it each time draw is called, the speed will be in “pixels per frame”
  - since frameRate is 60 frames per second, multiply by 60 to get “pixels per second”

maxspeed

let maxSpeed = 0;

function draw() {
  background(220); // grey

  let speed = dist(mouseX, mouseY, pmouseX, pmouseY);

  // update the max speed so far
  if (speed > maxSpeed) {
    maxSpeed = speed;
  }

  // display the max speed and current speed
  fill(0); // black
  text(round(maxSpeed), width / 2, height / 2 - 40);
  text(round(speed), width / 2, height / 2 + 40);
}

https://editor.p5js.org/cs105/sketches/dmaaz93v7
let speed = dist(mouseX, mouseY, pmouseX, pmouseY);

// draw an ellipse with size representing speed
fill(255, 128); // semi transparent white
ellipse(mouseX, mouseY, speed, speed);

// draw line with hue based on speed
colorMode(HSB);
let hue = map(speed, 0, 30, 200, 0);
strokeWeight(10);
stroke(hue, 100, 100);
line(pmouseX, pmouseY, mouseX, mouseY);

https://editor.p5js.org/cs105/sketches/LxaXWkTfa