TEMPLE UNIVERSITY

Digital Systems

9.1 An Introduction to Programming

FOX MIS

An Introduction to Programming

Digital Product Management

FOX
MIS

Machines

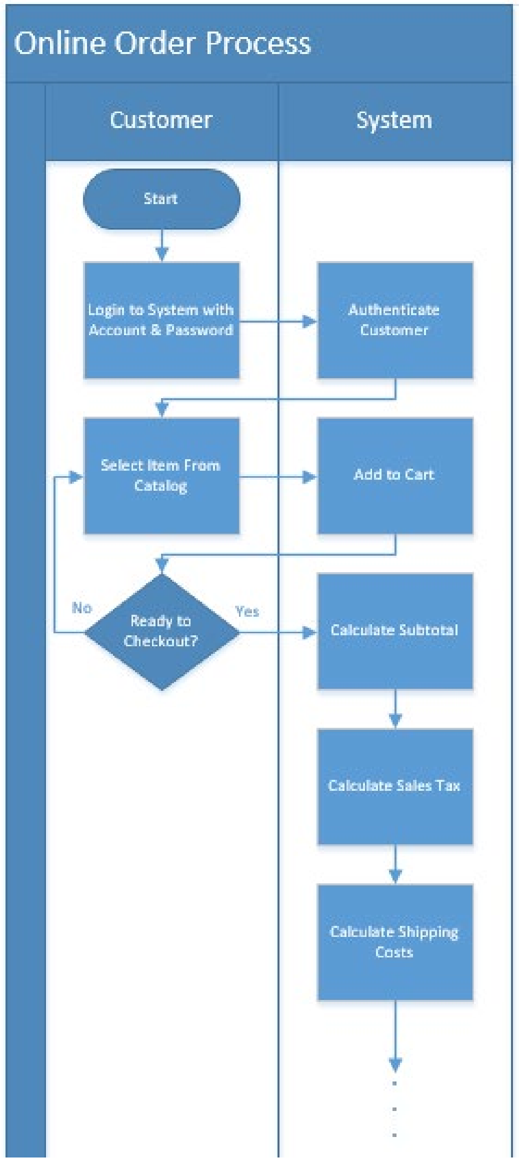
How many different things does a machine do?

Source: https://lh3.googleusercontent.com/_MsstYyQ4UORpFRdwNu6BN2_cL3vu_raiYq-RErSsdf_Y6v_lKZss8N-8qCO-Tnf3aalDw=s170
Source: https://lh3.googleusercontent.com/GB8PJSrB61OSat3rBAJO-wEjTGstYfB2N4ECL2LSYOahESElAVf7LWJMNrJ0undIP10o=s119
Source: https://lh3.googleusercontent.com/VVZxPvd2BpOPvECeBytaSZqFUfAg5UqPmeroft5ONc3bQVKYqr2s5Ixol4NvEP4unuJiFDI=s94
Source: https://lh3.googleusercontent.com/eZGdlrLtdzW9nEKR758LiagZ3_-wjxflYbpU-lfZeVdny34bXSgn4s0DQTsfS3t_O2eZ_g=s85
Source:
https://lh3.googleusercontent.com/OCVAphS1iC9nYvF2FeP_lINXwWQ69mCX8HjK2DJ7UU_xGnoHOgXd4YX8O4Zes09WIcIFxP8=s85
Source: https://lh3.googleusercontent.com/mCtpPS6Qz3guK0-cyCruwUwgbKYAJhWH1iTqGbCNjMOvSnGMvL1FeBGeDIBYboOC7P-gfQ=s102

Machines

How many different things does a computer do?

How can this machine do so many different things?



Source: https://lh3.googleusercontent.com/59wm5Z8eXC40_WKib3Ghe3ORRuSITa17XI5BN8QIikEh_cDnScrGnOouZnO4X-cBY6ad5w=s128

Source: https://lh3.googleusercontent.com/luQcPRIrni_WsHytr016U1SPu7FvyVIK2AfyBH3fXEjhoBPrmpzuTfgjvQV40UR63ZTq1jk=s134

Source: https://lh3.googleusercontent.com/Mkj8bl6o0XbrVqnFs40Y3AFVMibSJX1xra-LI5rwbMCxGs67gpljSny2TUI4KASK1u8sD-M=s85

Source: https://lh3.googleusercontent.com/dMhdY1wyo91ER5uMkI1recx4-0w7fOjNUNmSB9ZCgVnIMqqqXNDkxiw9ckuWij_vtzFSEw=s128

Source: https://lh3.googleusercontent.com/qu1kdMmaM8SbvpyiDQEULPMPGfCrHneGT7WZ494FhgM8N4SHp-TOQvmuKBK2leESFBSPUjg=s151

Software

Programs = Software = Apps

But aren't all programmers geeks?

What exactly is a program?

```
a.fn.scrollspy=d,this},a(window).on(
y),+function(a){"use strict";function b(b){return this.each(function(){
&e[b]())}}var c=function(b){this.element=a(b)};c.VERSION="3.3.7",c.TRANSITION_DURATION=150,c.prot
opdown-menu)"),d=b.data("target");if(d||(d=b.attr("href"),d=d&&d.replace(/.*(?=#[^\s]*$)/,"")),!
st a"),f=a.Event("hide.bs.tab",{relatedTarget:b[0]}),g=a.Event("show.bs.tab",{relatedTarget:e[0]
faultPrevented()){var h=a(d);this.activate(b.closest("li"),c),this.activate(h,h.parent(),functio
rigger({type:"shown.bs.tab",relatedTarget:e[0]})})}}},c.prototype.activate=function(b,d,e){func
u > .active").removeClass("active").end().find('[data-toggle="tab"]').attr("aria-expanded",!1),
ia-expanded",!0),h?(b[0].offsetWidth,b.addClass("in")):b.removeClass("fade"),b.parent(".dropdow
().find('[data-toggle="tab"]').attr("aria-expanded",!0),e&&e()}var g=d.find("> .active"),h=e&&
de")||!!d.find("> .fade").length);g.length&&h?g.one("bsTransitionEnd",f).emulateTransitionEnd
;var d=a.fn.tab;a.fn.tab=b,a.fn.tab.Constructor=c,a.fn.tab.noConflict=function(){return a.fn.t
"show")};a(document).on("click.bs.tab.data-api",'[data-toggle="tab"]',e).on("click.bs.tab.data
se strict";function b(b){return this.each(function(){var d=a(this),e=d.data("bs.affix"),f="ob
typeof b&&e[b]()})}var c=function(b,d){this.options=a.extend({},c.DEFAULTS,d),this.$target=a
",a.proxy(this.checkPosition,this)).on("click.bs.affix.data-api",a.proxy(this.checkPositionWi
null,this.pinnedOffset=null,this.checkPosition()};c.VERSION="3.3.7",c.RESET="affix affix-top
State=function(a,b,c,d){var e=this.$target.scrollTop(),f=this.$element.offset(),g=this.$targ
"bottom"==this.affixed)return null!=c?!(e+this.unpin<=f.top)&&"bottom":!(e+g<=a-d)&&"bottom"
!=c&&e<=c?"top":null!=d&&i+j>=a-d&&"bottom"},c.prototype.getPinnedOffset=function(){if(this
.RESET).addClass("affix");var a=this.$target.scrollTop(),b=this.$element.offset();return
WithEventLoop=function(){setTimeout(a.proxy(this.checkPosition,this),1)}
nt.height(),d=this.options.offset,e=d.top,f=d.bottom
peof e&&(e=d.top(this.$element)),"
nt.css("top","")
```

Source: https://lh3.googleusercontent.com/[illegible]ToGjynKLosbP2EQ=s114

Source: https://www.wvxu.org/sites/wvxu/files/styles/x_large/public/201802/Algorithm.jpeg

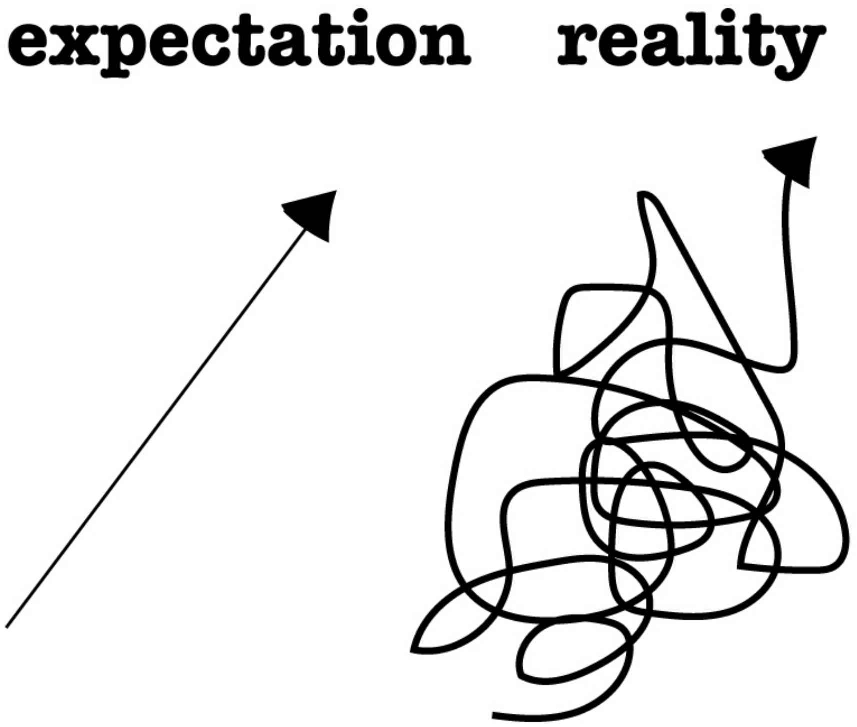
How to learn how to program

- Program
- Program some more
- Program more after that
- Delete everything and program again!

