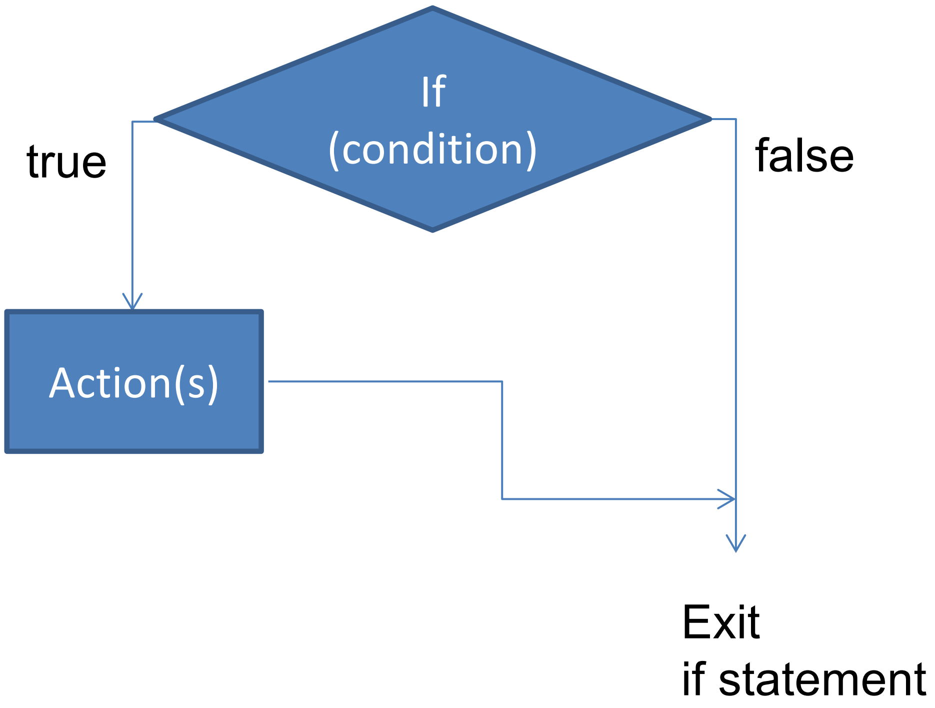
If Statements
(One Conditional Expression)

