Προγραμματισμός Διαδικτύου

Δρ. Μηνάς Δασυγένης
mdasygenis@uowm.gr

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.

Σκοπός ενότητας

- ✓ Ικανότητα δόμησης των προγραμμάτων με συναρτήσεις.
- ✓ Χρήση και επεξεργασία πινάκων και δεδομένων.
- ✓ Χρήση δομών ελέγχου και επανάληψης.

Προγραμματισμός Διαδικτύου

Javascript #2 : Συναρτήσεις, Πίνακες

Περιεχόμενα

1. Συναρτήσεις
    - Υπολογισμός τετραγώνου
    - Εύρεση μεγίστου
    - Υπολογισμός παραγοντικού

2. Πίνακες
    - Δήλωση και δέσμευση
    - Δομές πρόσβασης πινάκων

Δομήστε όλα τα προγράμματά σας με συναρτήσεις

Κίνητρα για διαμόρφωση ενός προγράμματος με συναρτήσεις.

- Κάνει την ανάπτυξη του προγράμματος πιο εύχρηστη.
- Δίνει τη δυνατότητα *επαναχρησιμοποίησης του λογισμικού.*
- Τα προγράμματα μπορούν να δημιουργηθούν από τυπικές συναρτήσεις αντί να χτιστούν από προσαρμοσμένο κώδικα.

Παράδειγμα: **parseInt (), parseFloat ()**

Οι συναρτήσεις πρέπει να περιορίζονται στην εκτέλεση μιας ενιαίας, καλά καθορισμένης εργασίας.

- Αποφύγετε την επανάληψη κώδικα στο πρόγραμμα.

  - Μην εφεύρετε ξανά τον τροχό.
  - Εξοικονόμηση χρόνου.

Είναι πολύ απλό να ορίσουμε μια συνάρτηση στη Javascript

- Μορφή ορισμού συνάρτησης.

```
function  όνομα συνάρτησης (λίστα παραμέτρων)
{
    Δηλώσεις
}
```

- Όνομα συνάρτησης - οποιοδήποτε έγκυρο αναγνωριστικό.
- Λίστα παραμέτρων - λίστα χωρισμένη με κόμματα που περιέχει ονόματα παραμέτρων που λαμβάνονται από τη λειτουργία όταν καλείται.
- Εάν η λειτουργία δεν λαμβάνει τιμές, παραμένει η λίστα παραμέτρων άδεια.

Λειτουργικά θέματα των συναρτήσεων

- Σώμα συνάρτησης ή μπλοκ:

  - Δηλώσεις μέσα σε στηρίγματα (braces).

- Έλεγχος

- Επιστρέφθηκε στο σημείο στο οποίο κλήθηκε η συνάρτηση.

- Η λειτουργία if δεν επιστρέφει αποτέλεσμα
  - Όταν φτάσει στο δεξί στήριγμα, εκτελείται η δήλωση επιστροφής.

- Η λειτουργία if επιστρέψει ένα αποτέλεσμα
  - Όταν εκτελείται η ***έκφραση επιστροφής***.
    - Επιστρέφει την τιμή των εκφράσεων στον καλούντα.

- Ένα όρισμα στην κλήση συνάρτησης για κάθε παράμετρο της συνάρτησης.

Παράδειγμα τετραγώνου αριθμών

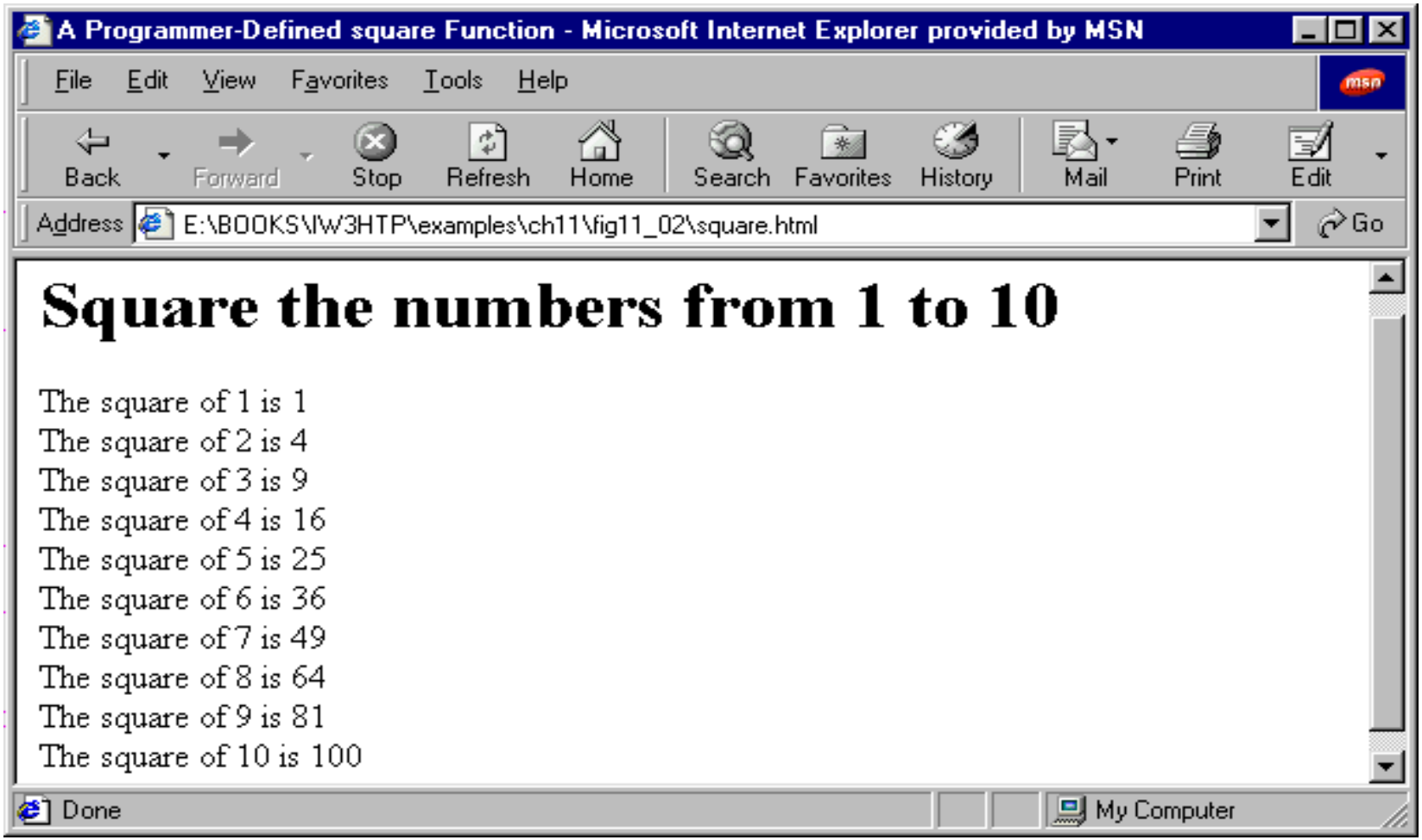
```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<!-- Fig. 11.2: SquareInt.html -->

<HEAD>
<TITLE>A Programmer-Defined square Function</TITLE>
<SCRIPT LANGUAGE = "JavaScript">
   document.writeln("<H1>Square the numbers from 1 to 10 </H1>");
//  square the numbers from 1 to 10
   for ( var x = 1; x <= 10; ++x )
         document.writeln( "The square of " +x+ " is " +square( x ) +
"<BR>" );

//  square function definition

function square( y )
{
   return y * y;
}

</HEAD><BODY></BODY>
</HTML>
```

Καλεί τη συνάρτηση square και περνάει την τιμή του x.

