

## Flow of Control

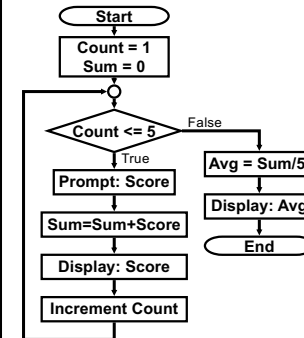
- ❖ **Definition:** The sequential execution of statements in a program
  - ◆ Sequential Control Structure (Top-Bottom)
  - ◆ Selection Control Structure (Decisions)
  - ◆ Repetition Control Structure (Looping)
    - ◆ Loop back and repeats code execution
    - ◆ Relational and Logical Operators
    - ◆ Tests an Assertion (T/F) to loop again or exit
    - ◆ Counter controlled or Sentinel controlled loops
    - ◆ Keywords: **while** **do while** **for**
    - ◆ Computers Never Get Bored
      - Best for iterative well structured processing
      - Not well suited for creative problem solving

Copyright © 2019 R.M. Laurie 1

1

## Repetition (Loop) Structure

- ❖ Repeat a sequence of instructions in a loop
- ❖ The simplest loop structure is the while( )
- ❖ Beware of infinite loops, exit must occur!



```

var fScore, nCount = 1, fSum = 0;
while(nCount <= 5)
{
    fScore = parseFloat(window.prompt(
        "Enter Score " + nCount, "0"));
    fSum = fSum + fScore;
    document.write("<p>Score "+nCount
        +" = " + fScore+"</p>");
    nCount = nCount + 1;
}
document.write("<h3>Average = "
    +(fSum/5) + "</h3>");
    
```

Copyright © 2019 R.M. Laurie 2

2

## while statement loop control

- ❖ Contents of loop executed repeatedly while(assertion) is **true**
- ❖ Loop terminated when while(assertion) is **false**
- ❖ Counter-Controlled Repetition Structure
  - ◆ Initialize a counter to count loops
  - ◆ Increment or decrement counter
  - ◆ while(assertion) checks for total loops reached
- ❖ Sentinel-Controlled Repetition Structure
  - ◆ while(assertion) checks for a sentinel termination value

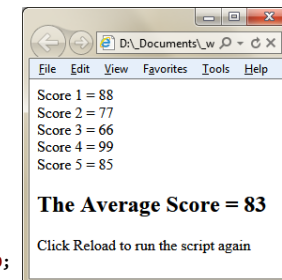
Copyright © 2019 R.M. Laurie 3

3

## Counter-Controlled Pre-test Repetition Structure

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Counter Controlled Loop</title>
</head>
<body>
  <script>
    var nScore=0, nScoreTotal=0, nCount=0;
    while(nCount < 5)
    {
      nScore=parseInt(window.prompt("Enter Score", ""));
      nScoreTotal = nScoreTotal + nScore;
      nCount = nCount + 1;
      document.write("Score " + nCount + " = " + nScore + "<br>");
    }
    document.write("<h2>The Average Score = "+ nScoreTotal/5 + "</h2>");
  </script>
  <p>Click Reload to run the script again</p>
</body>
</html>
    
```



1. Define counter
2. Initialize counter
3. Increment counter

Copyright © 2019 R.M. Laurie 4

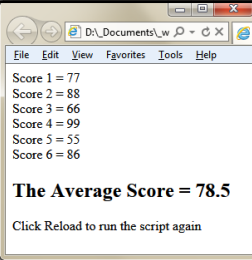
4

### Sentinel-Controlled Pre-test Repetition Structure

```

<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Sentinal Controlled Loop</title>
</head>
<body>
  <script>
    var fScore, fScoreTotal=0;
    var nCount=0;
    fScore = parseFloat(window.prompt("Enter Score (-1 to end)", ""));
    while(fScore >= 0)
    {
      fScoreTotal = fScoreTotal + fScore;
      nCount = nCount + 1;
      document.writeln("Score " + nCount + " = " + fScore + "<br>");
      fScore = parseFloat(window.prompt("Enter Score (-1 to end)", ""));
    }
    document.writeln("<h2>The Average Score = "
      + fScoreTotal/nCount + "</h2>");
  </script>
  <p>Click Reload to run the script again</p>
</body>
</html>

```



The Average Score = 78.5

Click Reload to run the script again

What is sentinel?  
What are advantages?

Copyright © 2019 R.M. Laurie 5

5

### More JavaScript Operators

- ++ Increment (Unary)**  
`Number++; // Number = Number + 1;`
- Decrement (Unary)**  
`Number--; // Number = Number - 1;`
- Object Property (Encapsulated in object)**  
Select property or method of an object.  
`document.write("<h3>Average = " + (Sum / 5) + "</h3>");`
- Combined Assignment**
  - += Addition Assignment Operator**
  - = Subtraction Assignment Operator**
  - \*= Multiplication Assignment Operator**
  - /= Division Assignment Operator**
  - %= Remainder Assignment Operator**

Copyright © 2019 R.M. Laurie 6

6

### Operator Examples