- Examples of Conditions:
  - X==10
  - X > Y



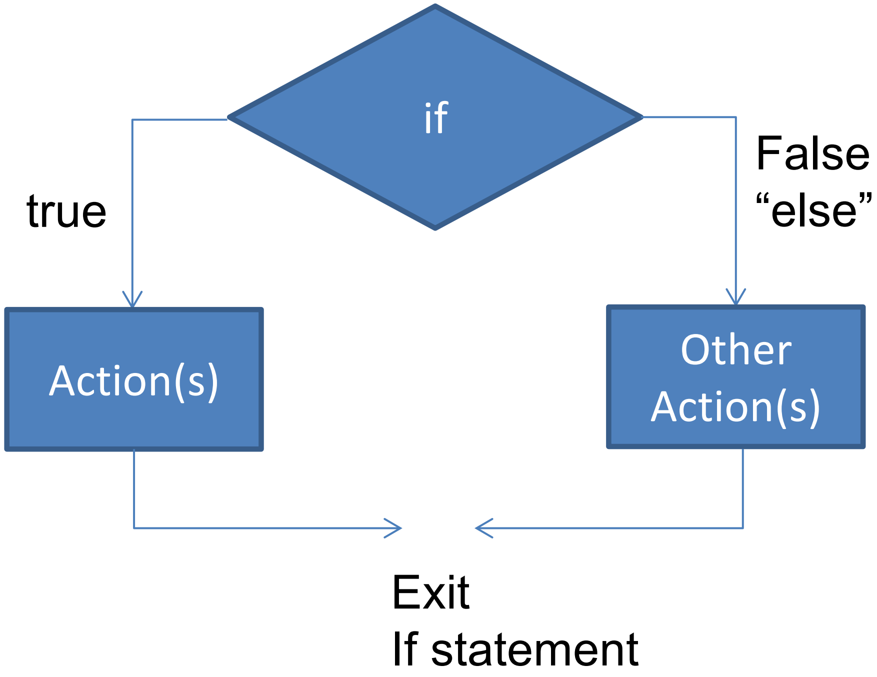
Adapted from K. Bursic

Example – One Condition

```
<script type="text/javascript">
    var a = 1, b = 2
    var sum1, sum2;
    var c = prompt("What value would you like to enter for c?");
    c = Number(c);
    sum1 = a + b;
    document.write("sum1 is equal to ", sum1, "<br \>");
    document.write("c is equal to ", c, "<br \>");
    if (sum1 == c)
    {
        document.write("sum1 is equal to c!", "<br \>");
        sum2 = a + b + c;
        document.write("sum2 is equal to ", sum2, "<br \>");
    }
    document.write("<p\>We are out of the if statement");
</script>
```

If Statements (Two Conditions)

- In the last example, if the condition wasn't true we didn't do anything – but there could be actions for a false situation as well ("else")