WE'RE GOING TO HAVE TO ASK YOU TO COME IN THIS WEEKEND AND PRACTICE MORE CODING

Source: Photofest: https://www.hollywoodreporter.com/review/office-space-review-1999-movie-1086336

Managing Expectations



https://www.futuredesigncoaching.com/single-post/2017/12/12/When-Expectation-and-Reality-are-Misaligned

Hello World!

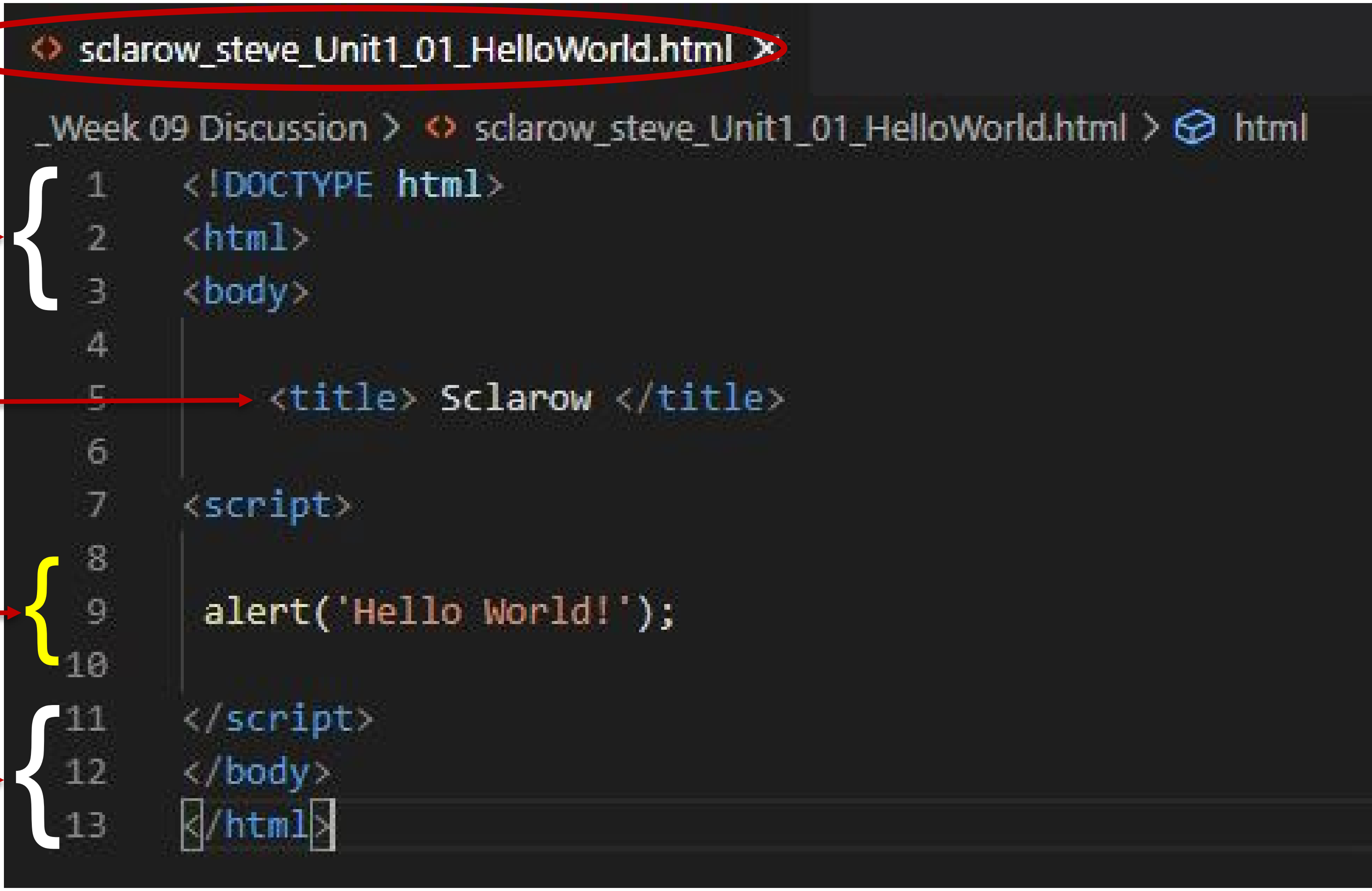
Keep track of where you store your code!

This is the HTML and never changes

Rename your title

This is the JavaScript code

This is the HTML and never changes


sclarow_steve_Unit1_01_HelloWorld.html
_Week 09 Discussion > sclarow_steve_Unit1_01_HelloWorld.html > html

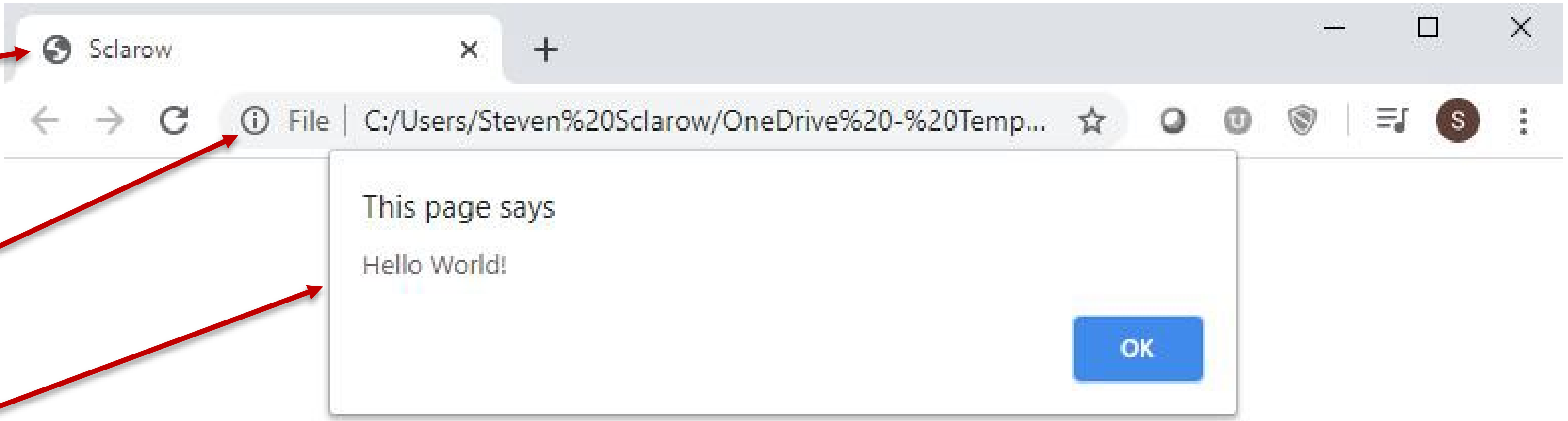
```
<!DOCTYPE html>
<html>
<body>

    <title> Sclarow </title>

<script>

 alert('Hello World!');

</script>
</body>
</html>
```

Hello World!

This is the Title

Keep track of where you store your code!

This is the output from Alert



This page says

Hello World!

Coding Tools

- We will use Visual Studio Code text editor
  - (which you loaded at the beginning of the semester)
    - Installing-VS-Code-Windows
    - Installing-VS-Code-Mac-OS
- We need a browser to view our work.
  - Make CHROME your default browser

File Download: PC

- Create a folder on your hard drive
  - This is where your will save all of your coding files!
- Visit course site for the coding files
- Download each week's coding files into your new folder
  - You may need to "unzip" the files and extract them into your folder:
    - Mac help (just google it!)
    - PC help (just google it!)

File Download: Mac

- Create a folder on your hard drive
  - This is where your will save all of your coding files!
- Visit course site for the coding files
- Download each week's coding files into your new folder
  - You may need to "unzip" the files and extract them into your folder:
    - Mac help [(just google it!)]
    - PC help [(just google it!)]

File Naming Convention:

- lastname_firstname_Unit1_01_HelloWorld.html
    - Sample renaming: doyle_mart_Unit1_01_HelloWorld.html
- Properly naming your file is very important!
    - Improperly named files will not receive points/credit (no exceptions)
- Always check that you are saving your files to the correct location
- Always verify that you are saving an .html file type

Challenges

- **HelloWorld**
- Address
- GuessANumber
- Profits
- LandCalculations
- TotalPurchases

Let’s Code Hello World!

In-Class Activity

lastname_firstname_Unit1_01_HelloWorld.html

Values and Variables

Digital Product Management

FOX
MIS

TIPS FROM MIS 2101 VIRTUAL HELPDESK

Programming is the "Reading, Writing and Arithmetic" of the Digital Age with Michelle Purnama

TEMPLE UNIVERSITY

Values

Variables

Values

In JavaScript, every piece of data that you provide or use is considered to contain a value.

