Program Design Phase

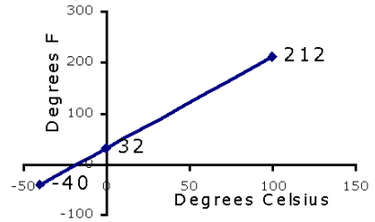
- Write Program Specifications
  - Analysis of requirements
  - Program specifications description
    - Describe what the goals of the program
    - Describe appearance of input and output
- Algorithm Design
  - Mathematical Analysis and Algorithm
  - Flow Chart to describe event sequencing
- Verify algorithm
  - Test with known data
  - Solve manually

 1

1

Algorithm Design - Mathematical

- Mathematical Description
  - Boiling point
    F =212
    C = 100
  - Freezing point
    F = 32
    C = 0



$Y = MX + B$

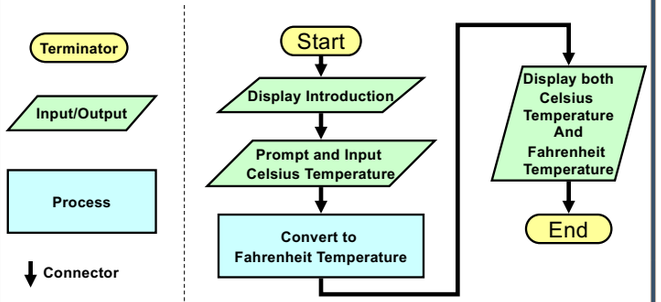
$F = (180 / 100) C + 32$
$= (9/5) C + 32$
$= 1.8 C + 32$

 2

2

Algorithm Design - Sequence

- Flowcharts are an excellent way to plan the sequence of operations for the program to run



 3

3

Verify Algorithm

- Testing with known data
  - Boiling point
    F =212 C = 100
  - Freezing point
    F = 32 C = 0
  - Collect Data
    - Bank thermometer
    - Radio weather report
- Solve manually by hand using calculator

 4

4

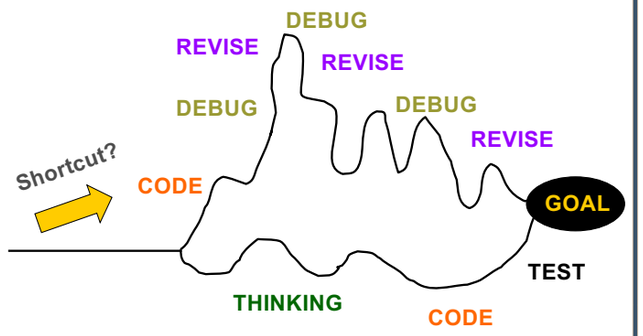
Implementation Phase

- Translate Algorithm into Code
  - Create HTML source code file embedding JavaScript code
  - Run to detect *syntax errors*
- Test Program
  - Test with known data
  - Detects program *logic errors*
  - Often requires several iterations
  - May require re-evaluation of specifications and algorithms

 5

5

Coding First Is No Shortcut?



 6

6

JavaScript Programming Language

- All Web browsers support the JavaScript client-side scripting language and contain the JavaScript Interpreter, which processes JavaScript commands.
- JavaScript code usually appears in the <head> section of the HTML document. The browser interprets the contents of the <head> section first, before the <body> of the HTML document is rendered.
- JavaScript is Case Sensitive and all Keywords must be lower case
- JavaScript is an object based language
- Whitespace is ignored = space, tabs, new lines

 7

7

HTML <script> Element

- <script> element indicates to browser that text that follows is part of a script.
  - Most browser use JavaScript as the default scripting language
  - type attribute specifies type of scripting language and is optional for HTML5

```
<script type="text/javascript">
   script code statements;
</script>
```

 8

8

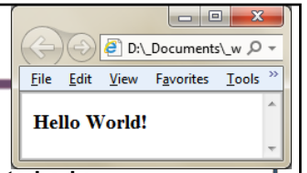
JavaScript Comments and Statements

- Text contained within a JavaScript comment is not executed by the JavaScript interpreter
  - Single-line comments // This is a comment
  - Multi-line comments /* This is a comment */
- Browser that does not support scripts, ignores the <script> element and the script code
- All JavaScript statements end with a semicolon ;
- JavaScript can output HTML code to the browser which then displays the contents.
  document.write ("<h3>Hello World!</h3>");