τετράγωνο Η μεταβλητή y παίρνει την τιμή της μεταβλητής x.

Η δήλωση επιστροφής περνά την τιμή του y * y πίσω στη λειτουργία κλήσης.

Έξοδος προγράμματος υπολογισμού τετραγώνου αριθμών



(α) Η συνάρτηση μπορεί να είναι μέθοδος κλάσης (β) σημεία προσοχής

- Μέθοδος **Math.max(y, z)**

  - Επιστρέφει τη μεγαλύτερη από τις δύο εισάγουσες τιμές.

- Όταν γράφετε μια συνάρτηση, **ΜΗΝ**

  - Ξεχάσετε να επιστρέψετε μια τιμή εάν η λειτουργία υποτίθεται ότι επιστρέφει μια τιμή.

  - Ξεχάσετε να περιβάλλετε το σώμα της συνάρτησης με άγκιστρα.

  - Περάσετε ένα όρισμα στη συνάρτηση που δεν είναι συμβατή με τον αναμενόμενο τύπο δεδομένων.

Παράδειγμα συνάρτησης μεγίστου

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<!-- Fig. 11.3: maximum.html -->
<HEAD>
<TITLE>Finding the Maximum of Three Values</TITLE>
<SCRIPT LANGUAGE = "JavaScript">
        var input1= window.prompt("Enter first number", "0" );
        var input2 = window.prompt("Enter second number", "0" );
        var input3= window.prompt("Enter third number", "0" );

        var value1 = pasreFloat(input1);
        var value2 = pasreFloat(input2);
        var value3 = pasreFloat(input3);

        var maxValue = maximum( value1,value2, value3) ;

        document.writeln( "First number: " +value1+ "<BR>Second number: " + value2 +
        "<BR>Third number: " + value3 + "<BR>Maximum: " + maxValue) ;

        function maximum(x,y,z) {
                return Math.max( x, Math.max( y, z ) );
        }
</SCRIPT>
</HEAD>
<BODY>
<P>Click Refresh (or Reload) to run the script again</P>
</BODY>
</HTML>
```

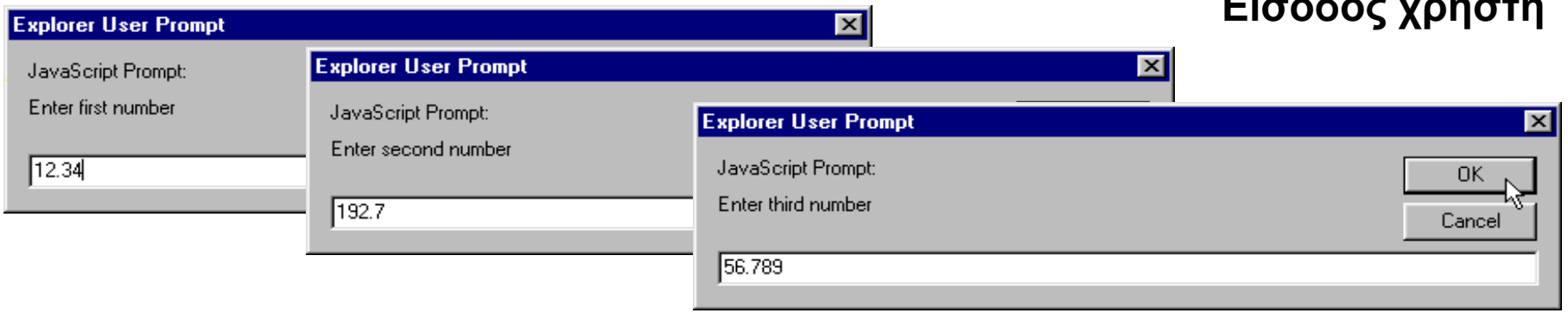
Καλεί τη συνάρτηση maximum και περνάει την τιμή των μεταβλητών value1, value2 και value3.

Οι μεταβλητές x, y και z παίρνουν την τιμή των μεταβλητών **value1**, **value2** και **value3**, αντίστοιχα.

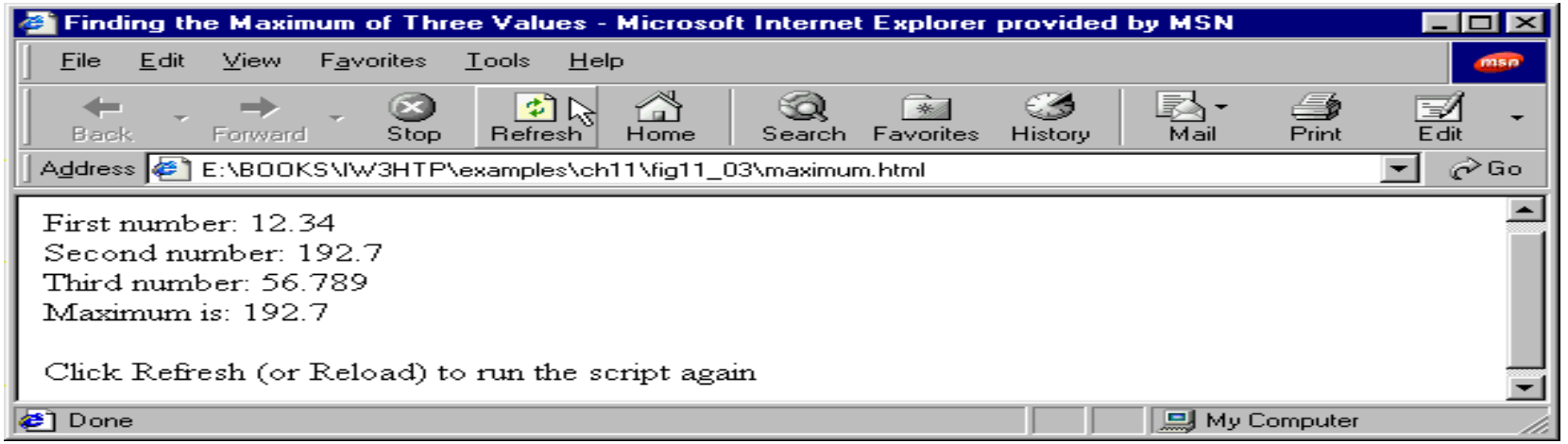
Η συνάρτηση maximum επιστρέφει το μεγαλύτερο από τους δύο ακεραίους που περνάνε σε αυτή.

Παράδειγμα συνάρτησης μεγίστου

Είσοδος χρήστη



Έξοδος προγράμματος



Συναρτήσεις & Αναδρομή
Υπολογισμός παραγοντικού

- Παράγοντες
  - Προϊόν υπολογισμού $n \cdot (n - 1) \cdot (n - 2) \cdot \ldots \cdot 1$
  - Εξελικτική προσέγγιση:

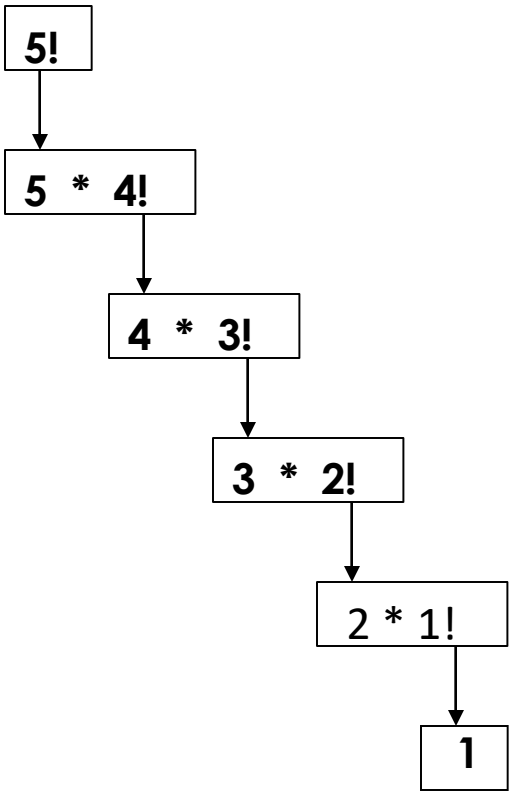
```
var  factorial  =  1;