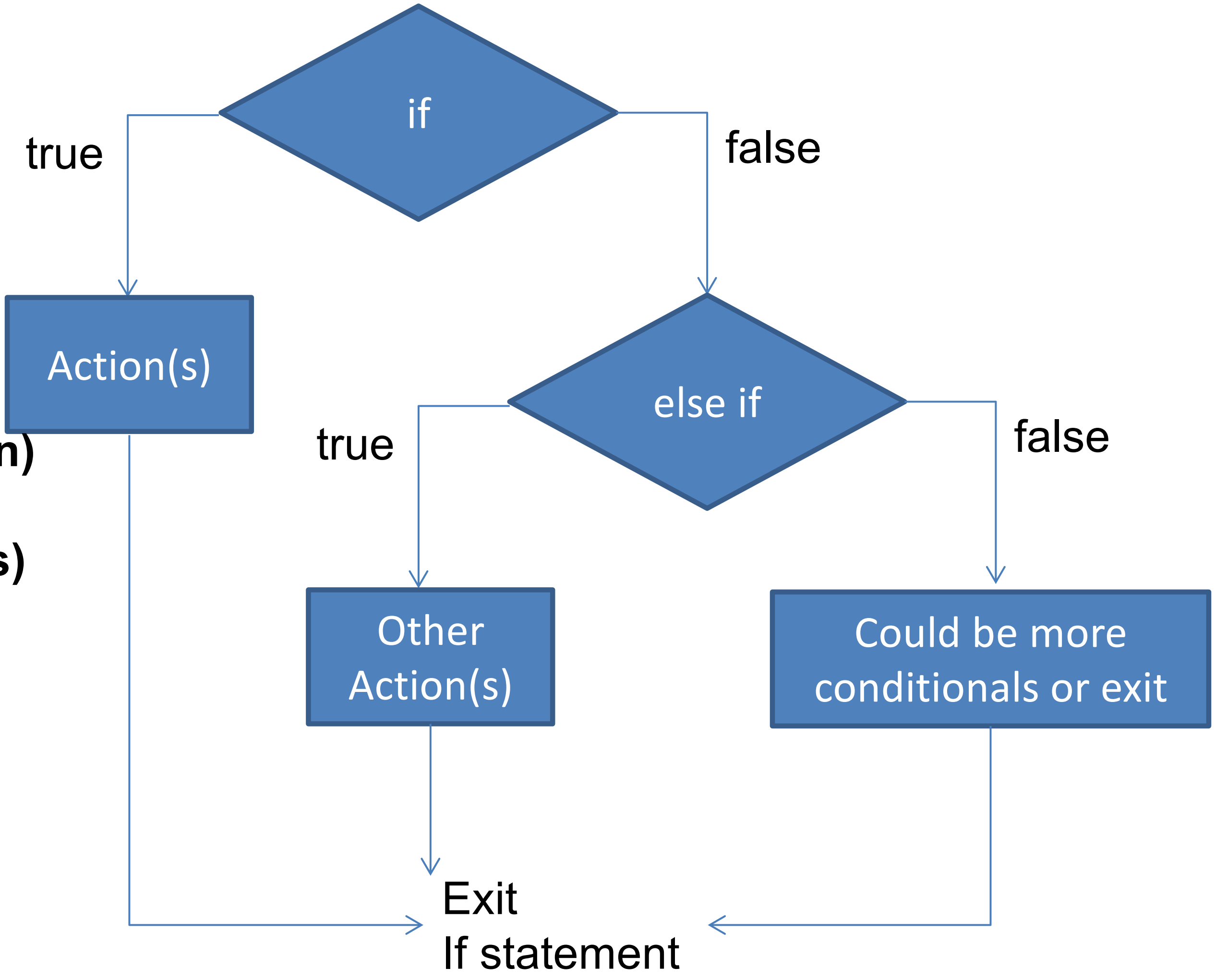


Example – Two conditions

```
<script type="text/javascript">
  var a = 1, b = 2
  var sum1, sum2;
  var c = prompt("What value would you like to enter for c?");
  c = Number(c);
  sum1 = a + b;
  document.write("sum1 is equal to ", sum1, "<br \>");
  document.write("c is equal to ", c, "<br \>");
  if (sum1 == c) {
    document.write("sum1 is equal to c!", "<br \>");
    sum2 = a + b + c;
    document.write("sum2 is equal to ", sum2, "<br \>");
  }
  else {
    document.write("sum1 is <u>not</u> equal to c!", "<br \>");
    sum2 =a*b*c;
    document.write("sum2 is equal to ", sum2, "<br \>");
  }
  document.write("<p\>We are out of the if statement");
</script>
```

Multiple Conditionals

```
if (condition)
  {
   Action(s)
   }
else if (condition)
   {
   Other Action(s)
   }
….
```



Example - Multiple Conditionals

```
<script type="text/javascript">
    var a = 1, b = 2
    var sum1, sum2;
    var c = prompt("What value would you like to enter for c?");
    c = Number(c);
    sum1 = a + b;
    document.write("sum1 is equal to ", sum1, "<br \>");
    document.write("c is equal to ", c, "<br \>");
    if (sum1 == c) {
        document.write("sum1 is <u>equal</u> to c!", "<br \>");
        sum2 = a + b + c;
        document.write("sum2 is equal to ", sum2, "<br \>");
    }
    else if(sum1 < c) {
        document.write("sum1 is <u>less than </u>c!", "<br \>");
        sum2 = c;
        document.write("sum2 is equal to ", sum2, "<br \>");
    }
    else {
        document.write("sum1 is <u>greater than </u>c!", "<br \>");
        sum2=1000;
        document.write("sum2 is equal to ", sum2, "<br \>");
    }
    document.write("<p\>We are out of the if statement");
</script>
```

Multiple Conditionals: Another Example

```
if (condition)
  {
   Action(s)
   }
else if (condition)
   {
   Other Action(s)
   }
….
```

```
<script type="text/javascript">
    var a = 1, b = 2, c = 3;
    var sum1, sum2;

    sum1 = a + b;
    document.write("sum1 is equal to ", sum1, "<br \>");
    document.write("c is equal to ", c, "<br \>");

    if (sum1 == c) {
        document.write("sum1 is equal to c!", "<br \>");
        sum2 = a + b + c;
        document.write("sum2 is equal to ", sum2, "<br \>");
    }
    else if (b > a) {
        sum2 = a * b * c;
        document.write("sum2 is equal to ", sum2, "<br \>");
    }
    else {
        document.write("The other conditions were not true "<br \>");
    }
</script>
```

