

CS50 - SECTION 5

10/16/18



POST-MORTEM ON PSET 4

QUICK REMINDERS

1. Remember to check for a `NULL` return when using `malloc()`, `fopen()`, etc.
2. Be careful how you implement your functions and try to encapsulate as much output as possible
 - a. `hash()` should perform the `tolower()` and `%` operations for you
3. Extract common code from if-else branches
 - a. Only use the if-else statement for non-common, distinct code (remember DRY)

CONCEPTS DEEP-DIVE

“SHORTS” FOR THE WEEK

A man with short brown hair, wearing a maroon polo shirt, stands against a light gray background. The text "INTERNET PRIMER" is overlaid in white.

INTERNET PRIMER

https://www.youtube.com/watch?v=04GztBlVo_s

A man with short brown hair, wearing a maroon polo shirt, stands against a light gray background. The text "IP" is overlaid in white.

IP

<https://www.youtube.com/watch?v=A1q9SokDJSU>

A man with short brown hair, wearing a maroon polo shirt, stands against a light gray background. The text "HTTP" is overlaid in white.

HTTP

<https://www.youtube.com/watch?v=4axL8Gfw2nI>

A man with short brown hair, wearing a maroon polo shirt, stands against a light gray background. The text "TCP" is overlaid in white.

TCP

https://www.youtube.com/watch?v=GP7uvl_6uas

A man with short brown hair, wearing a maroon polo shirt, stands against a light gray background. The text "HTML" is overlaid in white.

HTML

<https://www.youtube.com/watch?v=YK78KhMf7bs>

A man with short brown hair, wearing a maroon polo shirt, stands against a light gray background. The text "CSS" is overlaid in white.

CSS

<https://www.youtube.com/watch?v=Ub3FKU21ubk>

A man with short brown hair, wearing a maroon polo shirt, stands against a light gray background. The text "JAVASCRIPT" is overlaid in white.

JAVASCRIPT

<https://www.youtube.com/watch?v=Z93IaNfavZw>

TCP/IP MODEL

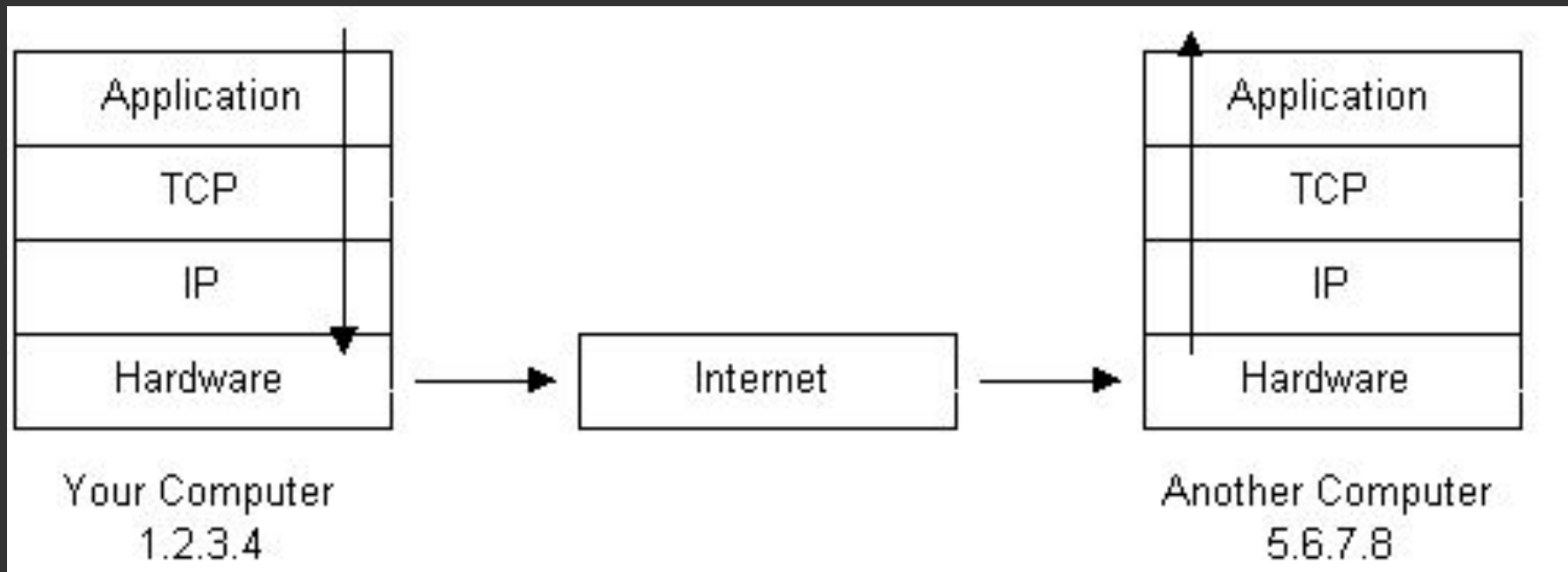
- The **TCP/IP Model** is simply a set of protocols that govern how communication happens over the internet
- Communication happens over a set of *layers* and information is broken up into *packets*