Data Types:

- **Number**
  - ✓ **Integers** – Examples (6, 10, 67, 92)
  - ✓ **Floating Point Numbers** – Examples (5.2, 65.21, 87.64, 92.3)

Data Types:

- **Number**
  - ✓ **Integers** – Examples (6, 10, 67, 92)
  - ✓ **Floating Point Numbers** – Examples (5.2, 65.21, 87.64, 92.3)
- **String**
  - ✓ **Single quotes** – Examples ('hello', 'radius', 'area')
  - ✓ **Double quotes** – Examples ("hello", "radius", "area")

Data Types:

- **Number**
  - ✓ **Integers** – Examples (6, 10, 67, 92)
  - ✓ **Floating Point Numbers** – Examples (5.2, 65.21, 87.64, 92.3)
- **String**
  - ✓ **Single quotes** – Examples ('hello', 'radius', 'area')
  - ✓ **Double quotes** – Examples ("hello", "radius", "area")
- **Boolean**
  - ✓ The Boolean type has only two values: True and False.
  - ✓ This type is commonly used to store yes/no values: True means “yes, correct”, and False means “no, incorrect”.

We saw this example earlier…

Example:

```
alert("hello, world!");
```

Values

Because you'll be working with values a whole lot, there are **two things** you need to simplify your life when working with them. You need to be able to:

1. Easily identify them
2. Reuse them without unnecessarily duplicating the value itself

Variables

Variables

- Variables are used to store values to be used later in a program.
- They are called variables because their values can be changed.

How Do We Use Variables?

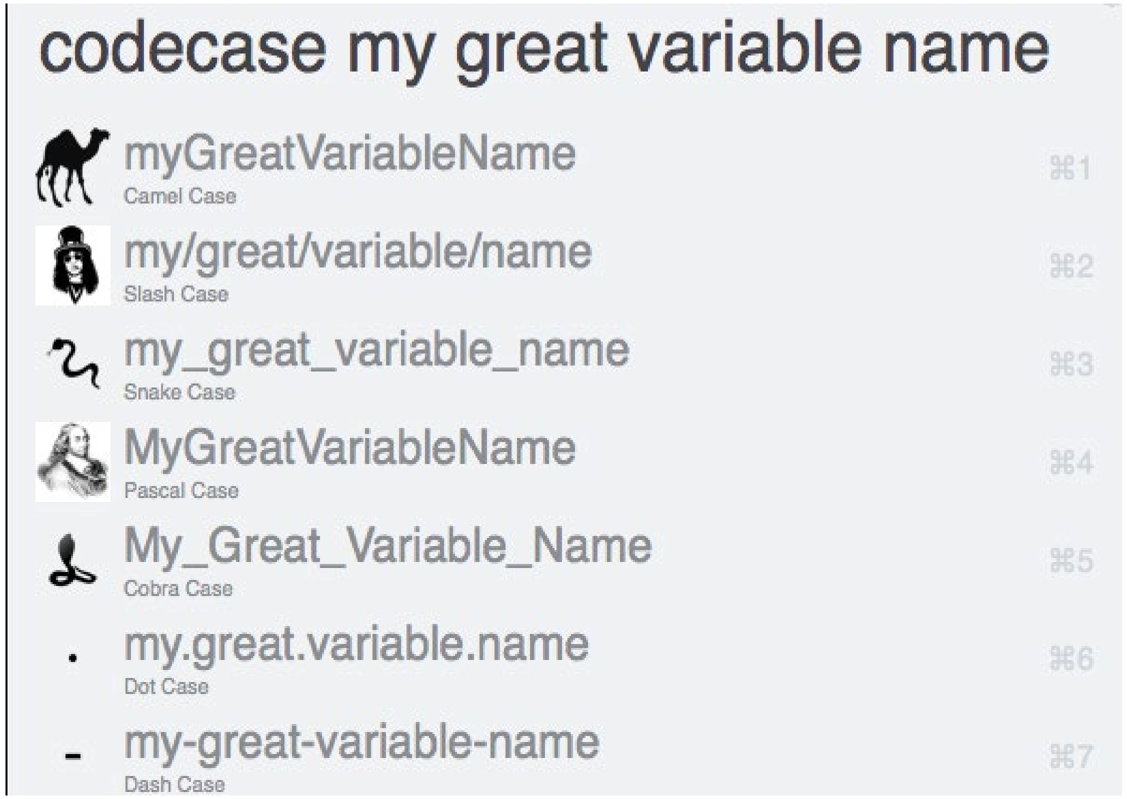
- **Step 1:** Choose a name for the variable
- **Step 2:** Declare the variable
- **Step 3:** Assign a value to the variable
- **Step 4:** Use the variable

How Do We Use Variables?

- **Step 1:** Choose a name for the variable

Naming Variables

You have a lot of freedom in naming your variables however you see fit. Ignoring what names you should give things based on philosophical / cultural / stylistic preferences, from a technical point of view, JavaScript is **very lenient** on what characters can go into a variable name.



codecase my great variable name

myGreatVariableName	Camel Case	⌘1
my/great/variable/name	Slash Case	⌘2
my_great_variable_name	Snake Case	⌘3
MyGreatVariableName	Pascal Case	⌘4
My_Great_Variable_Name	Cobra Case	⌘5
my.great.variable.name	Dot Case	⌘6
my-great-variable-name	Dash Case	⌘7

Source:
https://content.invisioncic.com/r229491/monthly_2018_02/codecase.png.9abc0521d2c0925906f5d0c309ed3f7b.png

Naming Variables

Consider the following guidelines:

1. Your variables can be as short one character, or they can be as long as you want - think thousands and thousands...and thousands of characters.

2. Your variables **can** start with a **letter**, **underscore**, or the **$** character. They **can't** start with a **number**.

3. Outside of the first character, your variables can be made up of any combination of letters, underscores, numbers, and $ characters. You can also mix and match lowercase and uppercase to your heart's content.

4. **Spaces** are **not** allowed.

Example 1: myText

Example 2: $

Example 3: r8

Example 4: _counter

Example 5: $field

Example 6: 4B

Example 7: __$abc;

Example 8: OldSchoolNamingScheme

Example 9: 6radius

Example 10: 8 area

How Do We Use Variables?

- **Step 1:** Choose a name for the variable
- **Step 2:** Declare the variable

Declaring Variables

The way to use variables is by using the **let** keyword followed by the name you want to give your variable. Here is us declaring a variable:

```
let myText
```

or

```
let yourName
```

versus

```
var oldSchoolDeclaration
```

(note: **var** is an older keyword used to declare variables, JavaScript is flexible and will still recognize **var**; however, for our class we will use **let**.)

How Do We Use Variables?

- **Step 1:** Choose a name for the variable
- **Step 2:** Declare the variable
- **Step 3:** Assign a value to the variable

Assigning Value to Variables

- Right now, your variable has simply been declared. It doesn't contain a value. We can fix that by initializing our variable to a value like...

- We can assign a value to a variable using an **assignment operator**.

- In JavaScript, the equal sign (**=**) is used as the **assignment operator.**

Assigning Value to Variables

Our variable **myText** has now been initialized to the value of **hello, world!**

```
let myText = “hello, world!”;
```

How Do We Use Variables?

- **Step 1:** Choose a name for the variable
- **Step 2:** Declare the variable
- **Step 3:** Assign a value to the variable
- **Step 4:** Use the variable

```
<script>
  let myText = "hello, world!";
  alert(myText);
</script>
```

The “assignment” operator

“alert()” is used to display information

What gets displayed is **hello, world!**

```
<script>
  let myText = “hi everybody!”;
  alert(myText);
</script>
```

What gets displayed now?

```
<script>
  let myText = prompt("What text would you like to display? ");
  alert(myText);
</script>
```