Things to watch for with IF structures

- Each expression or condition must be enclosed in ( )
- No ; at the end of the condition statement
- Can add && or ||
  - Example: if (b > a && sum1==c)
- Actions for each condition must be enclosed in { }
- Space between “else” and “if”
- If structures can be nested – See next example

Example - Nested If statements

```
<script type="text/javascript">
    var a = 1, b = 2
    var sum1, sum2;
    var c = prompt("What value would you like to enter for c?");
    c = Number(c);
    sum1 = a + b;
    document.write("sum1 is equal to ", sum1, "<br \>");
    document.write("c is equal to ", c, "<br \>");
    if (sum1 == c) {
        if (sum1 != 30) {
            document.write("sum1 is equal to c and not equal to 30");
        }
        else {
            document.write("sum1 is equal to c and equal to 30")
        }
    }
    else {
        document.write("sum1 is <u>not equal</u> to c!", "<br \>");
    }
    document.write("<p\>We are out of the if statement");
</script>
```

Practice Conditionals

- Write a script for checking grade assignments:
  - Define a variable to be student_grade=85; (which can be changed to check different scenarios)
  - Using conditional statements determine the appropriate letter grade that would be assigned
    - Use document.write statements to show the result to the screen:
      - "This student earned a grade of: ____"

Percentage	Letter Grade
90-100	A
80-89	B
70-79	C
60-69	D
<60	F

Practice

- Start a new script.
- Define three variables:
  - Number of apples (User enters with a prompt)
  - Cost of one apple (User enters with a prompt)
  - Total cost of sale (don’t assign a value)
- Compute the total cost of the sale (number of apples * cost of one apple).
- Write a multiple condition “If” statement that states something like “Your total is more than $10”, “Your total is less than $10”, “Your total is exactly $10”, depending on the total you compute.
- Refresh your page, changing the values you enter to see if each branch works as expected.

Types of Loops

- While loop
  - Condition controlled loop
  - Condition check is at the start (so loop may never execute)

```
While (condition)
{
Actions
}
```

- Do-while loop
  - Condition controlled loop
  - Condition check is at the end (so loop will execute at least 1x)

```
do
{
Actions
}while (condition)
```

- For loop
  - Counter controlled loop

```
For (initialization, relational, increment)
{
Actions
}
```

While loops

```
While (condition)
{
Actions
}
```

- Basic Concept:
  - Test the condition
    - if it is true perform all the actions between { }
      - Repeat loop
    - if it is false skip to the end of the loop (after })
      - Do not repeat loop go onto the next portion of the program

While loops

```
<script type="text/javascript">
<!--
    var a=1, b;
    while (a <= 3) {
          b=( a <=3);
          document.write (“ <br> In loop a=“, a, b=“, b);
          a=a+1;
    }
    b=( a <=3);
    document.write (“ <br> Done a=“, a, b=“, b);
    // -->
</script>
```

Practice While Loops

```
<script type="text/javascript">

var loops = 0  //initialize loop counter
  var check = 0  //initialize condition check variable

  document.write("value of check before the loop starts:  ", check, "<br \>");
  document.write("value of loops before the loop starts:  ", loops, "<br \>");
  document.write("enter the loop", "<br \>");

while (check < 9) {
    check = check + 2;
    document.write("value of variable check:  ", check, "<br \>");
    loops = loops + 1;
    document.write("Number of loops:  ", loops, "<br \>");
  }

  document.write("exit the loop", "<br \>");

</script>
```

Practice while loop: grade assignment

Add to your script for checking grade assignments an option to check another grade (as many times as you want).

Define variable doagain = `y’.