for  (var counter= number; counter>=1; --counter)
       factorial  *=  counter;
```

- Σημειώστε ότι κάθε παράγοντας είναι μικρότερος από τον προηγούμενο παράγοντα.

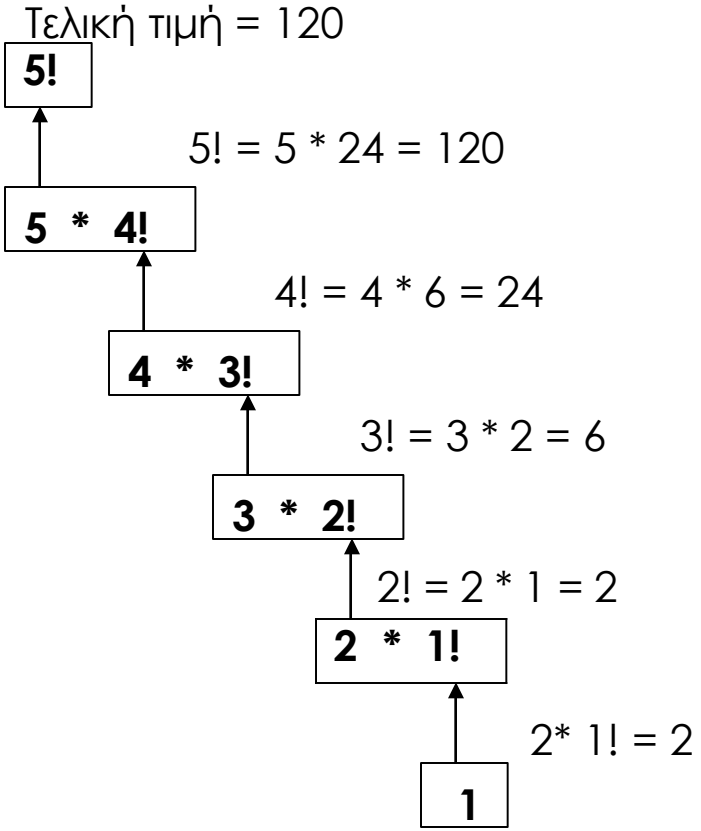
  - Σταματάει στο 1: βασική περίπτωση.
  - Ιδανικός υποψήφιος για αναδρομική λύση.

Κάποια προβλήματα (π.χ. παραγοντικό), λύνονται πιο εύκολα με αναδρομή

Αναδρομικός Υπολογισμός του 5!



Πορεία αναδρομικών κλήσεων



Οι τιμές που επιστρέφονται από κάθε επαναλαμβανόμενη κλήση

Υπολογισμός παραγοντικού σε Javascript

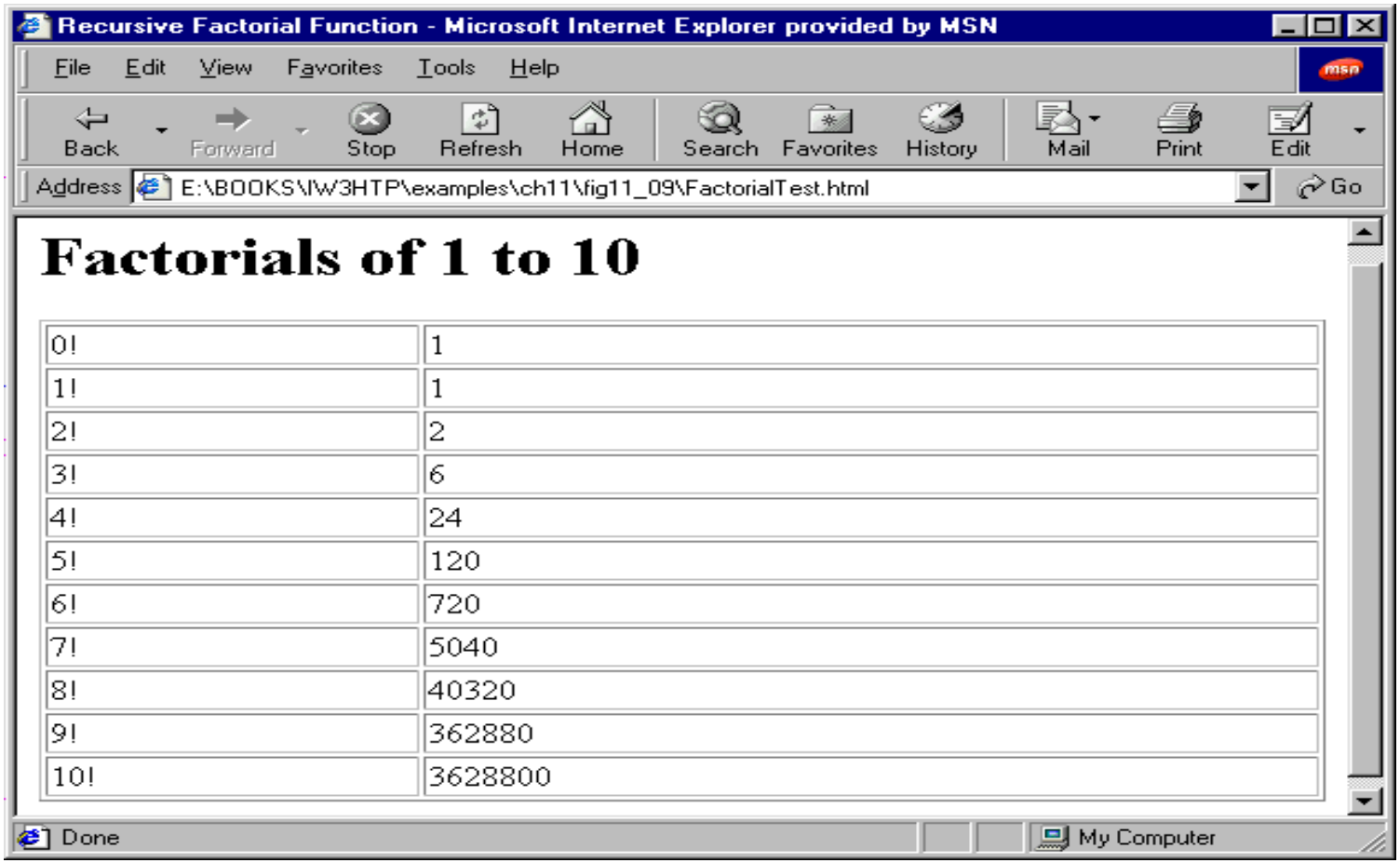
```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0
Transitional//EN">
<HTML>
<!-- Fig. 11.9: FactorialTest.html --> 4
<HEAD>
<TITLE>Recursive Factorial Function</TITLE>