STANDARD LAYERS OF THE PROTOCOL STACK

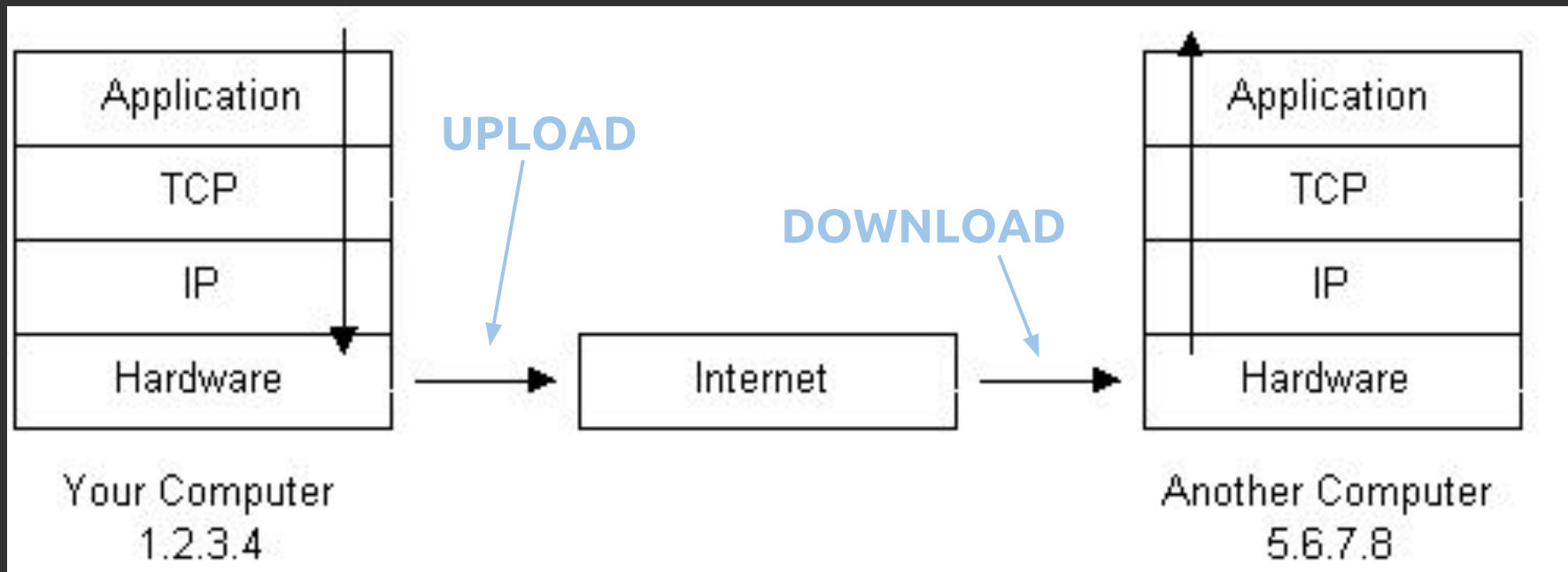
PROTOCOL LAYER	DESCRIPTION
APPLICATION	Protocols specific to applications such as WWW, e-mail, FTP, etc.
TCP	TCP directs packets to a specific application on a computer using a port number.
IP	IP directs packets to a specific computer using an IP address.
HARDWARE	Converts binary packet data to network signals and back. (E.g. ethernet network card, modem for phone lines, etc.)

Source: <https://web.stanford.edu/class/msande91si/www-spr04/readings/week1/InternetWhitepaper.htm>

AN EXAMPLE OF UPLOAD/DOWNLOAD



AN EXAMPLE OF UPLOAD/DOWNLOAD



HYPertext TRAnSFER PROTOCOL (HTTP)

- **HTTP** defines the the way different messages are transferred over the images (“what’s inside the envelope”)

GET / HTTP/1.1

Host: www.google.com

HYPertext TRAnSFER PROTOCOL (HTTP)

- **HTTP** defines the the way different messages are transferred over the images (“what’s inside the envelope”)

**REQUEST
METHOD**



GET / HTTP/1.1
Host: www.google.com

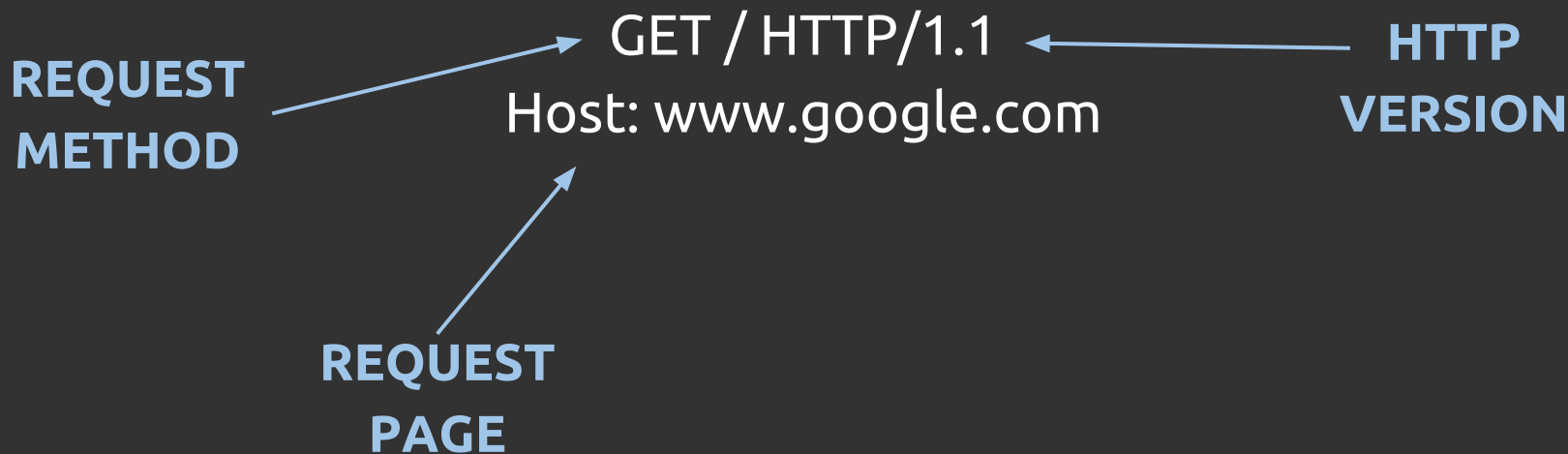
HYPertext TRAnSFER PROTOCOL (HTTP)

- **HTTP** defines the the way different messages are transferred over the images (“what’s inside the envelope”)

REQUEST METHOD → GET / HTTP/1.1 ← **HTTP VERSION**
Host: www.google.com

HYPertext TRAnSFER PROTOCOL (HTTP)

- **HTTP** defines the the way different messages are transferred over the images (“what’s inside the envelope”)