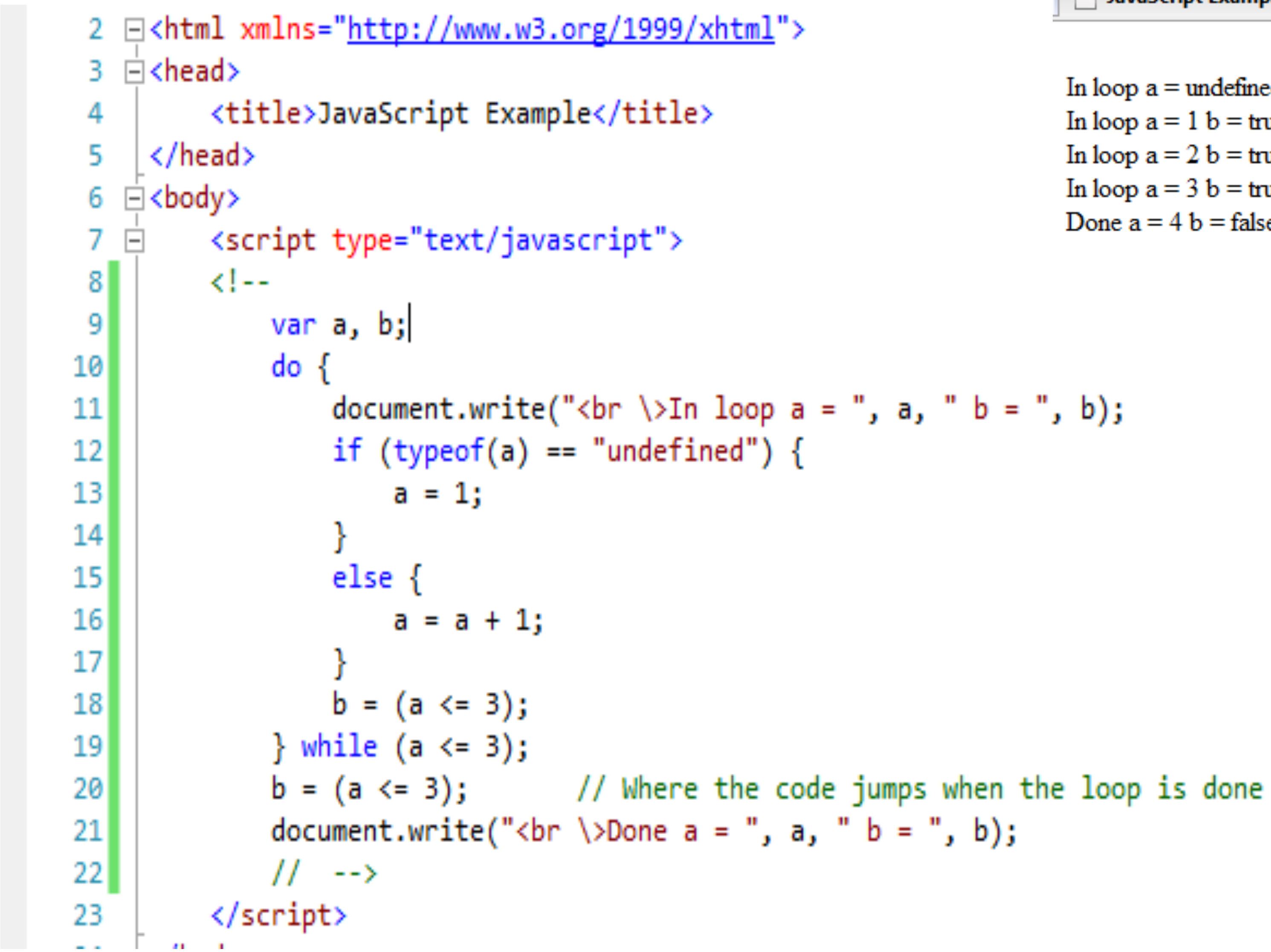
```
  while (doagain == 'y')
  {

                              …

                             ….

             while (doagain != 'y' && doagain != 'n')
              {
                    error = error + 1;
                     doagain = prompt("Do you want to try a different grade
(y/n)?");
               }

    }
```