   <SCRIPT LANGUAGE = "JavaScript">
         document.writeln( "<H1>Factorials of 1 to 10</H1>" );
         document.writeln( "<TABLE BORDER = '1' WIDTH = '100%'>" );
         for ( var i = 0; i <= 10; i++ )
          document.writeln("<TR><TD>" + i + "!</TD><TD>" +factorial( i ) + "</TD></TR>" );
         document.writeln( "</TABLE>" );
         // Recursive definition of function factorial
         function factorial( number ){
         if ( number <= 1 )   // base c
                   return 1;
          else
           } return number * factorial( number - 1 );
 </SCRIPT>
</HEAD><BODY></BODY>
</HTML>
```

Καλεί τη συνάρτηση factorial και περνάει την τιμή του i.

Η μεταβλητή number παίρνει την τιμή από τη μεταβλητή i.

Καλεί τη συνάρτηση factorial και περνάει σε αυτή 1 λιγότερο από την τρέχουσα τιμή της μεταβλητής number.

Έξοδος του προγράμματος υπολογισμού παραγοντικού



Συναρτήσεις με καθολική (global) τοπικότητα

- Οι καθολικές συναρτήσεις αποτελούν μέρος του JavaScript.

  Καθολικό αντικείμενο (Global object)

- Περιέχει όλες τις καθολικές μεταβλητές στο σενάριο.

- Πολλοί προγραμματιστές αναφέρονται σε αυτές τις συναρτήσεις ως μέθοδοι.

  Καθολικές συναρτήσεις και συναρτήσεις που ορίζονται από το χρήστη, είναι μέρος του Global object.

- Δεν χρειάζεται να χρησιμοποιήσετε άμεσα το **Καθολικό** αντικείμενο

Η JavaScript το κάνει για εσάς.

Καθολικές συναρτήσεις της Javascript

- Καθολικές συναρτήσεις (συνέχεια)

- **isNaN**

- Παίρνει αριθμητικό όρισμα και επιστρέφει **true** αν η τιμή δεν είναι ένας αριθμός. Διαφορετικά επιστρέφει **false**.
- Συνήθως χρησιμοποιείται με τιμή επιστροφής της **parseInt** και **parseFloat** για να διαπιστωθεί αν το αποτέλεσμα είναι σωστή αριθμητική τιμή.

- **parseFloat**

- Παίρνει όρισμα συμβολοσειράς και μετατρέπει την αρχή της συμβολοσειράς σε τιμή κυμαινόμενου σημείου.
- Αν η μετατροπή είναι ανεπιτυχής, επιστρέφει **NaN.** Διαφορετικά, επιστρέφει την τιμή μετατροπής.
- (π.χ. parseFloat ("abc123.45") επιστρέφει NaN, και parseFloat ("123.45abc") επιστρέφει την τιμή 123.45).

Καθολικές συναρτήσεις της Javascript

Καθολικές συναρτήσεις (συνέχεια)

- **parseInt**

- Παίρνει όρισμα συμβολοσειράς και προσπαθεί να μετατρέψει την αρχή της συμβολοσειράς σε ακέραιη τιμή.
- Αν η μετατροπή δεν είναι επιτυχής, επιστρέφει **NaN.** Διαφορετικά, επιστρέφει την τιμή μετατροπής.
- Παίρνει ένα προαιρετικό δεύτερο όρισμα μεταξύ 2 και 36 προσδιορίζοντας το *Radix* (ή *βάση)* του αριθμού.
- (π.χ. parseInt ("abc123") επιστρέφει NaN και parseInt ("123abc") επιστρέφει την ακέραια τιμή 123) .

Εισαγωγή στους πίνακες

- Πίνακες

  - Δομές δεδομένων που αποτελούνται από σχετικά στοιχεία δεδομένων *(συλλογές από* αντικείμενα δεδομένων)

- Οι πίνακες στη JavaScript είναι "δυναμικές" οντότητες.

  - Μπορούν να αλλάξουν μέγεθος αφού δημιουργηθούν.

Ο δείκτης (index) αρχίζει από το 0

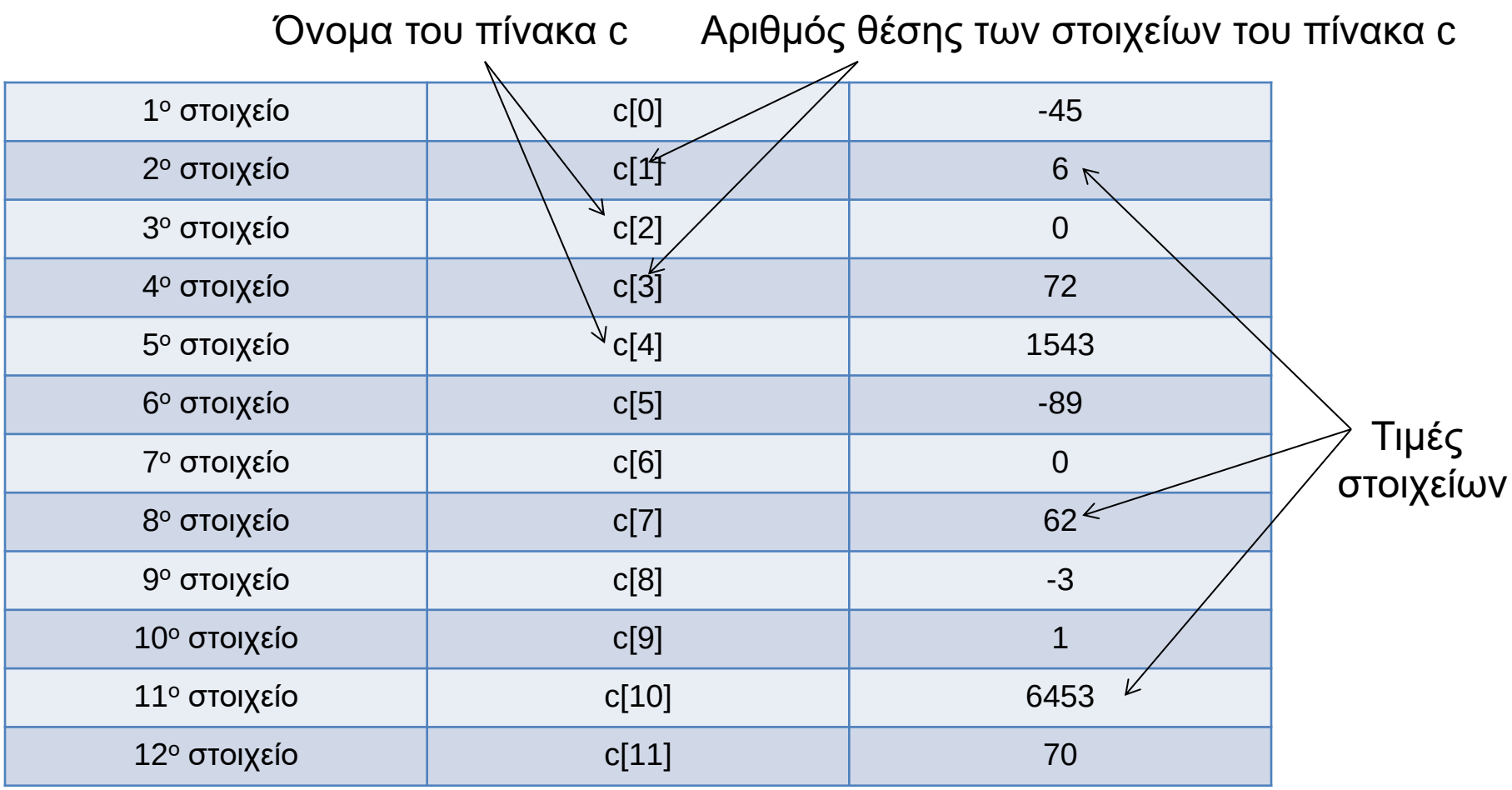
- Το πρώτο στοιχείο σε κάθε πίνακα είναι το *μηδενικό* (0th) *στοιχείο.*
  - Ως εκ τούτου, το στοιχείο **n** θα έχει τιμή δείκτη του **(n-1).**

- Το μήκος ενός πίνακα προσδιορίζεται από την έκφραση:
  όνομα_πίνακα .length

- Αγκύλες

  - Χρησιμοποιούνται για να περικλείουν τον δείκτη ενός πίνακα.
  - Έχουν την ίδια προτεραιότητα στις συναρτήσεις JavaScript ως παρενθέσεις.

Ένα παράδειγμα ενός πίνακα



1ο στοιχείο	c[0]	-45
2ο στοιχείο	c[1]	6
3ο στοιχείο	c[2]	0
4ο στοιχείο	c[3]	72
5ο στοιχείο	c[4]	1543
6ο στοιχείο	c[5]	-89
7ο στοιχείο	c[6]	0
8ο στοιχείο	c[7]	62
9ο στοιχείο	c[8]	-3
10ο στοιχείο	c[9]	1
11ο στοιχείο	c[10]	6453
12ο στοιχείο	c[11]	70

Δήλωση και δέσμευση πινάκων

- Για να δηλώσω 12 στοιχεία για πίνακα ακεραίων **c.**

  var c = newArray(12);

  - Αυτό μπορεί επίσης να γίνει σε 2 βήματα:

    var c;
    c = newArray(12);

  - Όταν δηλώνονται πίνακες, τα στοιχεία δεν αρχικοποιούνται.

- Διατήρηση μνήμης
  - Χρησιμοποιήστε μία μόνο δήλωση:

  var b = newArray(100), x = newArray(27);

Εξασφαλίζει 100 στοιχεία για πίνακα **b,** 27 στοιχεία για πίνακα **x.**

Παράδειγμα πινάκων

- Οι πίνακες μπορούν να αρχικοποιηθούν με υπάρχοντα στοιχεία.