```

Num++; // Num=Num+1 (Post-increment)
++Num; // Num=Num+1 (Pre-increment)
Num--; // Num=Num-1 (Post-decrement)
--Num; // Num=Num-1 (Pre-decrement)

A += 2; // A=A+2
B -= 1; // B=B-1
C *= 4; // C=C*4
D /= 2; // D=D/2
E %= 5; // E=E%5

```

Copyright © 2019 R.M. Laurie 7

7

### Operators Precedence (Highest to Lowest)

.	Property access of an object
( )	Defines order of operation
- ++ --	Minus, Increment, Decrement
!	Logical NOT Operator
* / %	Multiply, Division, Remainder
+ -	Addition, Subtraction
< <= > >=	} Relational Operators
== !=	
&&	Logical AND Operator
	Logical OR Operator
= += -= *= /= %=	Compound Assignment

Copyright © 2019 R.M. Laurie 8

8

## Logical Operator Examples

```

if(A==B && A==C)
while(!Valid)
if(A = 0) // Error use ==
else if(!(A || B))
while(!A && !B)
A <= B || C == D
A = B == 0;
if(Question == "C" || Question == "c")
while(SSN > 999999999 || SSN < 0)
if(Tax == 0 || Tax == 15 || Tax == 28)
    
```

Copyright © 2019 R.M. Laurie 9

9

## Counter-Controlled Loop with ++ and +=

```

<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>++ += Counter Controlled Program</title>
</head>
<body>
<script src="CounterControlLoop.js"></script>
<p>Click Reload to run the script again</p>
</body>
</html>
    
```

CounterControlLoop.js external linked file

```

var nScore = 0, nScoreTotal = 0, nCount = 0;
while(nCount < 5)
{
nScore = parseInt(window.prompt("Enter Score", ""));
nScoreTotal += nScore; // ScoreTotal = ScoreTotal + Score;
nCount++; // was Count = Count + 1;
document.write("Score " + nCount + " = " + nScore + "<br>");
}
document.write("<h2>The Average Score = " + nScoreTotal/5 + "</h2>")
    
```

Copyright © 2019 R.M. Laurie 10

10

## Input Data Validation Application 1

**Pre-test while( ) Loop**  
**Restricts user to enter only valid input data**  
**Sentinel Controlled**

```

<!DOCTYPE html>
<html lang="en">
<head>
<title>Filtered Input</title>
</head>
<body>
<script src="FilterEntry.js"></script>
</body>
</html>
    
```

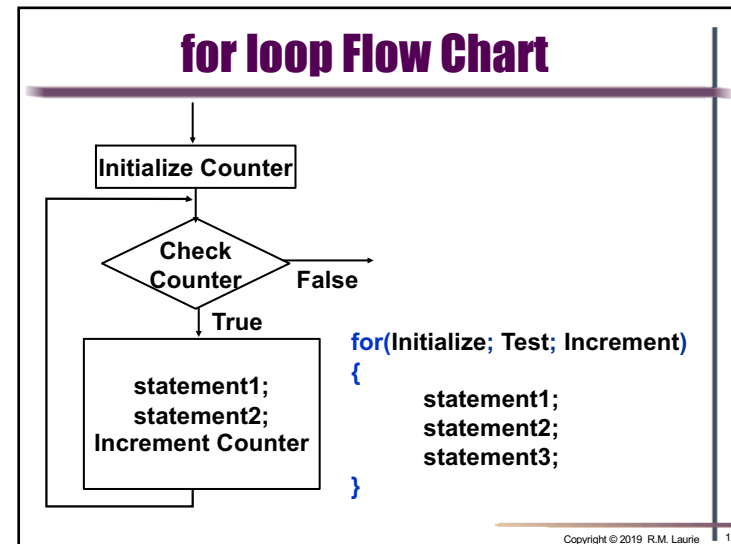
FilterEntry.js

```

var sEntry, bValid=false;
while(bValid == false) {
sEntry = window.prompt( "Do you like Programming? (y or n)","" );
if(sEntry == "y") {
document.writeln("<h2>I\'m glad you like programming!</h2>");
bValid = true;
}
else if(sEntry == "n") {
document.writeln("<h2>You will like it if you study.</h2>");
bValid = true;
}
else
window.alert("You must enter either y or n !");
} // <-- Note that this is the end of the while loop
    
```

Copyright © 2019 R.M. Laurie 11

11

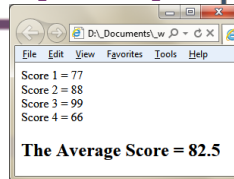


12

## Slide Set 5: Javascript-Loop

### For ( ) Counter Controlled Loop Example

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Average Calculation 2</title>
</head>
<body>
<script>
var nScore, nScoreTotal = 0, nCount, nQty;
nQty = parseInt(window.prompt("How Many Scores?", ""));
for(nCount = 1; nCount <= nQty; nCount++)
{
nScore = parseInt(window.prompt("Enter Score", ""));
nScoreTotal = nScoreTotal + nScore;
document.write("Score " + nCount + " = "
+ nScore + "<br/>");
}
document.write("<h2>The Average Score = "
+ (nScoreTotal / nQty) + "</h2>");
</script>
</body>
</html>
```