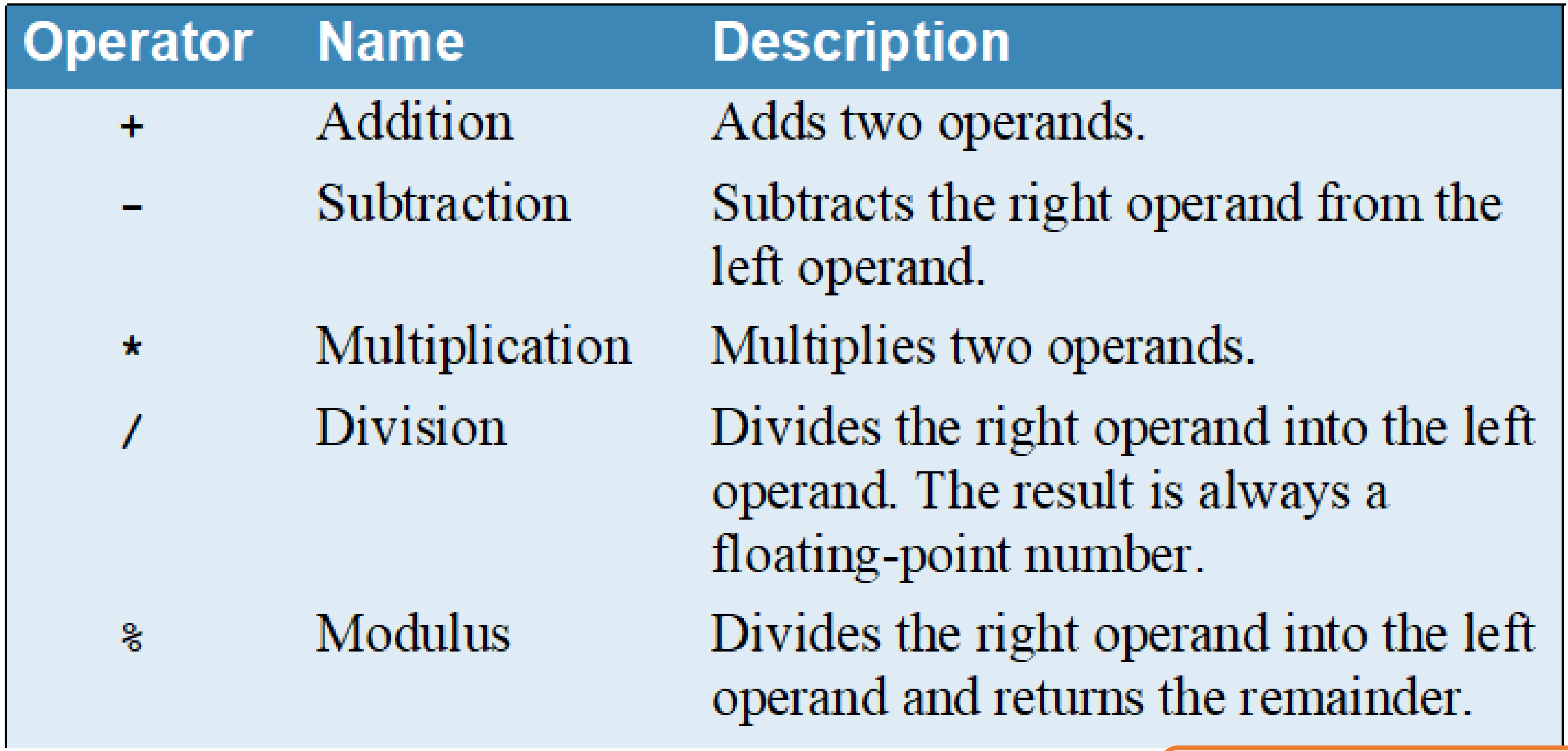
"prompt()" lets you get input from the user

What gets displayed now?

- Throughout your code, wherever you referenced the **myText** variable, you will now see the new text appear.

- For larger applications, this convenience with having just one location where you can make a change that gets reflected everywhere is a major time saver.

Arithmetic operators

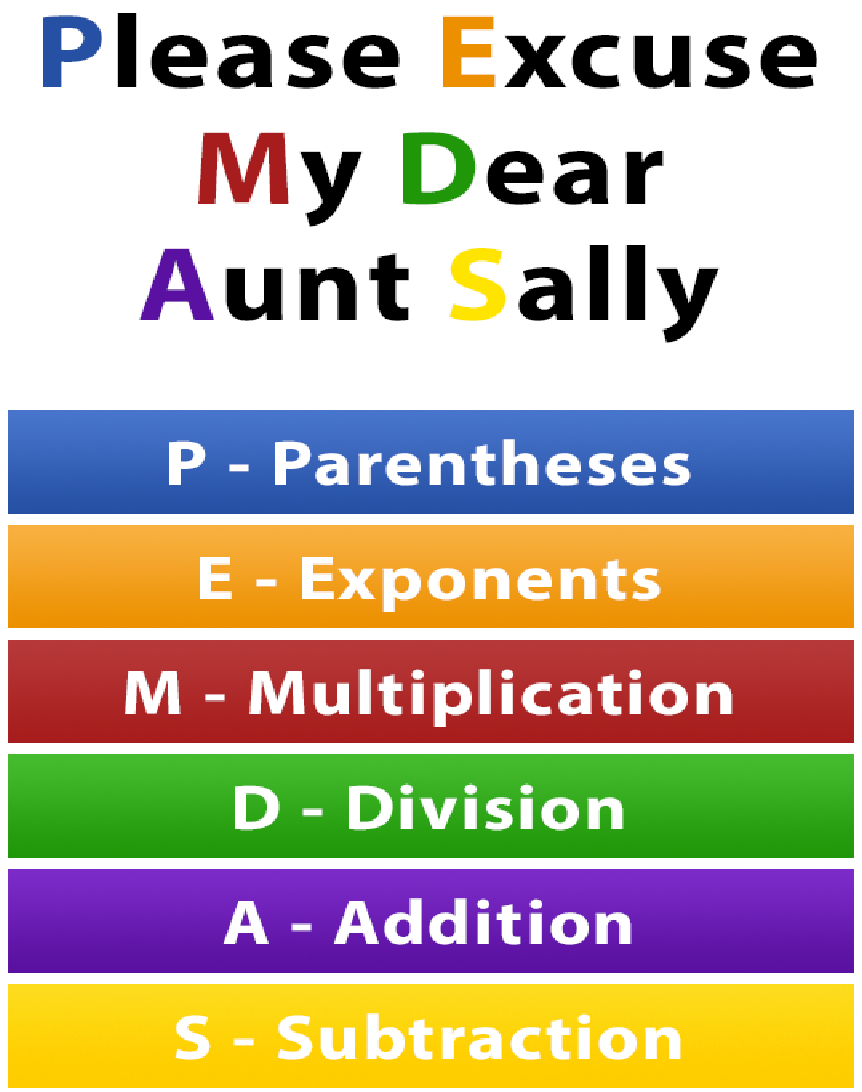


Operator	Name	Description
+	Addition	Adds two operands.
-	Subtraction	Subtracts the right operand from the left operand.
*	Multiplication	Multiplies two operands.
/	Division	Divides the right operand into the left operand. The result is always a floating-point number.
%	Modulus	Divides the right operand into the left operand and returns the remainder.

WARNING! This is ***not*** a complete list.

The Order of Operations

- Why is it important to remember PEMDAS while coding?
  - Because JavaScript follows these rules!



Please Excuse
My Dear
Aunt Sally

P - Parentheses
E - Exponents
M - Multiplication
D - Division
A - Addition
S - Subtraction

Source: https://www.pinterest.com/pin/10344274128829700/

```
subtotal = 200;
taxPercent = .05;
taxAmount = subtotal * taxPercent;              // 10
total = subtotal + taxAmount;                   // 210
```

Code that calculates the perimeter of a rectangle

```
width = 4.25;
length = 8.5;
perimeter = (2 * width) + (2 * length)
                                    // (8.5 + 17) = 25.5
```

Concatenation operator

Operator	Description
+	Concatenates two values.
+=	Adds the result of the expression to the end of the variable.

Just a fancy name for putting two strings together.

Concatenation is a very common task!

How to concatenate string variables with the + operator

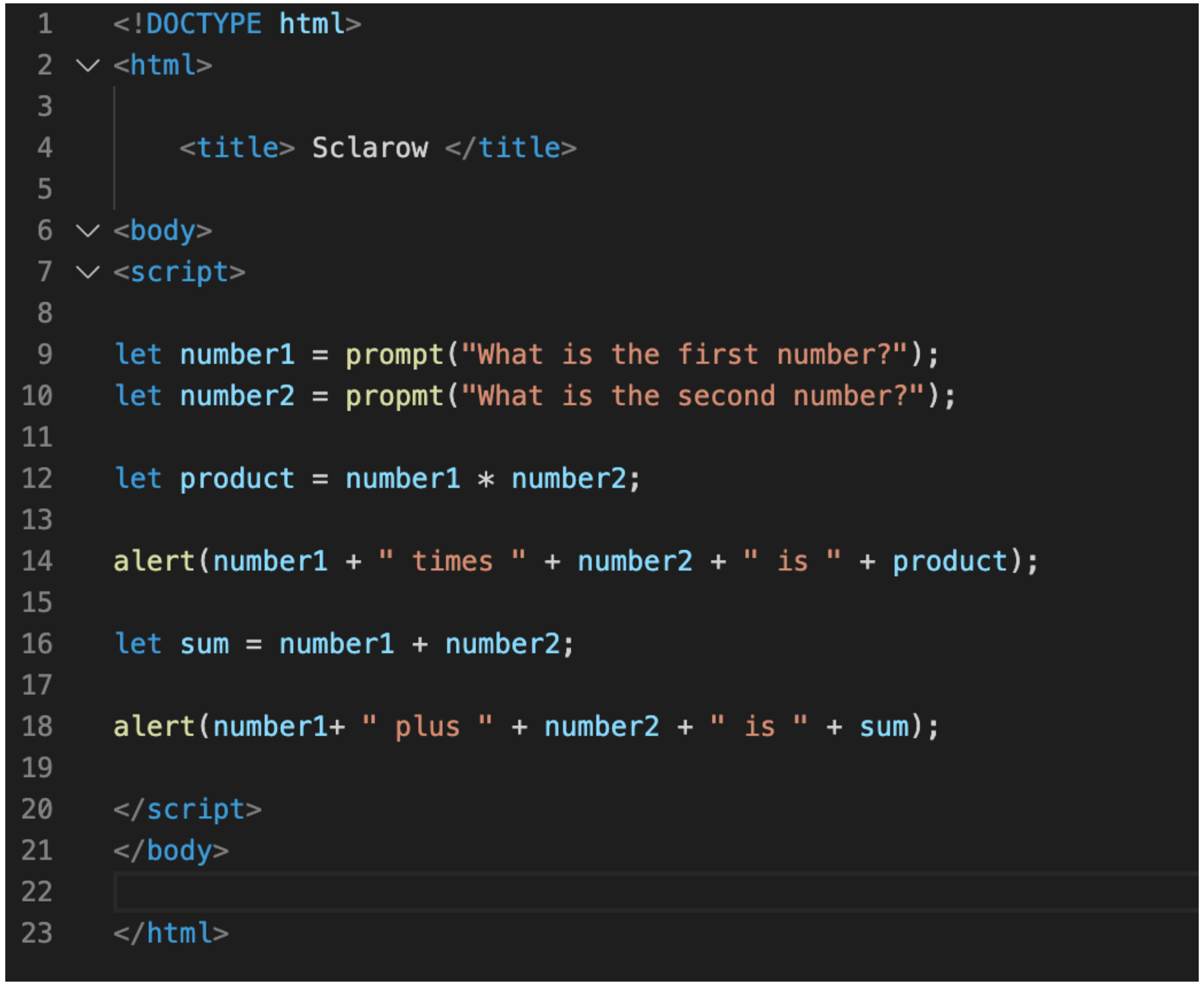
```
firstName = "Grace", lastName = "Hopper";
fullName = lastName + ", " + firstName;
                     // fullName is "Hopper, Grace"
```