```
var n1=newArray(5) ;
```

- Αρχικοποιεί πίνακα n1 με 5 στοιχεία.

- Οι πίνακες μπορούν επίσης να αρχικοποιηθούν χωρίς στοιχεία.

```
var n2=newArray() ;
```

- Αρχικοποιεί έναν άδειο πίνακα.

- Οι τιμές των στοιχείων μπορούν να τυπωθούν χρησιμοποιώντας την κανονική μέθοδο writeln.

```
document.writeln("The value is " +n2[2]) ;
```

Παράδειγμα πινάκων

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<!-- Fig. 12.3: InitArray.html --> 4
<HEAD>
<TITLE>Initializing an Array</TITLE>

   <SCRIPT LANGUAGE = "JavaScript">

         function initializeArray() {
                  var n1 = new Array( 5 ) ;
                  var n2 = new Array() ;

                  //assign values to each element of Array n1
                  for ( var i=0; i<n1.length; i++)
                           n1[i]=i;
                  //create and initialize five-elemets in Array n2
                  for (i=0; i<5; i++)
                           n2[i]=i;

                  outputArray("Array n1 contans", n1) ;
                  outputArray("Array n2 contains", n2) ;
         }
```

Ο πίνακας n1 έχει 5 στοιχεία.

Ο πίνακας n2 είναι ένας άδειος πίνακας.