 9

9

JavaScript Output



- document.write()
- Object is document
- Method is write = sends string to body

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>A First Program in JavaScript</title>
  </head>
  <body>
    <script>
        document.write( "<h3>Hello World!</h3>" );
    </script>
  </body>
</html>
```

 10

10

Strings and Escape Characters

- Character Strings are denoted by enclosing text in either 'single' or "double quotes"
- Escape Characters must use a backslash preceding the specification

Text string escape character specifications:

\n = new line	\\ = backslash
\" = double quote	\' = single quote
\t = tab	\r = carriage return

 11

11

String Concatenation and Escape Characters

- String Concatenation Operator +
  - Connects two strings together
  - Special Character \"


```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Using String Concatenation</title>
  </head>
  <body>
    <script>
       document.write("<h2>");
       document.write("Welcome to string" +
       " \"concatenation\"!</h2>");
    </script>
  </body>
</html>
```

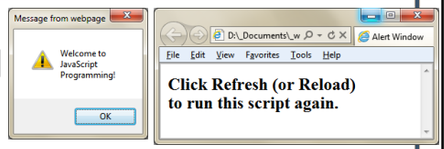
 12

12

JavaScript Output: Alert Window

- window.alert()

Object Method




```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Alert Window</title>
  </head>
  <body>
    <h2>Click Refresh (or Reload)<br>
    to run this script again.</h2>
    <script>
      window.alert("Welcome to\nJavaScript\nProgramming!");
    </script>
  </body>
</html>
```

 13

13

JavaScript Variables

- A Variable is a container of data
- Variables declared with var statement
  - var nI; // Single variable declaration
  - var sEntry1, sEntry2, nJ, nM; // Multiple variables
  - var nI=0, nJ=0; // Variables can be initialized to value
- Declaration statements end with semicolon (;)
- Multiple variable declaration comma separated
- Variable name can be any valid identifier.
  - An identifier is a name for a variable of function
  - Consisting of letters, digits, "_" and "$"
  - Can NOT begin with a digit
  - Can NOT have spaces or symbols other then _ and $
  - Can NOT be a JavaScript keyword

 14

14

JavaScript Keywords

- JavaScript has only 22 keywords that can NOT be used for an identifier name.

break	case	continue	delete	do
else	false	for	function	if
in	new	null	return	switch
this	true	typeof	var	void
while	with			

- Twelve other keywords also can not be used for identifiers

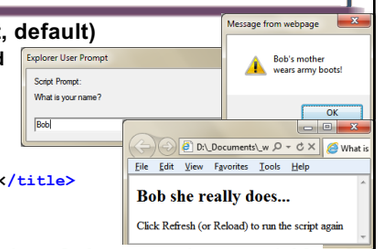
catch	class	const	debugger	default
enum	export	extends	finally	import
super				

 15

15

JavaScript Prompt for Input Data

- window.prompt(prompt, default)
  - Return the string entered to assigned variable