How to concatenate string variables with the += operator

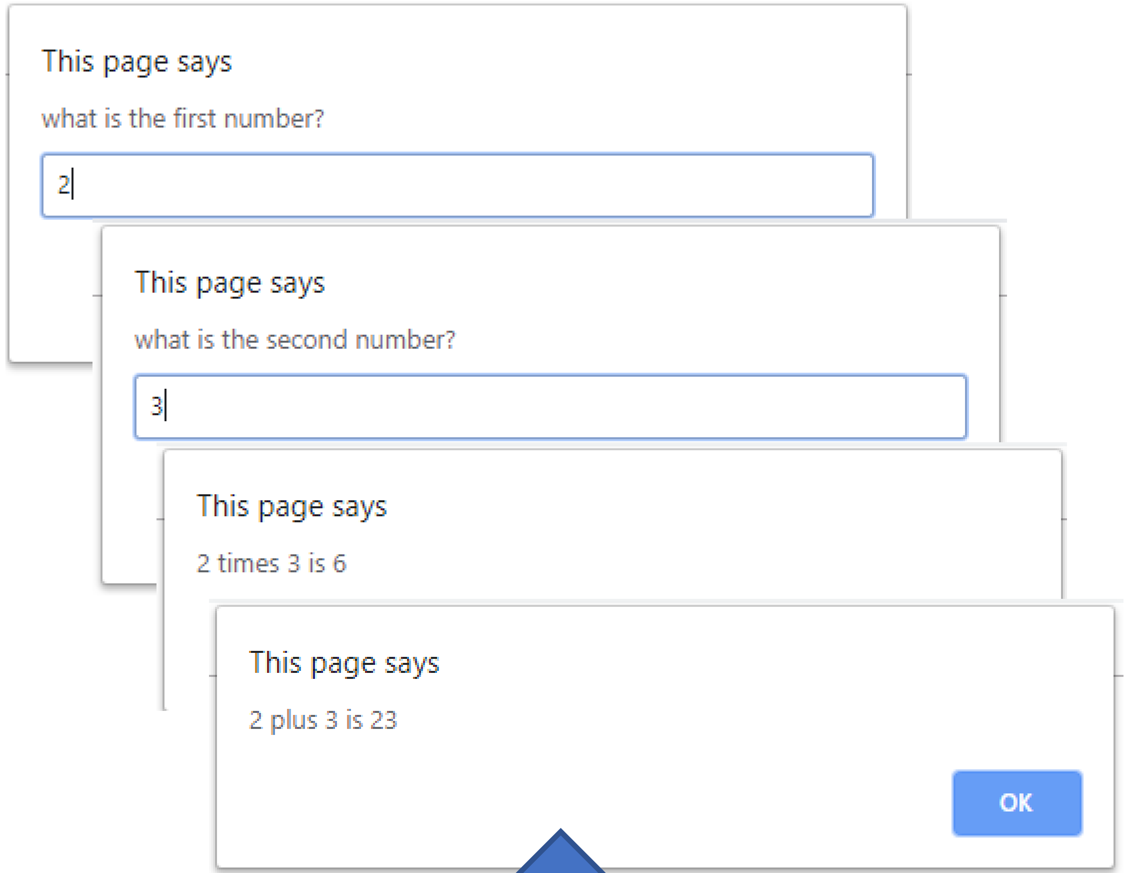
```
var firstName = "Grace", lastName = "Hopper";
var fullName = lastName;   // fullName is "Hopper"
fullName += ", ";          // fullName is "Hopper, "
fullName += firstName;     // fullName is "Hopper, Grace"
```

Prompt(), addition and concatenation

- Prompt always returns a string
- If the string looks like a number, JavaScript will convert the string to a number to do arithmetic
- JavaScript doesn't always do what you expect when using the "+" operator because it is used to perform addition when dealing with numbers and concatenation when dealing with strings
- ***If JavaScript isn't converting strings to numbers as you expect, use parseInt() or parseFloat() to convert them



```html
<!DOCTYPE html>
<html>

    <title> Sclarow </title>

<body>
<script>

let number1 = prompt("What is the first number?");
let number2 = propmt("What is the second number?");

let product = number1 * number2;

alert(number1 + " times " + number2 + " is " + product);

let sum = number1 + number2;

alert(number1+ " plus " + number2 + " is " + sum);

</script>
</body>

</html>
```



This page says

what is the first number?

2

This page says

what is the second number?