HYPERTEXT TRANSFER PROTOCOL (HTTP)

- **HTTP** defines the the way different messages are transferred over the images (“what’s inside the envelope”)



HYPertext TRAnSFER PROTOCOl (HTTP)

- HTTP requests return a **status code** to indicate how the request was processed

GET / HTTP/1.1
Host: www.google.com


200: OK

HTTP REQUEST METHODS

SAFE METHODS NO ACTION ON SERVER	{	GET	HTTP/1.1 MUST IMPLEMENT THIS METHOD
		HEAD	INSPECT RESOURCE HEADERS
MESSAGE WITH BODY SEND DATA TO SERVER	{	PUT	DEPOSIT DATA ON SERVER — INVERSE OF GET
		POST	SEND INPUT DATA FOR PROCESSING
		PATCH	PARTIALLY MODIFY A RESOURCE
		TRACE	ECHO BACK RECEIVED MESSAGE
		OPTIONS	SERVER CAPABILITIES
		DELETE	DELETE A RESOURCE — NOT GUARANTEED

HTTP STATUS CODES

HTTP STATUS CODE



Code	Message
200	OK
301	Moved Permanently
302	Moved Temporarily
404	Not Found
500	Internal Server Error
503	Service Unavailable

404
PAGE NOT
FOUND

A status code is a part of the response returned by the server when a client (e.g., a browser) calls a URL. With the help of a status code, the server tells the client whether the request (e.g., "send me the page www.newyorker.com") was successfully processed or whether an error occurred.

And many more...

APPLICATION PROGRAMMING INTERFACES (APIs)

- An API is a standardized way for two web applications to talk to one another
- APIs are all about *transferring data* from one place to another in a standard way
 - Example: Twitter has an API to allow you to request tweets from it, post tweets, etc.

APPLICATION PROGRAMMING INTERFACES (APIs)

Why do APIs matter?

APPLICATION PROGRAMMING INTERFACES (APIs)

Why do APIs matter?

1. We need them to communicate with a web application in a standardized way.
 - a. Twitter wouldn't know how to handle our requests if they weren't structured in a consistent way
2. They greatly expand the opportunities for interoperability on the web
 - a. You can now write web applications pull in data from other sources and post your own data elsewhere!

RESTful APIs

- **Representational state transfer (REST)** is web-lingo for a standardized way of creating web resources
- REST APIs are APIs that comply with all the HTTP concepts you just learned—so you can GET, POST, PUT, DELETE, etc. to them

GET	/movies	Get list of movies
GET	/movies/:id	Find a movie by its ID
POST	/movies	Create a new movie
PUT	/movies	Update an existing movie
DELETE	/movies	Delete an existing movie

HTML, CSS, & JAVASCRIPT

- **HyperText Markup Language (HTML)** is a standard way for documents to be written on the web
 - It's standardized so that every browser knows how to interpret web pages when loading them
- **Cascading Style Sheets (CSS)** is another standard that allows you control the formatting of your web pages
- **JavaScript (JS)** is a web-oriented programming language allowing you to add interactive functionality to your web application

HTML, CSS, & JAVASCRIPT

- Each of these plays an important role on the web:
 - HTML = Document
 - CSS = Styling
 - JS = Bridge between document, styling, and interactive functionality



QUICK REVIEW OF HTML

- HTML gives us the tools to create the “content” of our document
- HTML is a standardized markup language that every browser can read
 - N.b. there are different specifications of HTML
 - The most recent one is HTML5, but it’s not even fully rolled out (e.g. Internet Explorer often can’t view HTML5-specific elements)

QUICK REVIEW OF HTML



QUICK REVIEW OF HTML

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>hello, world</title>
  </head>
  <body>
    <p style="color: red;">hello</p>
  </body>
</html>
```

← DOCTYPE

← BEGINNING OF HTML DOC

← HEAD

← BODY

← ELEMENT

QUICK REVIEW OF CSS

- Creating HTML documents raises the natural question: “How do I format these?”
- CSS offers us a solution to this problem by allowing us to create **style sheets** with which we can insert different styles to format our HTML
- The first ‘C’ in CSS stands for *cascading* - This is because you can include multiple style sheets in your document and CSS will figure out the “precedence” of your styles for you when they conflict

QUICK REVIEW OF CSS