Η for αρχικοποιεί τα στοιχεία του πίνακα n1 σε τιμές από 0 έως 4.

Η for προσθέτει 5 στοιχεία στον πίνακα n2 και τα αρχικοποιεί με τιμές από 0 έως 4.

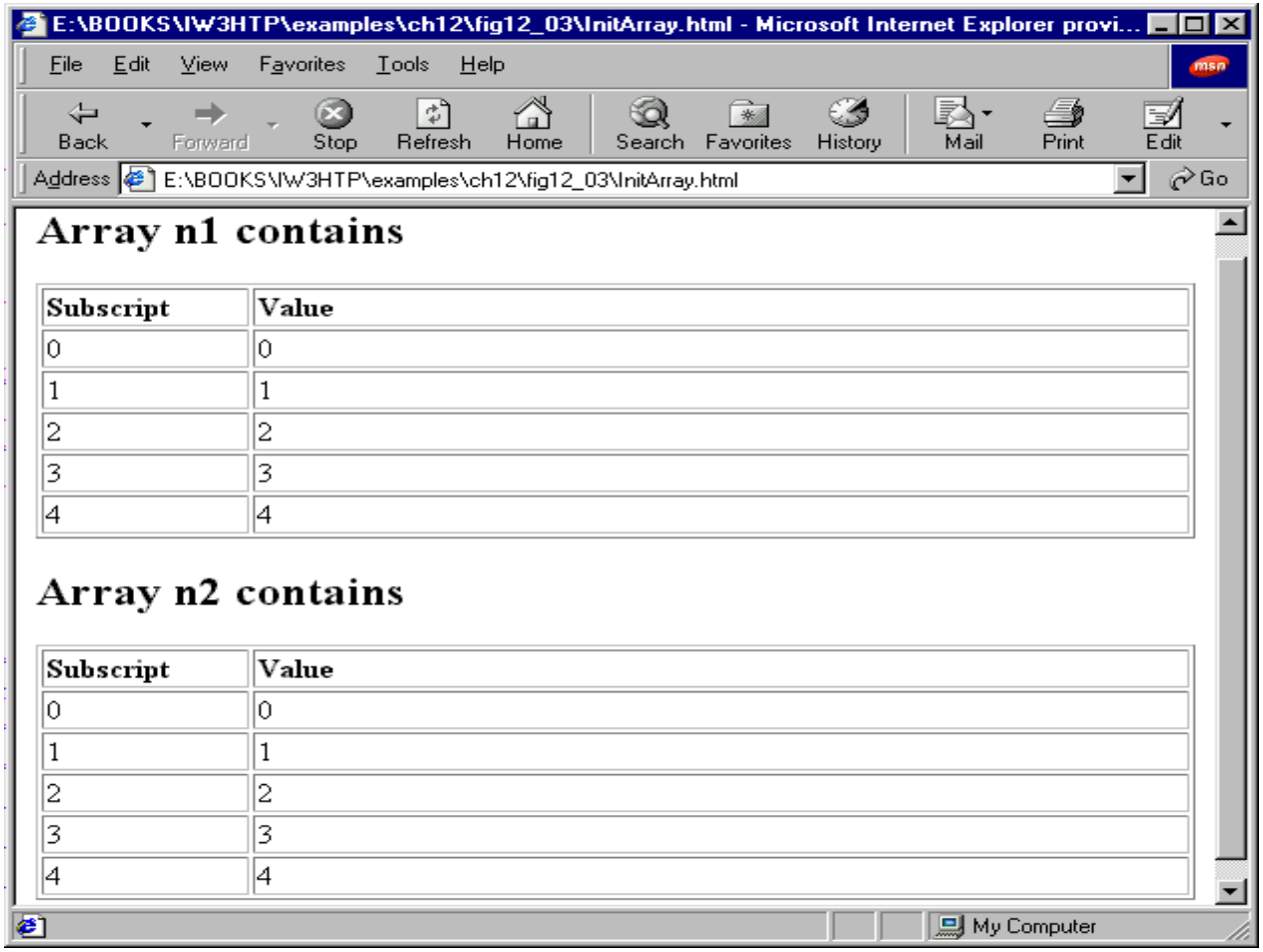
Κάθε συνάρτηση εμφανίζει τα περιεχόμενα του αντίστοιχου πίνακα σε έναν πίνακα html.

Παράδειγμα πινάκων

```
// output "header" followed by a two-column table
// cotaining subscripts and elements of "theArray"

function outputArray( header, the Array )
{
        document.writeln( "<H2>" +header+ "</H2>" ) ;
        document.writeln("<TABLE BORDER ='1' WIDTH='100%'>") ;
        document.writeln("<TR><TD WIDTH='100'><B>Subscript</B>"
                                +"<TD><B>Value</B></TR>");

        for ( var i=0; i<theArray.length; i++) {
                document.writeln("<TR><TD>" +i+ "<TD>"
                                        +theArray[i]+"</TR>") ;

                document.writeln("<TABLE>");
}
```

Την πρώτη φορά που καλείται η συνάρτηση outputArray, η μεταβλητή header παίρνει την τιμή "Array n1 contains" και η μεταβλητή theArray παίρνει την τιμή του n1.

Έξοδος προγράμματος δήλωσης πινάκων



Οι πίνακες μπορούν να αρχικοποιηθούν κατά τη δήλωση

- Τα στοιχεία ενός πίνακα μπορούν να διανεμηθούν και να αρχικοποιηθούν στη δήλωση του πίνακα. Αυτό μπορεί να γίνει με δύο τρόπους.

- Για να αρχικοποιήσετε έναν πίνακα n με 5 γνωστά στοιχεία:

```
var n = [10,20,30,40,50] ;
```

- Χρησιμοποιεί μια λίστα αρχικοποιήσεων διαχωρισμένη με κόμματα που περικλείεται σε αγκύλες.

```
var n =newArray(10,20,30,40,50) ;
```

Κατά την αρχικοποίηση μπορούμε να παραλείψουμε κάποιες τιμές

- Για να διατηρήσετε ένα χώρο σε έναν πίνακα για μια απροσδιόριστη τιμή.

  - Χρησιμοποιήστε ένα κόμμα ως κάτοχο τοποθεσίας στη λίστα προετοιμασίας.

```
var n = [10,20, , 40,50] ;
```

  Δημιουργεί 5 στοιχεία πίνακα χωρίς τιμή για το στοιχείο n[2] μέχρι αυτή να δοθεί.