3

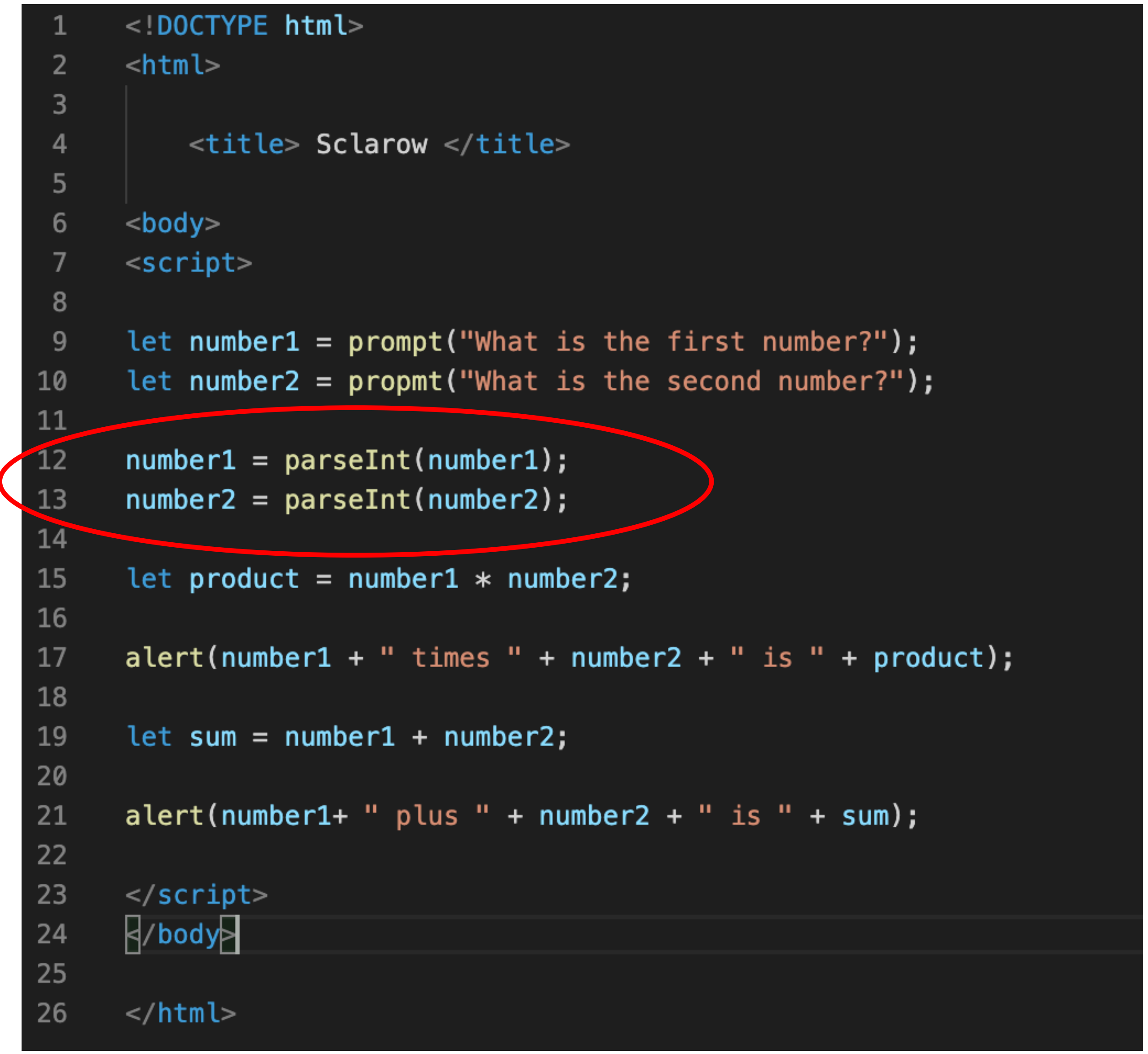
This page says

2 times 3 is 6

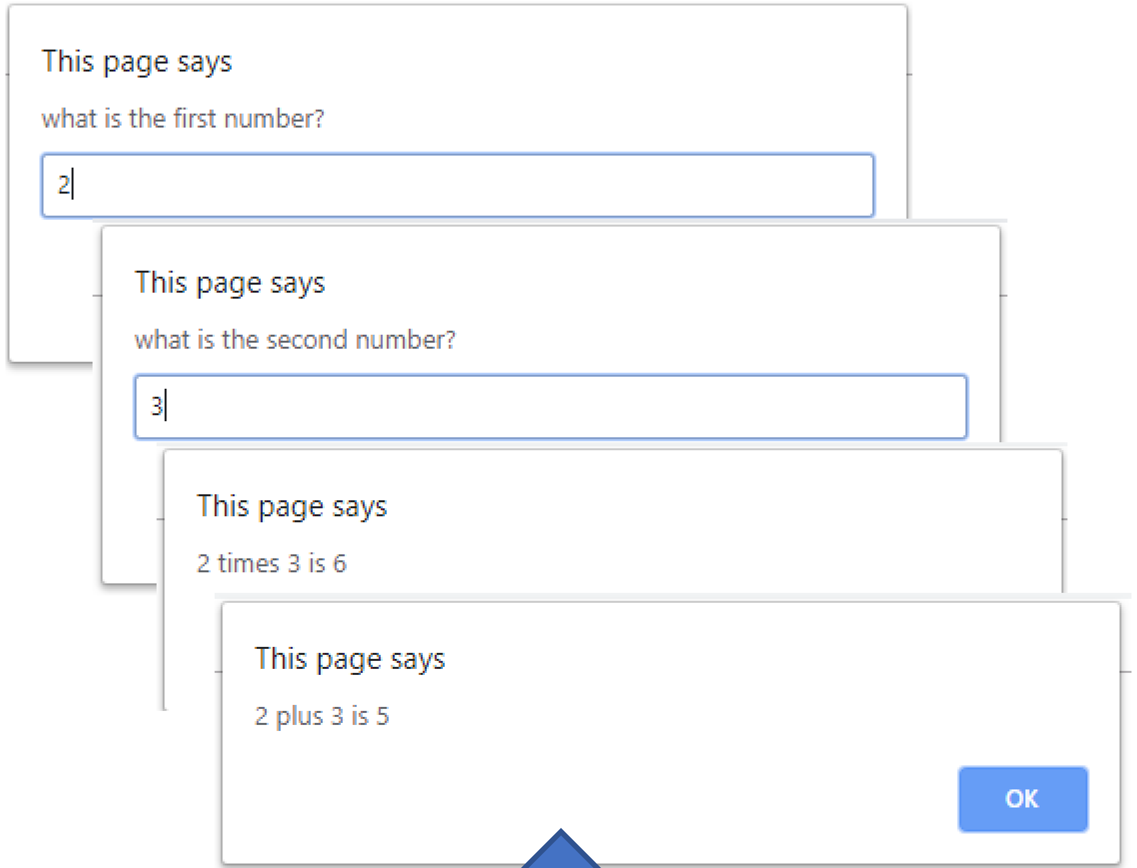
This page says

2 plus 3 is 23

Output if JavaScript doesn't convert the number to a number

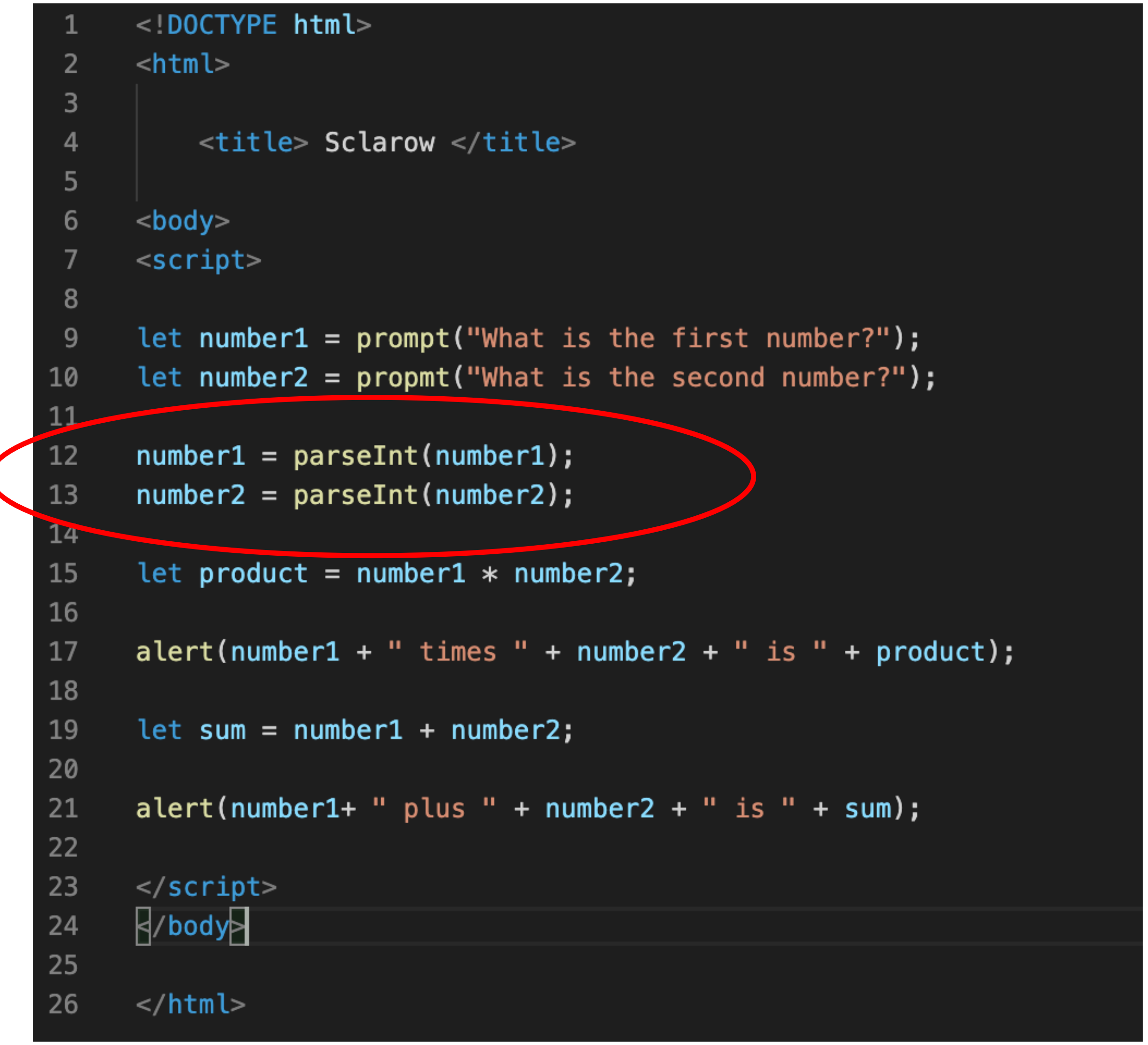


```
<!DOCTYPE html>
<html>

    <title> Sclarow </title>

<body>
<script>

let number1 = prompt("What is the first number?");
let number2 = propmt("What is the second number?");

number1 = parseInt(number1);
number2 = parseInt(number2);

let product = number1 * number2;

alert(number1 + " times " + number2 + " is " + product);

let sum = number1 + number2;

alert(number1+ " plus " + number2 + " is " + sum);

</script>
</body>

</html>
```