```
<p style="color: red;">hello</p>
```

How might we refactor the above code?

QUICK REVIEW OF CSS

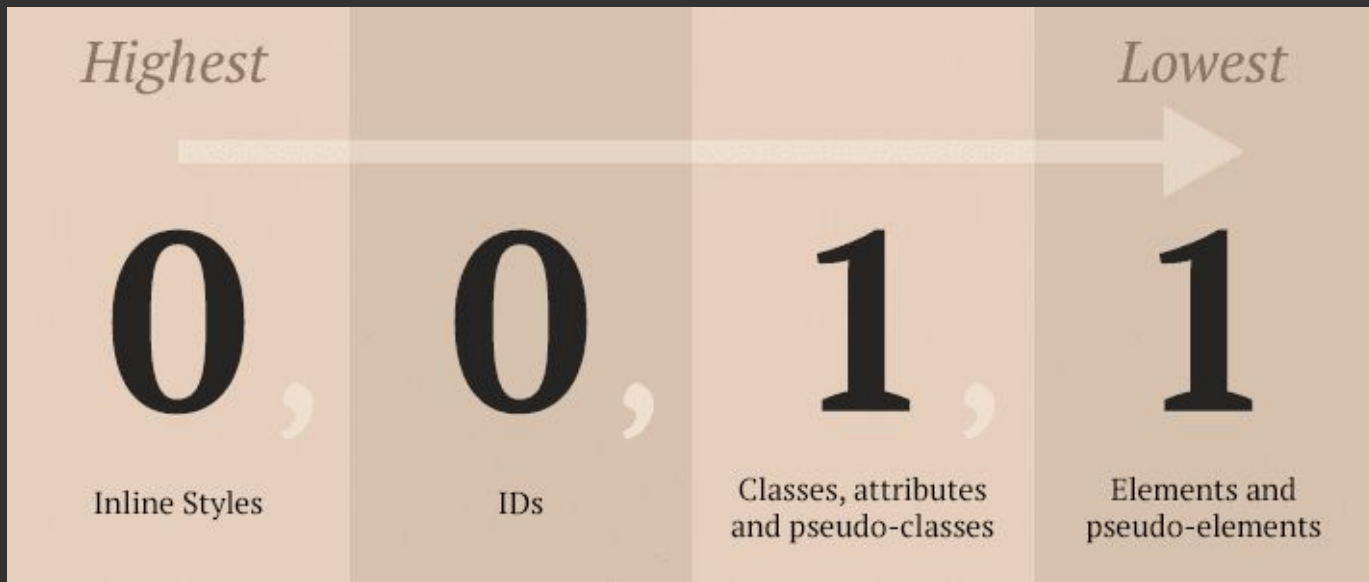
HTML CODE	CSS CODE
<code><p>hello</p></code>	<pre>p { color: red; }</pre> Targeting the type of tag
<code><p id="elem">hello</p></code>	<pre>#elem { color: red; }</pre> Targeting a specific ID
<code><p class="red">hello</p></code>	<pre>.red { color: red; }</pre> Targeting an entire class

QUICK REVIEW OF CSS

- There are different “best practices” for when we use a specific approach over another
 - Tag Selector - You want a particular element to always look a certain way
 - ID Selector - You want a single element to look a specific way (you generally avoid giving different elements the same ID)
 - Class Selector - You want different elements (e.g. `<a>`, `<p>`, etc) to all look a certain way

QUICK REVIEW OF CSS

- Note that CSS will handle conflicts for you in terms of “precedence”



BOOTSTRAP

- **Bootstrap** is a front-end framework developed by Twitter
- It's *massively* popular and you've probably seen instances of it all over the web
- It was originally designed to ensure internal consistency among applications at Twitter, but most web developers rely upon it to design **responsive web applications** using its grid

BOOTSTRAP

COL-3				COL-3				COL-3				COL-3					
COL-4						COL-4						COL-4					
COL-6								COL-6									
COL-2			COL-2			COL-2			COL-2			COL-2			COL-2		
COL-1	COL-1	COL-1	COL-1	COL-1	COL-1	COL-1	COL-1	COL-1	COL-1	COL-1	COL-1	COL-1	COL-1	COL-1	COL-1		

HOW DO I EVEN GET STARTED USING ANY OF THIS?

Include Bootstrap into your code:

```
<link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/css/bootstrap.min.css"
integrity="sha384-MCw98/SFnGE8fJT3GXwEOngsV7Zt27NXFoaoApmYm81iuXoPkFOJwJ8ERdknLPMO"
crossorigin="anonymous">
<script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTElPi6jizo"
crossorigin="anonymous"></script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.3/umd/popper.min.js"
integrity="sha384-ZMP7rVo3mIykV+2+9J3UJ46jBk0WLaUAdn689aCwoqBBJiSnjAK/18WvCWPIpM49"
crossorigin="anonymous"></script>
<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.3/js/bootstrap.min.js"
integrity="sha384-ChfqqxuZUCnJSK3+MXmPNIyE6ZbWh2IMqE241rYiqJxyMiZ6OW/JmZQ5stwEULTy"
crossorigin="anonymous"></script>
```

Include any custom CSS you want as well:

```
<link rel="stylesheet" href="custom.css">
```

BROWSER DEVELOPER TOOLS

- Most browsers include some sort of developer tools to aid in web development
 - By far, Google Chrome and Firefox offer the most robust and full-featured ones

BROWSER DEVELOPER TOOLS

- Most browsers include some sort of developer tools to aid in web development
 - By far, Google Chrome and Firefox offer the most robust and full-featured ones

DEMO TIME

INTERACTIVE ACTIVITY

Use developer tools to visit a website and modify its HTML and CSS in some way. Be ready to share what your remix!

JAVASCRIPT

- JavaScript allows us to add interactivity to our web applications, create dynamic interfaces, and manipulate HTML/CSS in complicated ways
- JavaScript is a **weakly typed** language—Unlike C, it doesn't ask for you to specify a variable type
- It typically runs **client-side**, but server-side JS (`node.js`) has exploded in popularity in recent years

JAVASCRIPT - VARIABLE ASSIGNMENT

```
// a simple variable
```

```
let age = 19;
```

```
// an array
```

```
let array = [1, 2, 3, 4, 5];
```

```
// string
```

```
let str = "Happy birthday, Maria!";
```

```
// an object
```

```
let teacher = {name: "David", course: 50};
```

JAVASCRIPT - CONDITIONS AND LOOPS

```
// while loop
while (true)
{
    // do something
}
```

```
// for loop
for (initialization; condition; update)
{
    // do something
}
```