Do-while loops

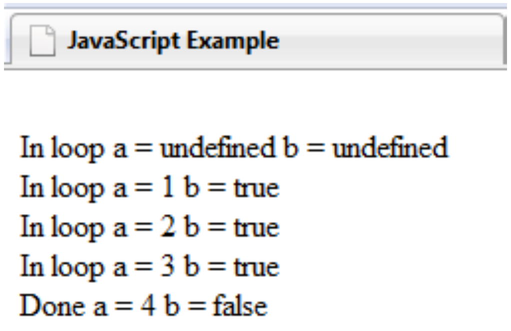
```
do
{
Actions
}while (condition)
```

- Basic Concept:
  - – Perform the actions { }
  - – Test the condition
    - if it is true repeat loop
    - if it is false skip to the end of the loop (after } and condition)
      - – Do not repeat loop go onto the next portion of the program

do-while loops



```
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>JavaScript Example</title>
</head>
<body>
    <script type="text/javascript">
    <!--
        var a, b;
        do {
            document.write("<br \>In loop a = ", a, " b = ", b);
            if (typeof(a) == "undefined") {
                a = 1;
            }
            else {
                a = a + 1;
            }
            b = (a <= 3);
        } while (a <= 3);
        b = (a <= 3);       // Where the code jumps when the loop is done
        document.write("<br \>Done a = ", a, " b = ", b);
        //  -->
    </script>
```



```
In loop a = undefined b = undefined
In loop a = 1 b = true
In loop a = 2 b = true
In loop a = 3 b = true
Done a = 4 b = false
```

Concatenation

There is a difference between using commas and + for concatenating values:

The + concatenation operator will automatically cast numbers into a string.

The , operator ***delimits*** strings and numbers

Thus, the following code lines act in very different ways:

A) document.write(“ I am “ + 21 + “ years old.”);

B) document.write(“ I am “ , 21 , “ years old.”);

The line A is the equivalent of:

var myString = “I am 21 years old.”; document.write(myString);

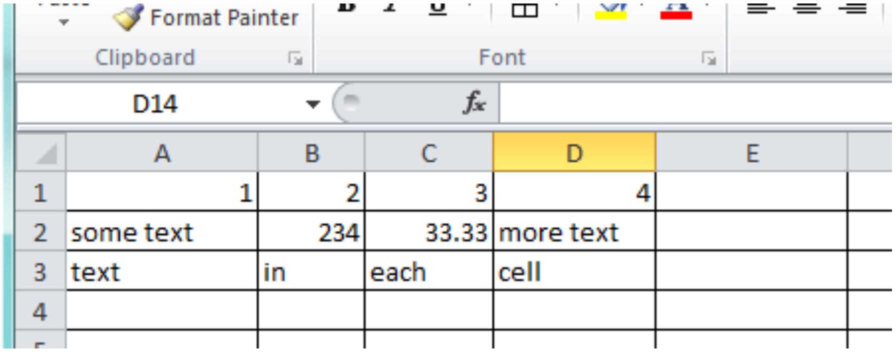
Line B is the equivalent of:

document.write(“I am “);document.write(21);document.write(“ years old“);

