OKANAGAN

COSC 122

Computer Fluency

Iteration and Arrays

Dr. Firas Moosvi

Acknowledgement: Original slides provided courtesy of Dr. Lawrence and Dr. Abdallah Mohamed.

IMPORTANT ANNOUNCEMENT

Classes on Friday **Nov 4** and Friday **Nov. 18th**
will be **CANCELLED.**

Bonus <u>Test 3 (Nov 4)</u> and <u>Test 4 (Nov. 18)</u>
will be done completely online, on your own, without invigilation,
and no password will be required to write the test.

The same Test rules as usual apply -
I'm trusting you to do things honourably and with integrity!

There will be no one in COM 201, so don't come to class
unless you have no other quiet place to write the test!

the ++ and -- Operators

It is very common to subtract 1 or add 1 from the current value of an integer variable.

There are two operators which abbreviate these operations:

- ++ add one to the current integer variable
- - - subtract one from the current integer variable

Example:

```
var j=0;

j++;     // j = 1;  Equivalent to j = j + 1;
j--;     // j = 0;  Equivalent to j = j - 1;
```

Key Points

1) A loop repeats a set of statements multiple times until some condition is satisfied.

2) Arrays are a data structure for storing multiple items using the same name. Individual items are referenced by index.

Introductory problem!

Suppose that you need to print a string a ***100 times***.

```
I will not throw paper airplanes in class
I will not throw paper airplanes in class
I will not throw paper airplanes in class
I will not throw paper airplanes in class
I will not throw paper airplanes in class
I will not throw paper airplanes in class
I will not throw paper airplanes in class
…
```

Iteration & Looping Overview

A computer does simple operations extremely quickly.

If all programs consisted of simple statements and decisions as we have seen so far, then we would never be able to write enough code to use a computer effectively.

To make a computer **do a set of statements multiple times** we use ***looping structures***.

A ***loop*** repeats a set of statements multiple times until some condition is satisfied.

- Each time a loop is executed is called an ***iteration***.

The While Loop

The most basic looping structure is the ***while*** loop.

A while loop continually executes a set of statements **while** a condition is true.

Syntax:

```
while (<condition>) {
      <statements>
}
```

Example:

```
var j=0;
while (j <= 5){
   document.write(j);
   j=j+1;
}
```

Question: What does this print?

The For Loop

The most common type of loop is the ***for loop***. Syntax:

```
for (<initialization>; <continuation>; <next iteration>){
  <statement list>

}
```

Explanation:

- 1) initialization section - is executed *once at the start* of the loop
- 2) continuation section - is evaluated *before every loop* iteration to check for loop termination
- 3) next iteration section - is evaluated *after every loop* iteration to update the loop counter

Iteration & Looping The For Loop

Although JavaScript will allow almost any code in the three sections, there is a typical usage:

```
for (i = start; i < end; i++){
    statements
}
```

Example:

```
var i;

for (i = 0; i < 5; i++)
{  document.write(i);      // Prints 0 to 4
}
```

For Loop and While Loop

The `for` loop is like a short-hand for the `while` loop:

```
var i=0;
while (i < 10)
{   document.write(i);
    i++;
}
```

```
var i;
for (i=0; i < 10; i++)
{   document.write(i);

}
```

JavaScript Rules for Loops

The loop variable `i` must be declared.

- `i`, `j`, and `k` are used by convention, but you can pick any name you want.

The starting point of the iteration can begin anywhere, including negative numbers.

The continuation test must be an expression that results in a Boolean value. It should contain the loop variable to avoid an ***infinite loop***.

The next iteration usually changes the value of the loop variable by 1. It does not always have to be one, and it can be positive (such as +2) or negative (-1).

Common Problems – Infinite Loops

Infinite loops are caused by an incorrect loop condition or not updating values within the loop so that the loop condition will eventually be false.