```
// if condition
if (true)
{
    // do something
}
```

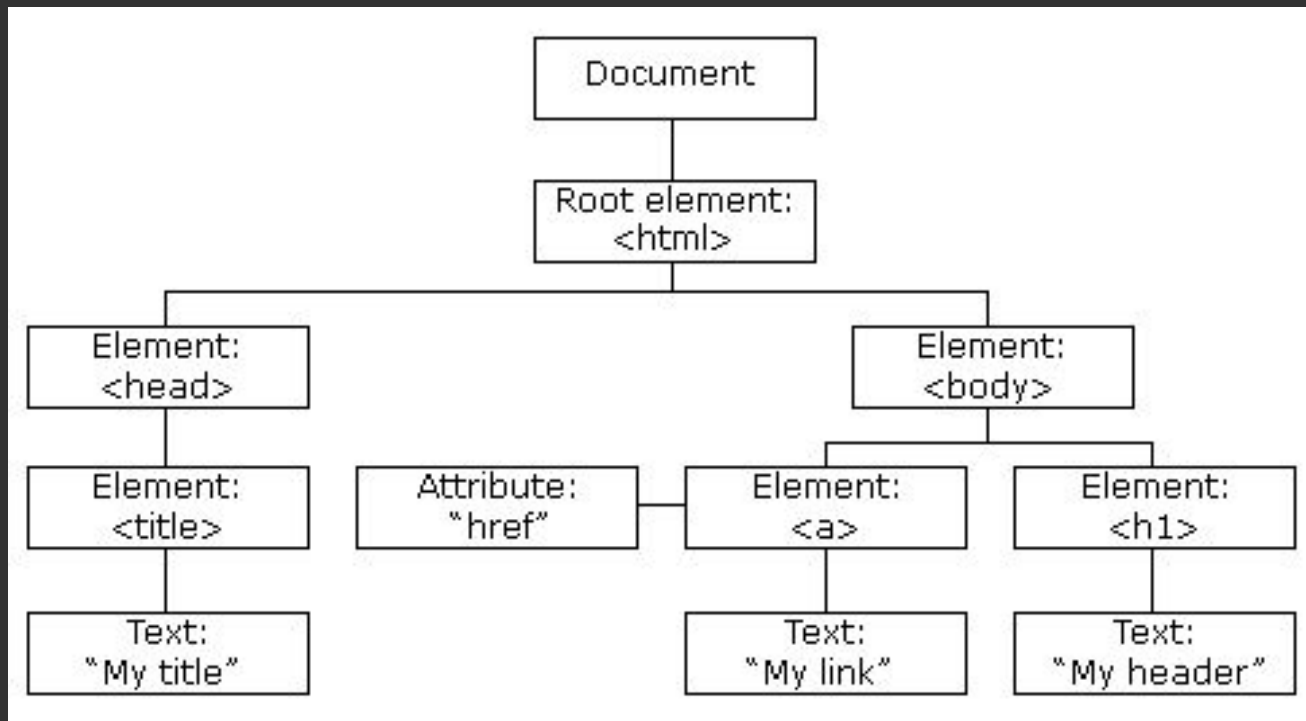

JAVASCRIPT

- On its own, JS is just like C: We can write programs in it, perform calculations, etc.
- Where JS becomes powerful is in how it interacts with the web ecosystem
 - We can use JS to *actively* modify HTML/CSS through the **Document Object Model (DOM)**
 - JS also includes a powerful **event listener** toolset to *respond* to changes in the DOM

DOCUMENT OBJECT MODEL

- Every HTML document is **parsed** by a browser into a hierarchical DOM and then rendered to the user
- We can access the DOM using JavaScript and begin to modify it dynamically as we wish

DOCUMENT OBJECT MODEL



MODIFYING THE DOM WITH JAVASCRIPT

```
let elem =  
document.querySelector("p");
```

Retrieves a single node from JavaScript using a query selector

```
let elems =  
document.querySelectorAll("p");
```

Retrieves *all* applicable nodes from JavaScript using a query selector

MODIFYING THE DOM WITH JAVASCRIPT

- Once we have selected a node using JS, we can begin to access its attributes using dot notation and modify them:

```
let elem = document.querySelector("p");  
elem.innerHTML = "New paragraph contents.";
```

MODIFYING THE DOM WITH JAVASCRIPT

- We might also want to traverse over multiple elements we've selected:

```
let elems = document.querySelectorAll("p");

elems.forEach(function(item) {
    item.innerHTML = "New HTML";
});
```

MODIFYING THE DOM WITH JAVASCRIPT

- We might also want to traverse over multiple elements we've selected:

Returns a collection
of nodes, similar to
an array

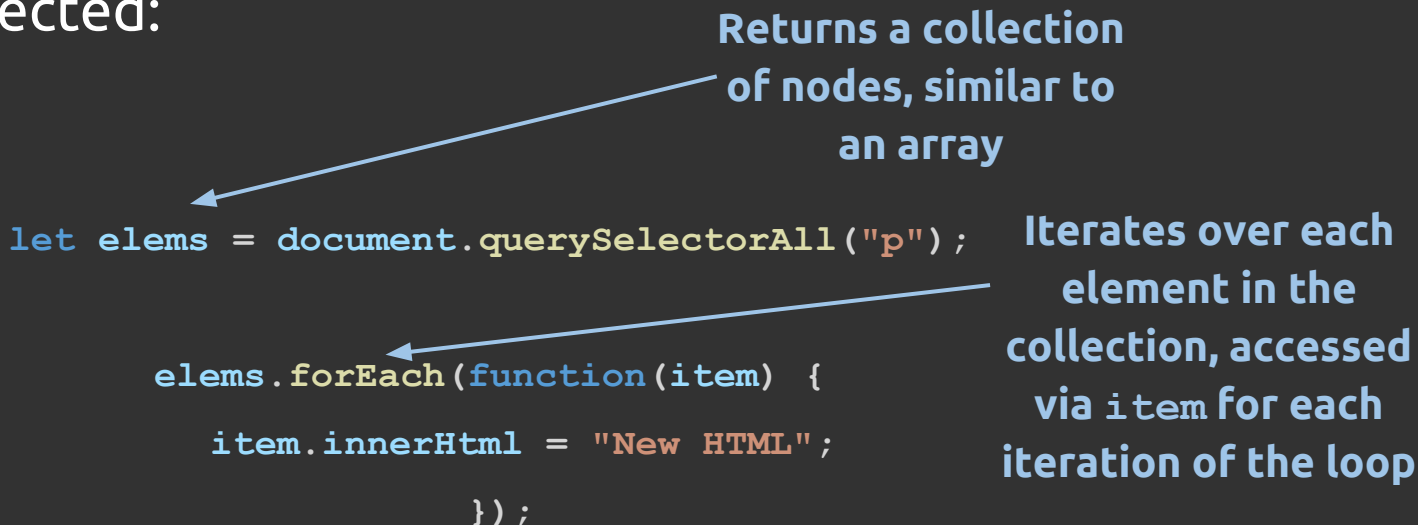


```
let elems = document.querySelectorAll("p");
```

```
elems.forEach(function(item) {  
    item.innerHTML = "New HTML";  
});
```

MODIFYING THE DOM WITH JAVASCRIPT

- We might also want to traverse over multiple elements we've selected:



The diagram illustrates the process of traversing multiple DOM elements. It features two lines of JavaScript code. The first line, `let elems = document.querySelectorAll("p");`, is annotated with an arrow pointing to the text "Returns a collection of nodes, similar to an array". The second line, `elems.forEach(function(item) { item.innerHTML = "New HTML"; });`, is annotated with an arrow pointing to the text "Iterates over each element in the collection, accessed via item for each iteration of the loop".