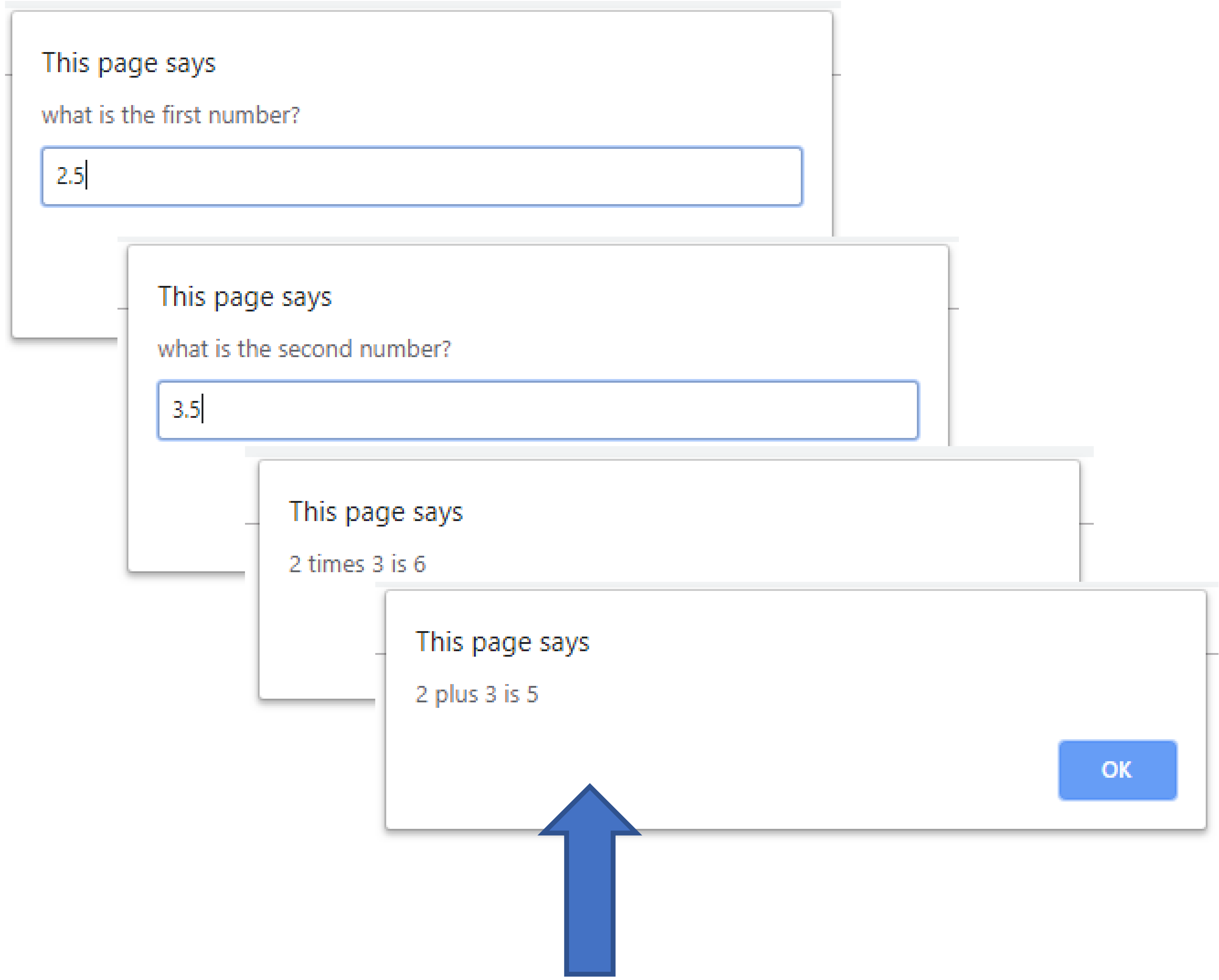


```
This page says
what is the first number?
2

This page says
what is the second number?
3

This page says
2 times 3 is 6

This page says
2 plus 3 is 5
```

Output once we use parseInt to convert the string to a number



```
<!DOCTYPE html>
<html>

    <title> Sclarow </title>

<body>
<script>

let number1 = prompt("What is the first number?");
let number2 = propmt("What is the second number?");

number1 = parseInt(number1);
number2 = parseInt(number2);

let product = number1 * number2;

alert(number1 + " times " + number2 + " is " + product);

let sum = number1 + number2;

alert(number1+ " plus " + number2 + " is " + sum);

</script>
</body>

</html>
```



This page says

what is the first number?

2.5

This page says

what is the second number?