```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>What is your name</title>
  </head>
  <body>
    <script>
      var sFirstName; // String of characters input variable
      sFirstName = window.prompt( "What is your name?", "" );
      window.alert(sFirstName + "\'s mother\nwears army boots!");
      document.write("<h2>" + sFirstName
        + " she really does...</h2>");
    </script>
    <p>Click Refresh (or Reload) to run the script again</p>
  </body>
<html>
```

 16

16

JavaScript Data Types and Values

- JavaScript is "loosely" typed language
- Simple Data Types
  - String of text var sFirstName, sEntry;
    - Symbolized using "abc123" or 'abc123'
    - Special Characters may be used \n \t \b \" \'
  - Numbers var nI = 0, fArea, fTotal = 0;
    - 8 byte (64 bit) floating point format $\pm 1.8 \times 10^{\pm 308}$
    - *int* parseInt(*string*)
      - Converts string to integer (whole number)
      - Drops all fractional part to right of decimal point
    - *float* parseFloat(*string*)
      - Converts string to floating point (real number)
      - Keeps fractional part to right of decimal point

 17

17

JavaScript Arithmetic Operators

- Used to perform arithmetic operations on numbers and data contained in variables, with the result usually assigned to variable
- Order of precedence determines which order the operations will be performed
- Note that the assignment operator = is defined last and precedence is last
- For readability insert parenthesis if order of operation not apparent in code

 18

18

Arithmetic Operators Precedence

(Highest to Lowest)

( )	Defines order of operation
-	Negative (unary)
* / %	Multiply, Division, Remainder
+ -	Addition (concatenation), Subtraction
=	Assignment

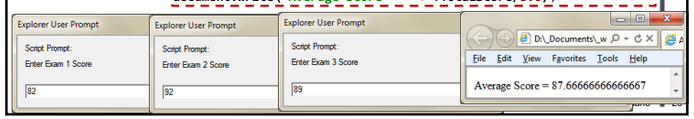
 19

19

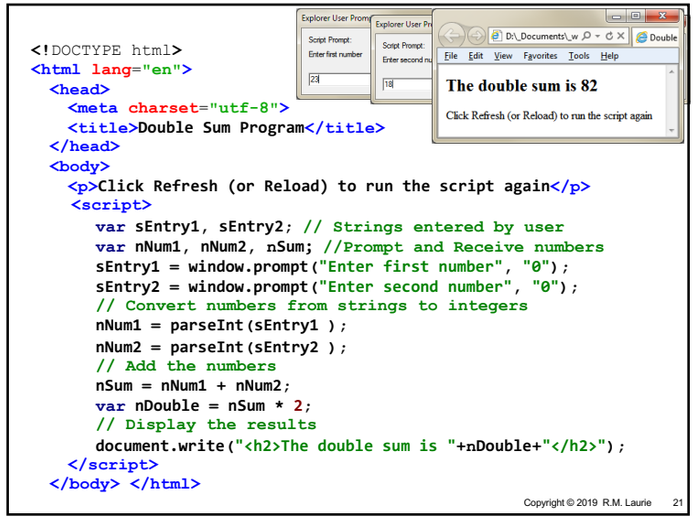
3 Exams Average Example

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Average Test Score</title>
  </head>
  <body>
    <script src="script.js"></script>
  </body>
</html>
```

```
// script.js file source code
var fAvgScore, fScore, fTotalScore = 0;
var sEntry = window.prompt( "Enter Exam 1 Score", "0" );
fScore = parseFloat(sEntry);
fTotalScore = fTotalScore + fScore;
sEntry = window.prompt("Enter Exam 2 Score", "0" );
fScore = parseFloat(sEntry);
fTotalScore = fTotalScore + fScore;
sEntry = window.prompt("Enter Exam 3 Score", "0" );
fScore = parseFloat(sEntry );
fTotalScore = fTotalScore + fScore;
document.write("Average Score = " + fTotalScore/3.0);
```



20



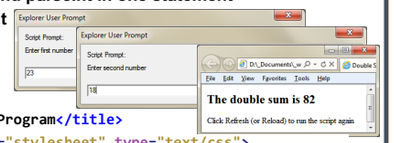

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Double Sum Program</title>
  </head>
  <body>
    <p>Click Refresh (or Reload) to run the script again</p>
    <script>
       var sEntry1, sEntry2; // Strings entered by user
       var nNum1, nNum2, nSum; //Prompt and Receive numbers
       sEntry1 = window.prompt("Enter first number", "0");
       sEntry2 = window.prompt("Enter second number", "0");
       // Convert numbers from strings to integers
       nNum1 = parseInt(sEntry1 );
       nNum2 = parseInt(sEntry2 );
       // Add the numbers
       nSum = nNum1 + nNum2;
       var nDouble = nSum * 2;
       // Display the results
       document.write("<h2>The double sum is "+nDouble+"</h2>");
    </script>
  </body> </html>
```

 21

21

A Shorter Double Sum Program

- Previous program with less code
  - How many variables?
  - Combine input prompt and parseInt in one statement
  - Do calculations in output




```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Shorter Double Sum Program</title>
    <link href="style.css" rel="stylesheet" type="text/css">
  </head>
  <body>
    <p>Click Refresh (or Reload) to run the script again</p>
    <script src="script.js"> </script>
    </body>
  </html>
```

script.js file

```
var nNum1, nNum2;      // Converted number entries
nNum1 = parseInt(window.prompt("Enter first number","0"));
nNum2 = parseInt(window.prompt("Enter second number","0"));
document.write("<h2>The double sum is " + (nNum1 + nNum2) * 2
  + "</h2>");
```

 22

22