```
let elems = document.querySelectorAll("p");
```

Returns a collection of nodes, similar to an array

```
elems.forEach(function(item) {  
    item.innerHTML = "New HTML";  
});
```

Iterates over each element in the collection, accessed via item for each iteration of the loop

COMMON DOM PROPERTIES

PROPERTY

DESCRIPTION

`innerHTML`

Holds the HTML inside a set of HTML tags.

`nodeName`

The name of an HTML element or element's attribute.

`id`

The "id" attribute of an HTML element

`parentNode`

A reference to the node one level up in the DOM.

`childNodes`

An array of references to the nodes one level down in the DOM.

`attributes`

An array of attributes of an HTML element.

`style`

An object encapsulating the CSS/HTML styling of an element.

INTERACTIVE ACTIVITY

Use developer tools to inject a JavaScript script of your own creation into a web page. It should utilize a `forEach()` loop and perform some sort of DOM manipulation systematically of your choice.

CS50 Operators

Overview	Key Terms
<p>You're probably familiar with operators from math: the + symbol means addition, the - symbol means subtraction, etc. C also has operators, which you can use to modify or combine values. In addition to having operators that perform basic mathematical operations like addition, subtraction, multiplication, and division, C also has operators that perform other functions, like finding the remainder when dividing, or updating the</p>	<ul style="list-style-type: none"> • operator • arithmetic operators • assignment operators

Arithmetic Operators

1	$x = z + 8;$	20	C arithmetic operators perform mathematical functions on numbers. The + operator adds two numbers, the - operator subtracts one number from another, the * operator multiplies numbers, and the / operator divides one number by another. See the + and - symbols discussed in the next paragraph.
2	$x = x + 10;$	21	Through 4 of the code in the left is to see how such operations are performed when working with data and avoiding it's especially important to make sure that the result of the operation is an integer value. For instance, in line 1, we want to store the value of $10 + 3$. C++ sees a division of 10 by 3, and so it makes the result of the operation an integer as well. So the "real" value of $10 + 3$ is a floating number, something after the decimal point. But we want an integer, so we just want 13, in order to avoid the floating point error. So we can use the integer value to avoid the floating point error.
3	$x = x * 2;$	28	
4	$x = x / 2;$		
5	$x = x \% 2;$		
6	$x = x \& 3;$		

ROUTERS

the first operand is the remainder when the number on the left of the operator is divided by the number on the right. Line 5 demonstrates the modulus operator. The remainder when dividing 10 by 2 is 0, so the value of x is 0.

Assignment Operators

[illegible]

CS50 Operators

Overview	Key Terms
<p>You're probably familiar with operators from math: the + symbol means addition, the - symbol means subtraction, etc. C also has operators, which you can use to modify or combine values. In addition to having operators that perform basic mathematical operations like addition, subtraction, multiplication, and division, C also has operators that perform other functions, like finding the remainder when dividing, or updating the</p>	<ul style="list-style-type: none"> • operators • arithmetic operators • assignment operators • increment operators

Arithmetic Operators

[illegible]

Assignment Operators

<p>Assignment operators (such as <code>=</code>) provide a variety of ways to update the value of a variable. The standard assignment operator (<code>=</code>) is demonstrated in the following example. The value of <code>total</code> is updated on the right side of the equals sign; in this case, the current value of <code>total</code> is added to <code>5</code>.</p>	$7 + 5 = 12$	2
<p>The variable you're assigning can also be on the right of the equals sign (swap). On line 6, the value of <code>total</code> is put on the positive value of <code>total</code> plus one.</p>		
<p><code>++</code> and <code>--</code> might not seem like logical ones in algebra, it's valid in C++ because the value of a variable can be changed by adding or subtracting one.</p>		
<p>Note that C++ uses an infix syntax for the <code>++</code> operators: <code>total++</code> or <code>total--</code> are valid, but <code>++total</code> and <code>--total</code> are invalid, as they don't follow the rule of placing the variable to be modified by some other number.</p>		
<p>C++ also includes special syntax for increasing the value of a variable by one or decreasing the value of a variable by one, by writing the name of the variable followed by <code>++</code> or <code>--</code>.</p>		

CS50 Operators

Overview	Key Terms
<p>You're probably familiar with operators from math: the + symbol means addition, the - symbol means subtraction, etc. C also has operators, which you can use to modify or combine values. In addition to having operators that perform basic mathematical operations like addition, subtraction, multiplication, and division, C also has operators that perform other functions, like finding the remainder when dividing, or updating the</p>	<ul style="list-style-type: none"> • operator • arithmetic operator • assignment operator

Arithmetic Operators