Examples:

```
var i;

for (i=0; i < 10; i--){      // Should have been i++
   document.write(i);        // Infinite loop: 0,-1,-2,..
}

i = 0;
while (i < 10){
   document.write(i);        // Infinite loop: 0,0,0,..
}                            // Forgot to change i in loop
```

Common Problems – Using Brackets

A one statement loop does not need brackets, but we will ***always use brackets***. Otherwise problems may occur:

```
var i=0;
while (i <= 10)
   document.write(i);       // Prints 0 (infinite loop)
   i++;                     // Does not get here…
// Forgot brackets { and } - i++ not in loop!
```

Do not put a semi-colon at the end of the loop:

```
for (i=0; i <= 10; i++);    // Causes empty loop
{  document.write(i);       // Prints 11
}
```

Common Problems – Off-by-one Error

The most common error is to be "***off-by-one***". This occurs when you stop the loop one iteration too early or too late.

Example:

- This loop was supposed to print 0 to 10, but it does not.

```
for (i=0; i < 10; i++)
    document.write(i);  // Prints 0..9 not 0..10
```

Question: How can we fix this code to print 0 to 10?

Looping Review

A loop structure makes the computer repeat a set of statements multiple times.

- `for` loop is used when you know exactly **how many iterations** to perform
- `while` loop is used when you keep repeating the loop until **a condition is no longer true**

When constructing your loop structure make sure that:

- you have the correct brackets to group your statements
- you do not add additional semi-colons that are unneeded
- make sure your loop terminates (no infinite loop)

Remember the operators ++ and -- as short-hand notation.

For Loops

Question: What is the output of this code?

```
var i;

for (i=0; i < 10; i++) {
   document.write(i);
}
```

A) nothing

B) error

C) The numbers 0, 1, 2, …, 9

D) The numbers 0, 1, 2, …, 10

For Loops

Question: What is the output of this code?

```
var i;

for (i=2; i < 10; i--) {
    document.write(i);
}
```

A) nothing

B) infinite loop

C) The numbers 2, 3, 4, …, 9

D) The numbers 2, 3, 4, …, 10

For Loops

Question: What is the output of this code?

```
var i;

for (i=0; i <= 10; i++);
{  document.write(i);
}
```

A) nothing

B) error

C) `11`

D) The numbers 0, 1, 2, …, 10

Practice Questions: Iteration

1) How many times does each loop execute:

```
a)  for(j=0; j <= 10; j--)
b)  for(j=0; j <= 10; j++)
c)  for(j=0; j < 10; j++)
d)  for(j=-10; j <= 10; j++)
e)  for(j=0; j <= 20; j=j+2)
```

2) Write a program to print the numbers from 1 to *N*.

- a) Modify your program to only print the even numbers.

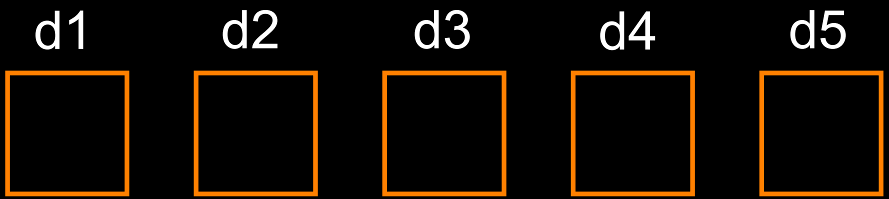
3) Write a program to calculate and print the sum of the numbers from 1 to *N*. E.g. If *N* = 4, print 10 (1+2+3+4=10).

Arrays Overview

Suppose you need many variables in your program.

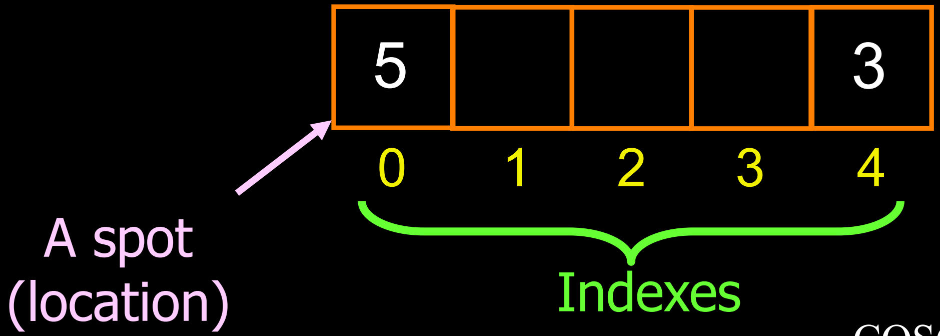
You could either create a separate name for each variable:

- `var d1, d2, d3, d4, d5;`



Or you could create an array that has multiple spots (indexes):

- `var nums = new Array(5);`
- `nums[0] = 5;`
- `nums[4]= 3;`



★ *Arrays*

An ***array*** is a collection of data items of the same type.

- Technically, **JavaScript allows an array to contain different types**, but that is not common in other programming languages.

The **name of the array is an** identifier like any other variable.

However, until you actually create the space for the array using ***new***, no array exists in memory.

- `var strings = new Array(10);`

Array Indexing

When creating an array using `new`, the number in parentheses is the number of spots in the array:

- `var a = new Array(20);` // 20 elements

Note that the first spot of the array has index 0 instead of 1.

- In the previous example, the first index is 0 and the last is 19.

When an array is first created, all its values are undefined.

To access or set a value in an array, use its index:

- `a[0] = 10;` // Sets first spot to 10
- `a[19] = a[0];` // Sets last element same as 1st

Array Details

If you provide an array index outside of the valid range, JavaScript will automatically re-size the array for you (for updates) or returned undefined (for reads).

- This is not common behavior. Most languages generate an error called an ***exception***.

To get the length of an array in your program:

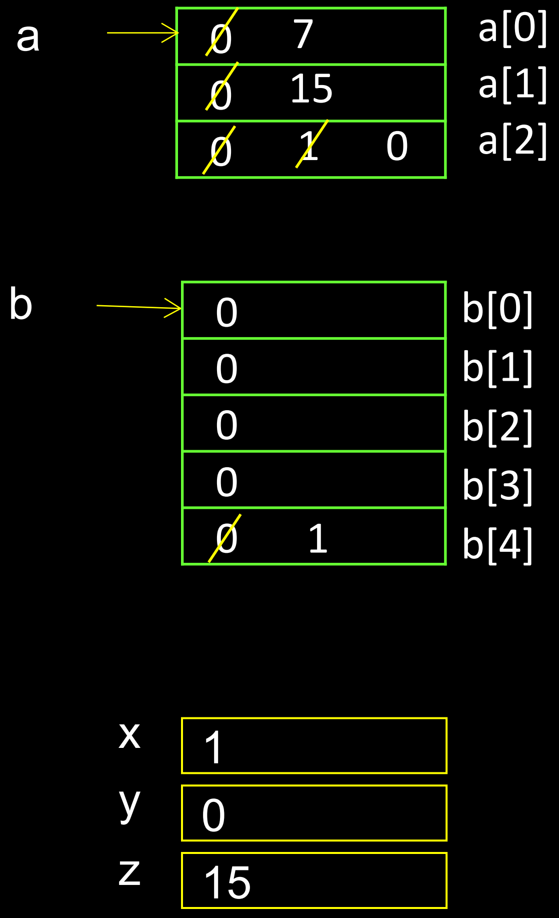
- `var a = new Array(20);`
- `var size = a.length;`   // Returns 20

Practice

```
var a = new Array(3);
a[0]= 0; a[1]=0; a[2]=0;

var b = new Array(5);
b[0]= 0; b[1]=0; b[2]=0;
b[3]= 0; b[4]=0;

a[0] = 7;
a[1] = a[0] + 8;
a[2]++;
var x = a[2];
var y = a[2] - 1;
var z = a[2-1];
b[b.length-1]++;
```



Arrays

Question: What is the size of this array?

```
var myArray = new Array(10);
```

A) error

B) 10

C) 9

D) 11

Arrays

Question: What is the size of this array?

```
var myArray = new Array[10];
```

A) error

B) 10

C) 9

D) 11

Arrays

Question: What are the contents of this array?

```
var myArray = new Array(4);
myArray[3] = 1;
myArray[2] = 2;
myArray[1] = 3;
myArray[0] = 4;
```