3.5

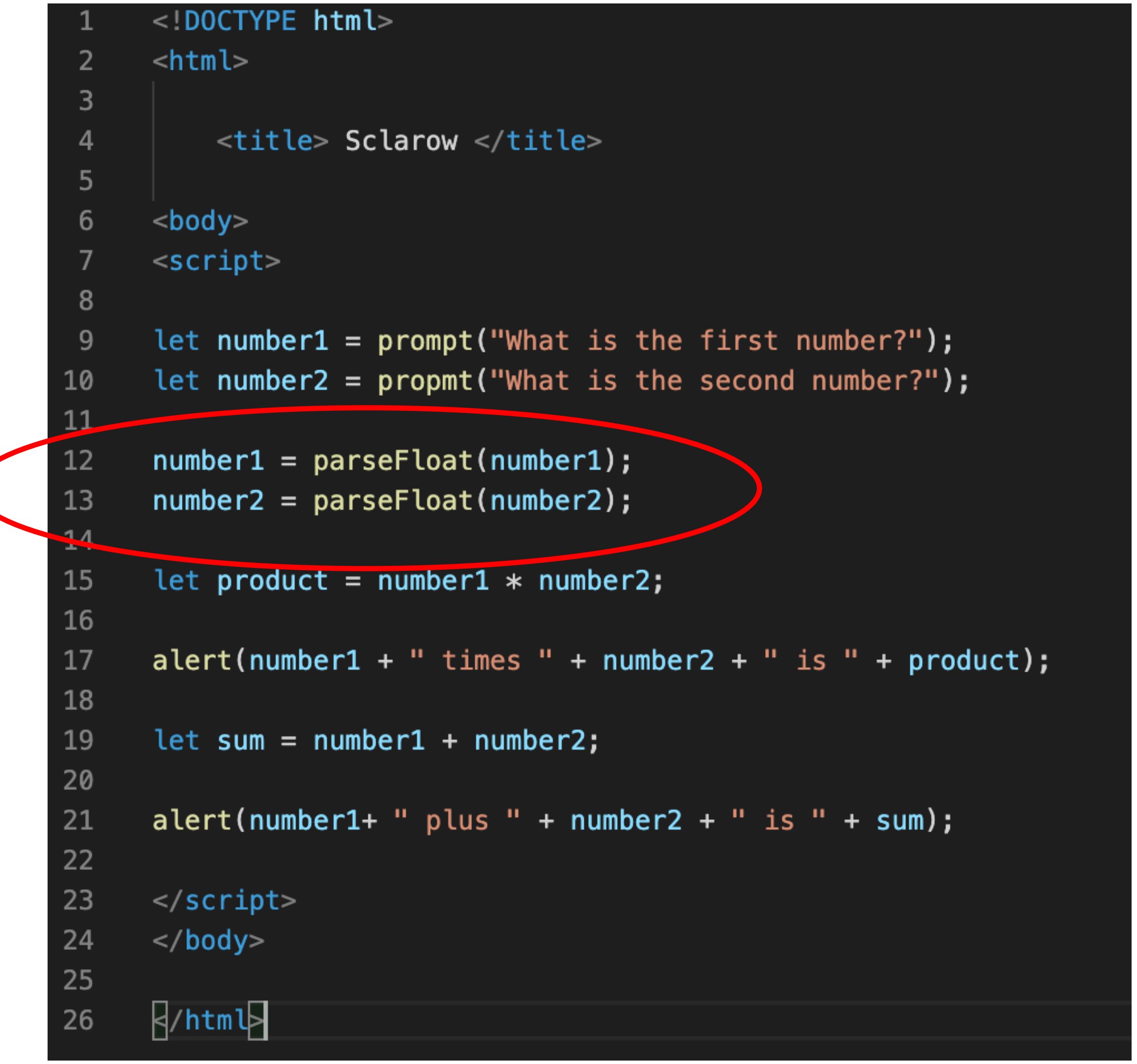
This page says

2 times 3 is 6

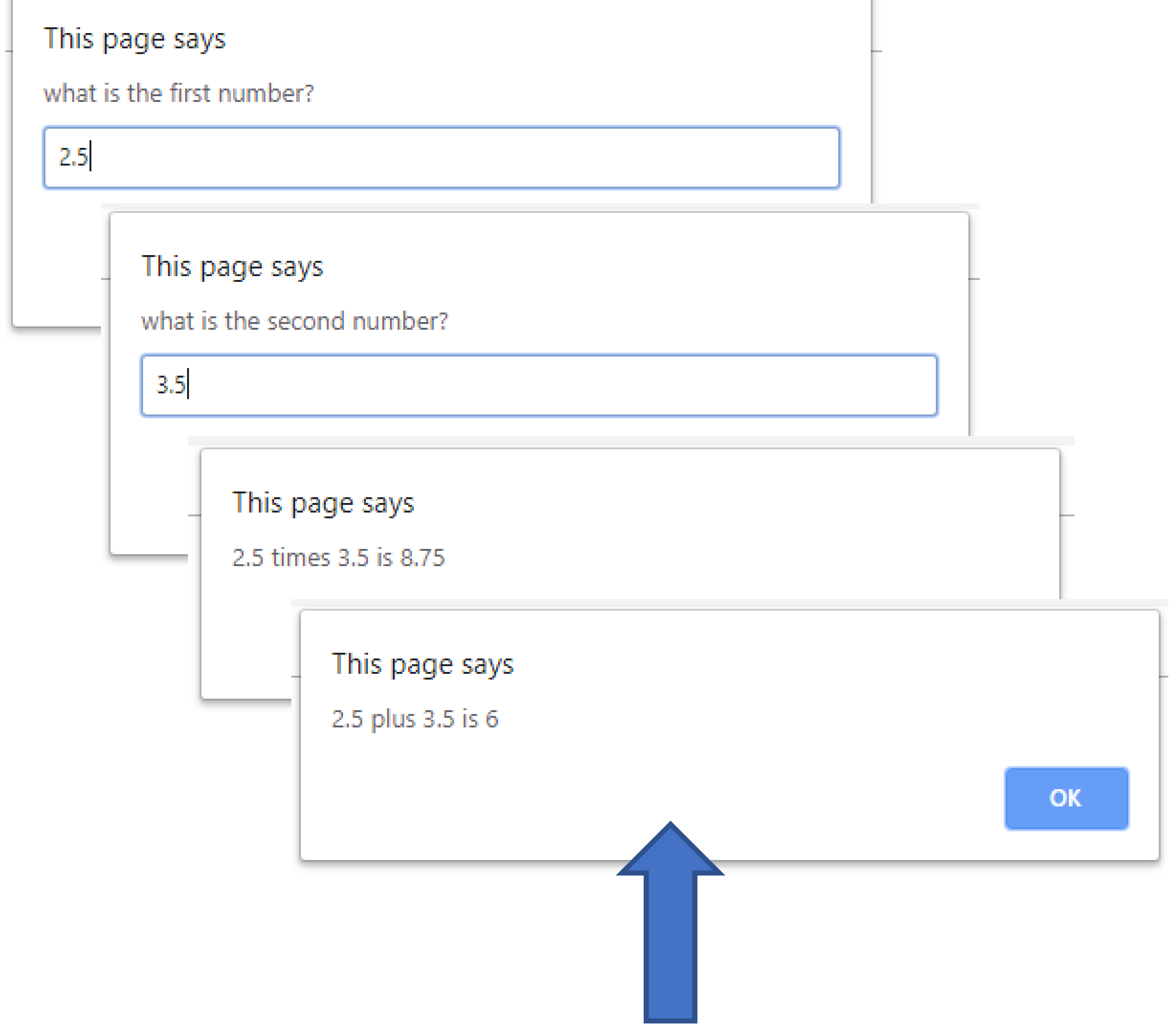
This page says

2 plus 3 is 5

Is parseInt what we want if we're dealing with floating point numbers (a.k.a. numbers with a decimal point)?



```
<!DOCTYPE html>
<html>

    <title> Sclarow </title>

<body>
<script>

let number1 = prompt("What is the first number?");
let number2 = propmt("What is the second number?");

number1 = parseFloat(number1);
number2 = parseFloat(number2);

let product = number1 * number2;

alert(number1 + " times " + number2 + " is " + product);

let sum = number1 + number2;

alert(number1+ " plus " + number2 + " is " + sum);

</script>
</body>

</html>
```



This page says

what is the first number?

2.5

This page says

what is the second number?

3.5

This page says

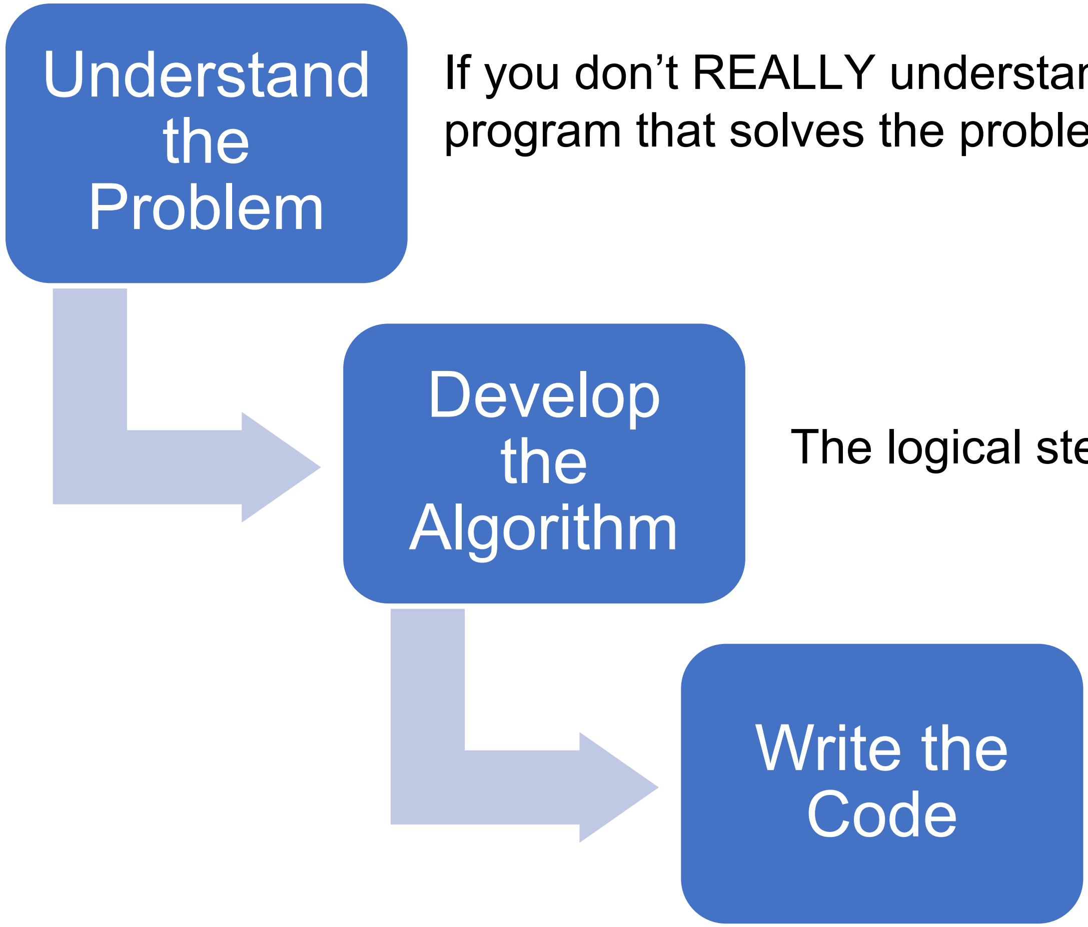
2.5 times 3.5 is 8.75

This page says

2.5 plus 3.5 is 6

Use parseFloat when we're dealing with floating point numbers (a.k.a. numbers with a decimal point)?

How to write a program in 3 easy steps!



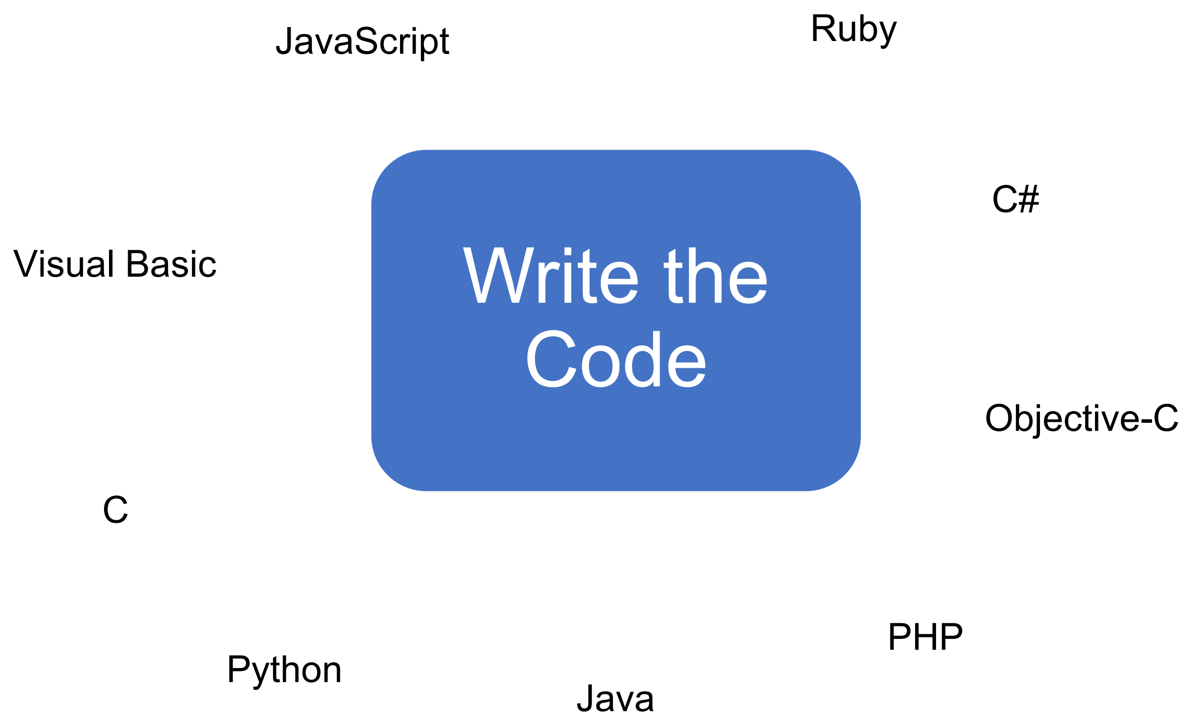
If you don't REALLY understand the problem, you'll never in a million years write a program that solves the problem

The logical steps that solves this particular problem

The automated process

The only part that is language specific…

- Once you have the algorithm (the hard part!), translating the algorithm to a particular programming language is fairly easy. If writing the code seems difficult, your problem is usually a bad algorithm!
- You can use lots of different languages. Some languages do some things better than others but they all do the same basic things.
- In this class we will be using JavaScript, the de-facto standard for applications that run in a browser.



If Swim Lane diagrams model processes and

If a program is just an automated process

Can we use a similar technique to model automated processes?

Pair programming is an agile software development technique in which two programmers work together at one workstation. One, the *driver*, writes code while the other, the *observer* or *navigator*, reviews each line of code as it is typed in. The two programmers switch roles frequently.

-Wikipedia

Time for “Challenges”!

Challenges

- HelloWorld (already done)
- Address
- GuessANumber
- Profits
- LandCalculations
- TotalPurchases

Diamond Peer Teacher
Ariella Izbinsky

Addresses Intro Walkthrough

FOX
MIS

More to Come

Prepare with Readings & Videos before our next class!!!