1	$3 \times 2 = 6$	10	3 arithmetic operators perform mathematical functions on numbers. The * operator adds two numbers, the + operator subtracts one number from another, the ^ operator multiplies two numbers, and the / symbol divides one number by another. See Figure 4-1 through 4-7 for the order to use these symbols.
2	$3 \times 3 = 9$	7	
3	$3 \times 4 = 12$	1	When entering with left and dividing, it's especially important to be aware that an integer cannot have a non-integer value. The value in 6, for example, is not a valid 3.0. If 3.0 is used as a division of 6, the result will be 2.0, as if the number 6 were entered as 6.0 and the value were 2.0. 3.0 is a valid number, everything after the decimal point is included, and is set to just 3.0, so 6.0/3.0 is included, we would need to use $= 6.0 / 3.0$.
4	$3 \times 5 = 15$	4	
5	$3 \times 6 = 18$	5	
6	$3 \times 7 = 21$	6	
7	$3 \times 8 = 24$	7	
8	$3 \times 9 = 27$	8	
9	$3 \times 10 = 30$	9	
10	$3 \times 11 = 33$	10	
11	$3 \times 12 = 36$	11	
12	$3 \times 13 = 39$	12	
13	$3 \times 14 = 42$	13	
14	$3 \times 15 = 45$	14	
15	$3 \times 16 = 48$	15	
16	$3 \times 17 = 51$	16	
17	$3 \times 18 = 54$	17	
18	$3 \times 19 = 57$	18	
19	$3 \times 20 = 60$	19	
20	$3 \times 21 = 63$	20	
21	$3 \times 22 = 66$	21	
22	$3 \times 23 = 69$	22	
23	$3 \times 24 = 72$	23	
24	$3 \times 25 = 75$	24	
25	$3 \times 26 = 78$	25	
26	$3 \times 27 = 81$	26	
27	$3 \times 28 = 84$	27	
28	$3 \times 29 = 87$	28	
29	$3 \times 30 = 90$	29	
30	$3 \times 31 = 93$	30	
31	$3 \times 32 = 96$	31	
32	$3 \times 33 = 99$	32	
33	$3 \times 34 = 102$	33	
34	$3 \times 35 = 105$	34	
35	$3 \times 36 = 108$	35	
36	$3 \times 37 = 111$	36	
37	$3 \times 38 = 114$	37	
38	$3 \times 39 = 117$	38	
39	$3 \times 40 = 120$	39	
40	$3 \times 41 = 123$	40	
41	$3 \times 42 = 126$	41	
42	$3 \times 43 = 129$	42	
43	$3 \times 44 = 132$	43	
44	$3 \times 45 = 135$	44	
45	$3 \times 46 = 138$	45	
46	$3 \times 47 = 141$	46	
47	$3 \times 48 = 144$	47	
48	$3 \times 49 = 147$	48	
49	$3 \times 50 = 150$	49	
50	$3 \times 51 = 153$	50	
51	$3 \times 52 = 156$	51	
52	$3 \times 53 = 159$	52	
53	$3 \times 54 = 162$	53	
54	$3 \times 55 = 165$	54	
55	$3 \times 56 = 168$	55	
56	$3 \times 57 = 171$	56	
57	$3 \times 58 = 174$	57	
58	$3 \times 59 = 177$	58	
59	$3 \times 60 = 180$	59	
60	$3 \times 61 = 183$	60	
61	$3 \times 62 = 186$	61	
62	$3 \times 63 = 189$	62	
63	$3 \times 64 = 192$	63	
64	$3 \times 65 = 195$	64	
65	$3 \times 66 = 198$	65	
66	$3 \times 67 = 201$	66	
67	$3 \times 68 = 204$	67	
68	$3 \times 69 = 207$	68	
69	$3 \times 70 = 210$	69	
70	$3 \times 71 = 213$	70	
71	$3 \times 72 = 216$	71	
72	$3 \times 73 = 219$	72	
73	$3 \times 74 = 222$	73	
74	$3 \times 75 = 225$	74	
75	$3 \times 76 = 228$	75	
76	$3 \times 77 = 231$	76	
77	$3 \times 78 = 234$	77	
78	$3 \times 79 = 237$	78	
79	$3 \times 80 = 240$	79	
80	$3 \times 81 = 243$	80	

Assignment Operators

<p>2 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100</p>	<p>1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100</p>	<p>1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100</p>	<p>1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100</p>	<p>1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100</p>	<p>1 2 3 4 5 6 7 8 9 10 11 12 13 14</p>
---	--	--	--	--	---

CS50 Operators

Overview	Key Terms
<p>You're probably familiar with operators from math: the + symbol means addition, the - symbol means subtraction, etc. C also has operators, which you can use to modify or combine values. In addition to having operators that perform basic mathematical operations like addition, subtraction, multiplication, and division, C also has operators that perform other functions like finding the remainder when dividing, or updating the</p>	<ul style="list-style-type: none"> operator arithmetic operator assignment operator assignment operator

Arithmetic Operators

[illegible]

Assignment Operators

7	$a \leftarrow a + b$	7	$a \leftarrow a + b$
8	assignment operator, which provide a variety of ways to update the value of a variable. The standard assignment operator (\leftarrow) is demonstrated on line 7.	8	assignment operator, which provide a variety of ways to update the value of a variable. The standard assignment operator (\leftarrow) is demonstrated on line 7.
9	On the right side of the equals sign, in this case, the current value of b is added to a .	9	On the right side of the equals sign, in this case, the current value of b is added to a .
10		10	
11	The variable b we assigned can be used on the right of the equals sign. For example, on line 11, the value of b is set to the square of a , plus one.	11	The variable b we assigned can be used on the right of the equals sign. For example, on line 11, the value of b is set to the square of a , plus one.
12	While $a \leftarrow a + b$ might not make logical sense in algebra, it's valid in C. It's reading the value of a , adding b to it, and then storing the result back into a .	12	While $a \leftarrow a + b$ might not make logical sense in algebra, it's valid in C. It's reading the value of a , adding b to it, and then storing the result back into a .
13	Remember that C has special syntax for the operators $++$ and $--$, which let a variable be incremented value plus/minus, modified by, or set to by some other number.	13	Remember that C has special syntax for the operators $++$ and $--$, which let a variable be incremented value plus/minus, modified by, or set to by some other number.