A) error

B) 0, 1, 2, 3

C) 1, 2, 3, 4

D) 4, 3, 2, 1

Processing Arrays using Loops

Loops are usually used to process arrays. For example:

```
// create the array
var names = new Array(4);

// initialize the array
var i;
for(i = 0; i < names.length; i++)
       names[i] = "no name yet!";

// print all elements in the array
document.write("<p>Your array has the following
values:<ul>");
for(i = 0; i < names.length; i++)
    document.write("<li>" + names[i] + "</li>")
document.write("</ul></p>");
```

Application of Arrays

We can use arrays to create a web page that shows a slideshow of our favorite pictures.

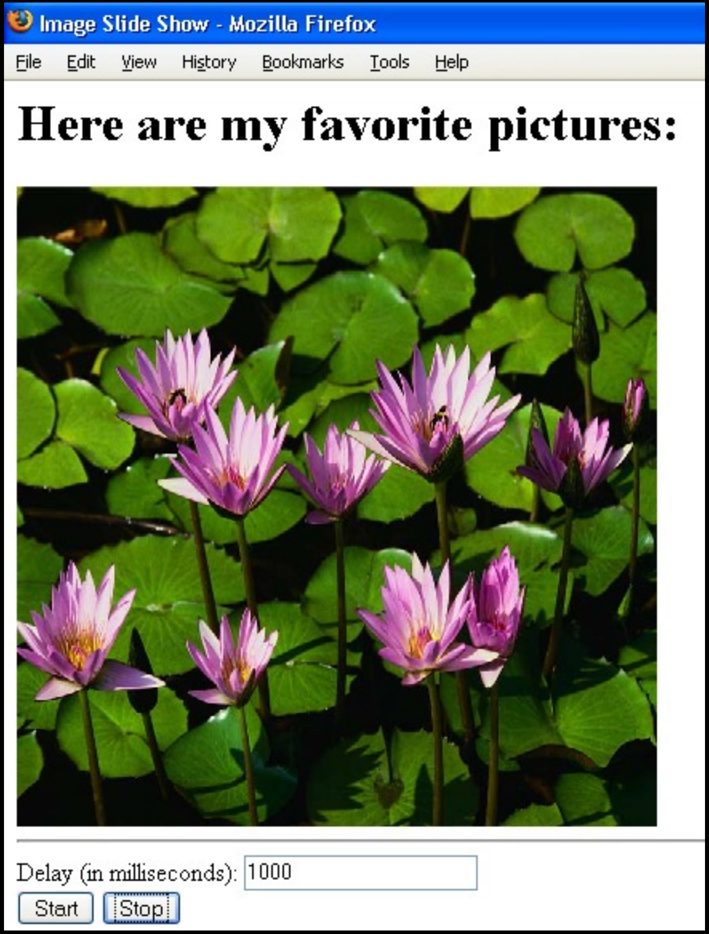


Image Slide Show - Mozilla Firefox
File Edit View History Bookmarks Tools Help
Start
Stop

Here are my favorite pictures:

Delay (in milliseconds): 1000

Practice Questions: Arrays

1) Create an array with name `myArray` that has 20 spots.

2) Set the value of the 1st spot to 10.

3) Set the value of the last spot to 1.

4) How do you know how many spots are in an array?

5) Create an array that has 10 spots. Put the numbers from 1 to 10 in the array.

Conclusion

A ***loop*** allows the repetition of a set of statements multiple times until some condition is satisfied.

- We will primarily use for loops that have 3 components:
  - initialization - setup iteration variable start point
  - continuation - use iteration variable to check if should stop
  - next iteration - increment/decrement iteration variable

Arrays are a data structure for storing multiple items using the same name. Arrays are often used with loops, as a loop can access each individual item by its index.

Objectives

- Define: loop, iteration
- Explain the difference between the while and for loops.
- Explain what ++ and -- operators do.
- Be able to use a for loop structure to solve problems.
- Define: infinite loop
- Define: array
- Be able to use arrays to solve problems.