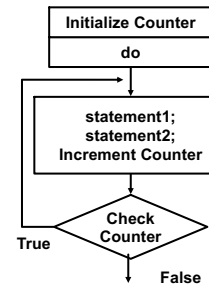


Copyright © 2019 R.M. Laurie 13

13

### do - while Post-test Structure

- ❖ A loop structure that guarantees the loop body is executed once.
- ❖ Condition is tested at bottom of loop
- ❖ Don't forget the semicolon for **while(...)**;



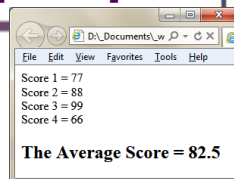
```
Initialize Counter;
do
{
statement1;
statement2;
Increment Counter;
}while(Check Counter);
```

Copyright © 2019 R.M. Laurie 14

14

### Sentinel Controlled Loop Example

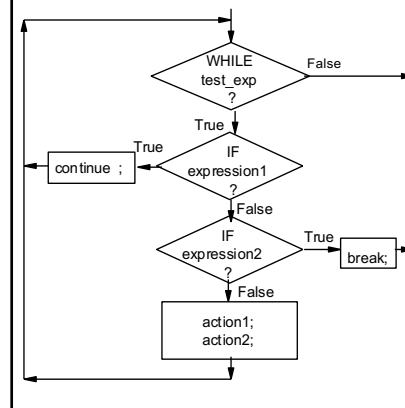
```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Average Calculation 2</title>
</head>
<body>
<script>
var nScore, nCount=0, nTotal = 0;
do
{
nScore = parseInt(window.prompt("Enter Score or [Q]=quit", "Q"));
if(isNaN(nScore)); // Score is Not a Number
else if(nScore < 0)
window.alert("Score cannot be negative");
else
{
nTotal += nScore;
nCount++;
document.write("<p>Score " + nCount+" = " + nScore + "</p>");
}
}while(!isNaN(nScore));
document.write("<h2>Average Score = " + nTotal/nCount + "</h2>");
</script>
</body>
</html>
```



Copyright © 2019 R.M. Laurie 15

15

### break; continue; *commands*



```
while(test_exp)
{
if(expression1)
continue;
if(expression2)
break;
action1;
action2;
}
```

Copyright © 2019 R.M. Laurie 16

16

### Filtered Input Application 1

```
<script>
var Entry;
while(true)
{
  Entry = window.prompt( "Do you like Programming? (y or n)", "" );
  if(Entry == "y" || Entry == "Y")
  {
    document.writeln("<h2>I'm glad you like programming!</h2>");
    break;
  }
  else if(Entry == "n" || Entry == "N")
  {
    document.writeln("<h2>You will like it if you study.</h2>");
    break;
  }
  else
    window.alert("You must enter either y or n !");
}
</script>
```

### Filtered Input Application 2

```
<script>
var Entry;
do
{
  Entry = window.prompt( "Do you like Programming?", "y or n" );
}while(!(Entry=="y" || Entry=="Y" || Entry=="n" || Entry=="N"));
if( Entry == "y" || Entry == "Y" )
  document.writeln("<h2>I'm glad you like programming!</h2>");
else
  document.writeln("<h2>You will like it if you study.</h2>");
</script>
```

## HTML: Tables

### ❖ Tables display information in tabular form

- ◆ <table> </table> Table start and end
- ◆ <tr> </tr> Table row start and end
- ◆ <th> </th> Table header element
- ◆ <td> </td> Table data element

Item	Price
Soy Milk	\$4.39
Frosted Flakes	\$3.89

```
<table border="1" summary="Prices">
<tr>
<th>Item</th><th>Price</th>
</tr>
<tr>
<td>Soy Milk</td><td>$4.39</td>
</tr>
<tr>
<td>Frosted Flakes</td><td>$3.89</td>
</tr>
</table>
```

## Nested for Example

```
<script>
document.write("<h3>Subtraction Table (X-Y)</h3>"
+ "<table summary='Subtraction' border='1'"
+ "<tr><th>Y\X</th>"
for(var i = 0; i <= 3; i++)
  document.write( "<th>" + i + "</th>");
document.write("</tr>");
for(i = 0; i <= 2; i++)
{
  document.write("<tr><th>" + i + "</th>");
  for(var j = 0; j <= 3; j++)
  {
    document.write( "<td>" + (j - i) + "</td>");
  }
  document.write("</tr>");
}
document.write("</table>");
</script>
```

Subtraction Table (X-Y)

Y\X	0	1	2	3
0	0	1	2	3
1	-1	0	1	2
2	-2	-1	0	1