14	C also includes special syntax for increasing the value of a variable by one or decreasing the value of a variable by one, by using the range of operators indicated by the arrows.	14	C also includes special syntax for increasing the value of a variable by one or decreasing the value of a variable by one, by using the range of operators indicated by the arrows.

CS50 Operators

Overview	Key Terms
You're probably familiar with operators from math: the + symbol means addition, the - symbol means subtraction, etc. C also has operators, which you can use to modify or combine values. In addition to having operators that perform basic mathematical operations like addition, subtraction, multiplication, and division, C also has operators that perform other functions, like finding the remainder when dividing, or updating the	<ul style="list-style-type: none"> • operator • arithmetic operator • assignment operator

Arithmetic Operators

[illegible]

Assignment Operators

[illegible]

CS50 Operators

Overview You're probably familiar with **operators** from math: the + symbol means addition; the - symbol means subtraction, etc. C also has operators, which you can use to modify or combine values. In addition to having operators that perform basic mathematical operations (like addition, subtraction, multiplication, and division), C also has operators that perform other functions, like finding the remainder when dividing, or updating the

Arithmetic Operators

1	$\text{let } x = 2;$	10	C arithmetic operators perform mathematical functions on numbers. The <code>+</code> operator adds two numbers, the <code>-</code> operator subtracts a number from another, the <code>*</code> operator multiplies two numbers, and the <code>/</code> symbol divides one number by another. See through 4 of the code in the left to see how each operator is used.
2	$\text{let } x = 10 * 3;$	20	When working with <code>let</code> and <code>dividing</code> , it's especially important to be aware that <code>let</code> does not have non-trivial values. For instance, in line 7, we try to take the value of <code>x</code> , <code>x = 3</code> , and <code>divide</code> it by 2. This results in a modular value, an integer or a real value, and the "value" value is <code>1.5</code> . <code>2</code> is a symbol, number, modular value, and so on.
3	$\text{let } x = x + 4;$	28	
4	$\text{let } x = 4 * 3;$	3	
5	$\text{let } x = 12;$	3	
6	$\text{let } x = 13 * 3;$	3	

JAVASCRIPT

Assignment Operators

[illegible]

https://www.dropbox.com/sh/5y662ey1hc4sde4/AACjgHN3NtSKk4ShsRDFd_Sj_a?dl=0&m=&preview=TCP+and+P.pdf

https://www.dropbox.com/sh/5y662ey1hc4sde4/AACiqHN3NtSKk4ShsRDFd_Sja?dl=0&m=&preview=HTML.pdf

https://www.dropbox.com/sh/5y662ey1hc4sde4/AACjgHN3NtSKk4ShsRDFd_Sj_a?dl=0&m=&preview=JavaScript.pdf

LOOKING AHEAD

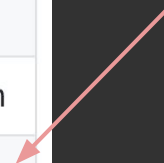
LOOKING AHEAD

Problem Set	Language	Released	Due
Problem Set 0	Scratch	Fri 9/7	Tue 9/11, 11:59pm
Problem Set 1	C	Fri 9/14	Thu 9/20, 11:59pm
Problem Set 2	C	Fri 9/21	Thu 9/27, 11:59pm
Problem Set 3	C	Fri 9/28	Thu 10/4, 11:59pm
Problem Set 4	C	Fri 10/5	Thu 10/11, 11:59pm
Problem Set 5	HTML, CSS	Fri 10/12	Tue 10/16, 11:59pm
Problem Set 6	Python	Fri 10/19	Thu 10/25, 11:59pm
Problem Set 7	Python, HTML, CSS	Fri 10/26	Thu 11/1, 11:59pm
Problem Set 8	SQL, Python, HTML, CSS	Fri 11/2	Thu 11/8, 11:59pm

LOOKING AHEAD

Problem Set	Language	Released	Due
Problem Set 0	Scratch	Fri 9/7	Tue 9/11, 11:59pm
Problem Set 1	C	Fri 9/14	Thu 9/20, 11:59pm
Problem Set 2	C	Fri 9/21	Thu 9/27, 11:59pm
Problem Set 3	C	Fri 9/28	Thu 10/4, 11:59pm
Problem Set 4	C	Fri 10/5	Thu 10/11, 11:59pm
Problem Set 5	HTML, CSS	Fri 10/12	Tue 10/16, 11:59pm
Problem Set 6	Python	Fri 10/19	Thu 10/25, 11:59pm
Problem Set 7	Python, HTML, CSS	Fri 10/26	Thu 11/1, 11:59pm
Problem Set 8	SQL, Python, HTML, CSS	Fri 11/2	Thu 11/8, 11:59pm

**YOU ARE
HERE**



LOOKING AHEAD

Problem Set	Language	Released	Due
Problem Set 0	Scratch	Fri 9/7	Tue 9/11, 11:59pm
Problem Set 1	C	Fri 9/14	Thu 9/20, 11:59pm
Problem Set 2	C	Fri 9/21	Thu 9/27, 11:59pm
Problem Set 3	C	Fri 9/28	Thu 10/4, 11:59pm
Problem Set 4	C	Fri 10/5	Thu 10/11, 11:59pm
Problem Set 5	HTML, CSS	Fri 10/12	Tue 10/16, 11:59pm
Problem Set 6	Python	Fri 10/19	Thu 10/25, 11:59pm
Problem Set 7	Python, HTML, CSS	Fri 10/26	Thu 11/1, 11:59pm
Problem Set 8	SQL, Python, HTML, CSS	Fri 11/2	Thu 11/8, 11:59pm

**MORE
WEB-ORIENTED
THINGS TO
COME...**

**AND PYTHON
TOO**

REMINDER: GOALS OF THE COURSE

1. Learn to think computationally and algorithmically.
2. Improve your problem-solving skills—this will help you in every other course and in real-life too!
3. Gain practical programming skills in a plethora of languages (C, Python, JavaScript, etc.)
4. Familiarize yourself with best practices for software design.
5. Build a community with your peers that will stay with you after this course.

LOOKING AHEAD

- ★ Now is the time to think back to the goals you set for yourself at the beginning of the course
- ★ Ask yourself:
 - What did you want out of this class when you began? Have those things changed?
 - How are you performing in meeting the goals you set for yourself? How might you adjust as needed?
 - What do you like about the class? What do you find frustrating? What did you want to learn that you haven't yet?

LOOKING AHEAD

- ★ Since there is not problem set due later this week, spend the extra time you have to sit down and think intentionally about these reflection questions
- ★ *I'm always a resource for you—Reach out if you want to co-strategize, get advice, or just talk*