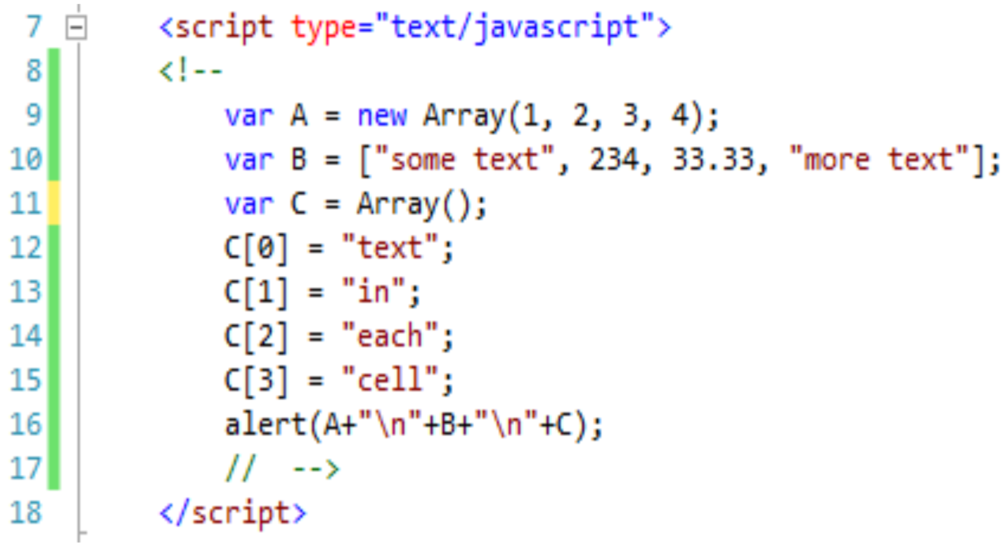
The **+** operator converts everything to a string; the **,** operator evaluates each statement independently.

Arrays

- A typical application of “loops” is to access elements in an array.
- In JavaScript, an array is an ordered list of data.
- The first row contains all numbers, the second row contains text and numbers, and the final row contains all text. So using this comparison, the first row of data could be called array A, the second row array B and the final row array C.



	A	B	C	D	E
1	1	2	3	4	
2	some text	234	33.33	more text	
3	text	in	each	cell	
4					



```
 7      <script type="text/javascript">
 8      <!--
 9          var A = new Array(1, 2, 3, 4);
10          var B = ["some text", 234, 33.33, "more text"];
11          var C = Array();
12          C[0] = "text";
13          C[1] = "in";
14          C[2] = "each";
15          C[3] = "cell";
16          alert(A+"\n"+B+"\n"+C);
17          //  -->
18      </script>
```

In the first case, line 9, we use the Array constructor with the keyword new operator included. In this case you just list the cell contents separated by commas.

In the second case, line 10, we use the array literal notation. An array literal is defined by using square brackets with each element separated by commas.

In the third case, line 11, we again use the Array constructor but in this case we only define the array and do not initialize its cells. In this case we also do not use the new operator.

There is no difference if you include the new operator. Then each element in the array is individually defined with the use of the assignment operator.

Arrays

- Notice after the array name C, there is a series of square brackets with different numbers.

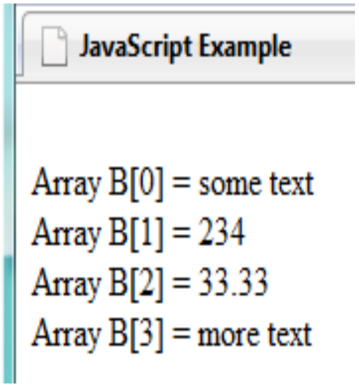
- Each different number is used to define each individual cell in the array. Each number in the square bracket is known as the index number.

- In JavaScript we start numbering the cells at zero.

- So if we return to our spreadsheet analogy Excel cell A1 would be array element A[0], Excel cell B1 would be Array element A[1], C1 would be A[2], and D1 would be A[3].

Using loops to access arrays

```
<script type="text/javascript">
<!--
  var i=0, B=[“some text”, 234, 33.33, “more text”];
  while (i < 4) {

        document.write (“ <br\> Array B[“, i ,”] = “, B[i] );
        i=i+1;
  }

  // -->
</script>
```



```
Array B[0] = some text
Array B[1] = 234
Array B[2] = 33.33
Array B[3] = more text
```