Παράδειγμα πινάκων

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<!-- Fig. 12.4: InitArray2.html -->
<HEAD>
<TITLE>Initializing an Array</TITLE>

   <SCRIPT LANGUAGE = "JavaScript">

         function start() {
         //initializer list specifies number of elements and value for aech element

                  var colors = new Array( "cyan", "magenta", "yellow", "black" ) ;
                  var integers1 = [2,4,6,8] ;
                  var integers2 = [2,,,8] ;

                  outputArray("Array colors contans", colors) ;
                  outputArray("Array integers1 contains", integers1) ;
                  outputArray("Array integers2 contains", integers2) ;

         }
```

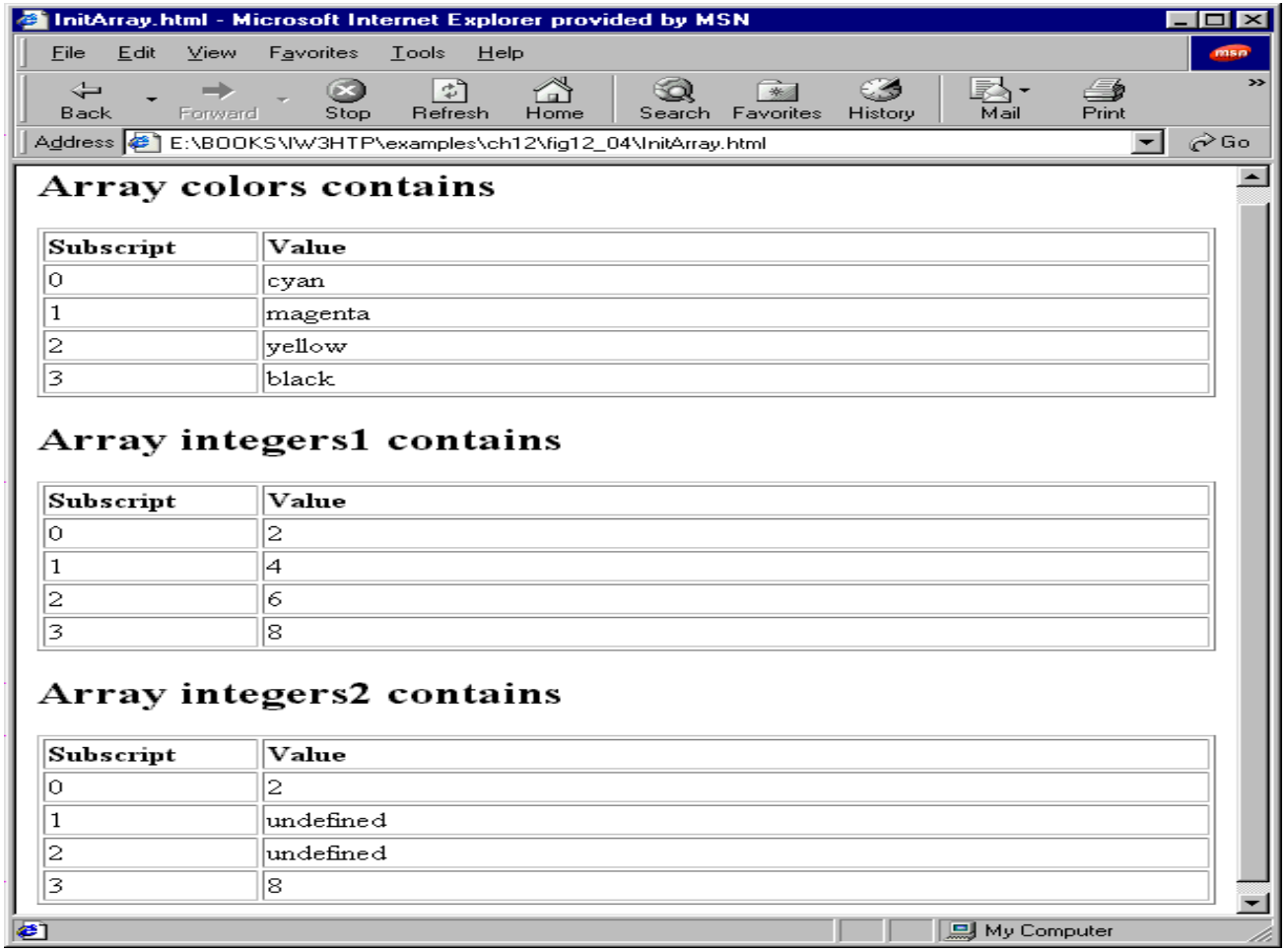
Ο πίνακας integers1 αρχικοποιείται χρησιμοποιώντας μία λίστα αρχικοποίησης.

Δύο τιμές του πίνακα integers2 δεν δηλώνονται.

Παράδειγμα πινάκων

```
// output “header” followed by a two-column table
// cotaining subscripts and elements of “theArray”

function outputArray( header, the Array )
{
        document.writeln( “<H2>” +header+ “</H2>” ) ;
        document.writeln(“<TABLE BORDER =‘1’ WIDTH=‘100%’>”) ;
        document.writeln(“<TR><TD WIDTH=‘100’><B>Subscript</B>”
                                +”<TD><B>Value</B></TR>”);

        for ( var i=0; i<theArray.length; i++) {
                document.writeln(“<TR><TD>” +i+ “<TD>”
                                        +theArray[i]+”</TR>”) ;

                document.writeln(“<TABLE>”);
}
</SCRIPT>
</HEAD><BODYONLOAD = “start()”><BODY>
</HTML>
```

Έξοδος προηγούμενου προγράμματος πινάκων



Δομή πρόσβασης σε πίνακα for/in

- Δομή επανάληψης for / in .

  - Επιτρέπει σε ένα σενάριο να εκτελέσει μια εργασία για κάθε στοιχείο ενός πίνακα.

  - Επίσης είναι γνωστό ως προσπέλαση στα στοιχεία του πίνακα.

- Μορφή:

```
for (var element in the Array)
        total2 += theArray[element] ;
```

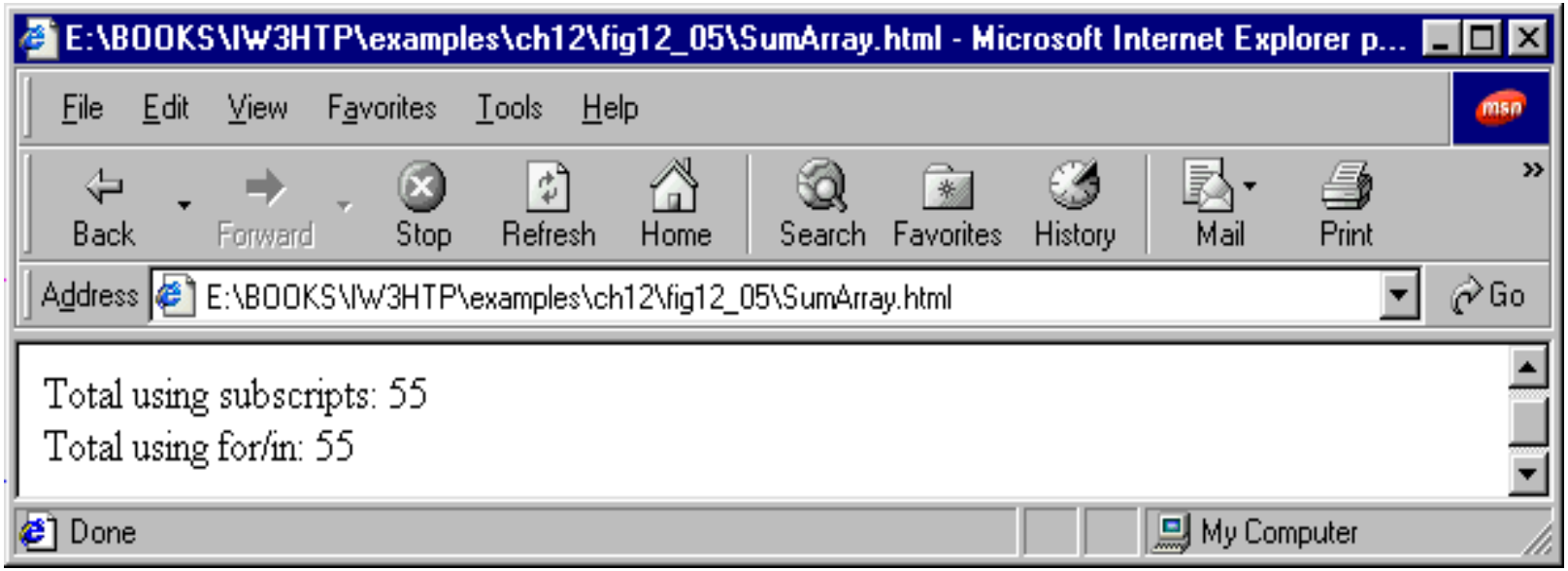
  - Προσθέτει την τιμή κάθε στοιχείου του πίνακα theArray στη μεταβλητή total2.

  - Η δομή τελειώνει μετά το τέλος της προσπέλασης για κάθε στοιχείο του πίνακα theArray.

Παράδειγμα αθροίσματος στοιχείων πίνακα

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<!-- Fig. 12.5: SumArray.html -->
<HEAD>
<TITLE>Sum the elements of an Array</TITLE>

   <SCRIPT LANGUAGE = "JavaScript">
         function start() {
                  var theArray=[1,2,3,4,5,6,7,8,9,10] ;
                  var total1=0, total2=0;
                  for(var i=0; i<theArray.length; i++)
                           total1+= theArray[i] ;
                  document.writeln("Total using subscripts: " +total1) ;

                  for(var element in theArray)
                           total2 += theArray[element] ;
                  document.writeln("<BR>Total using for/in: " +total2) ;
         }
   </SCRIPT>
</HEAD><BODY ONLOAD = "start()"></BODY>
</HTML>
```

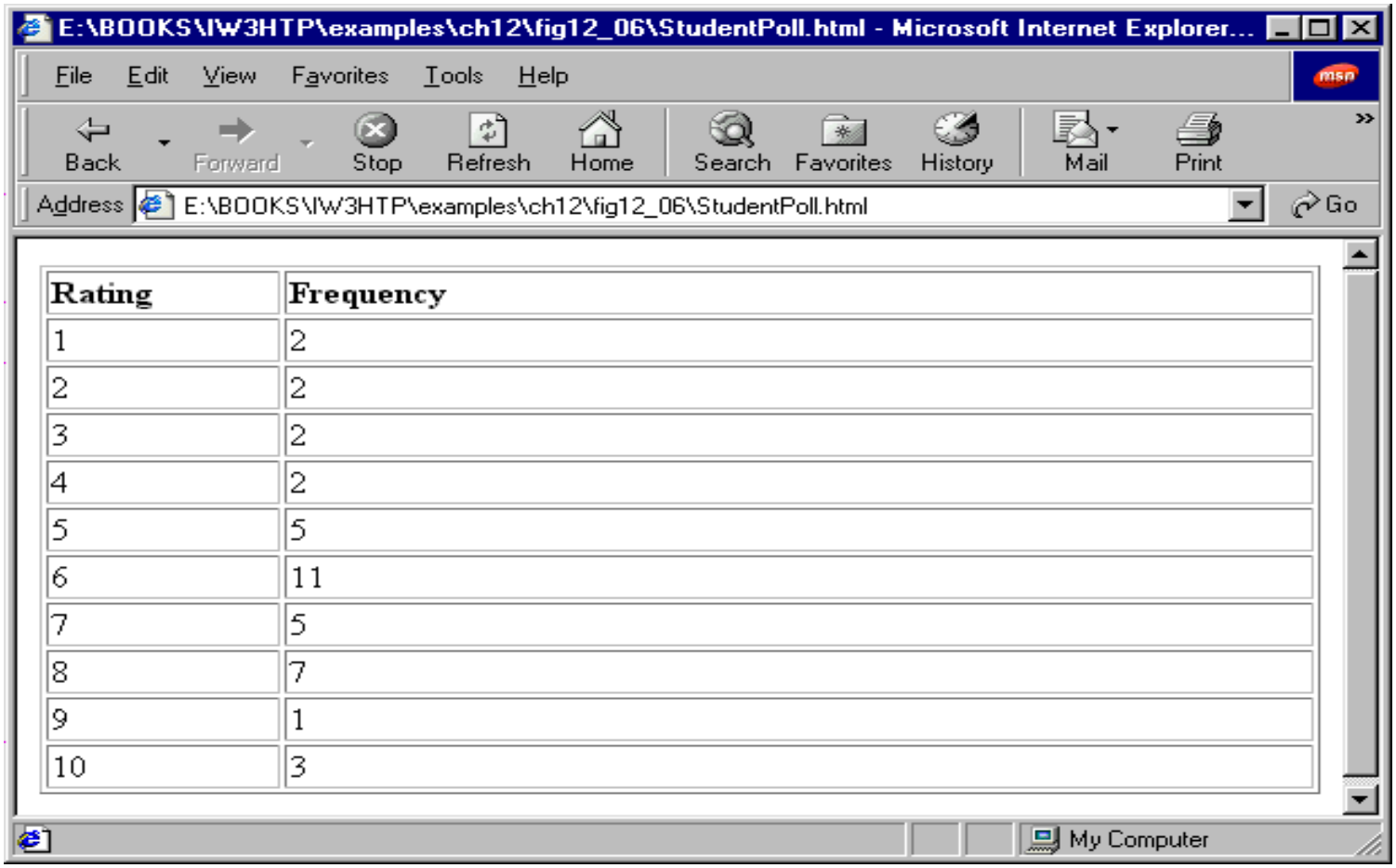
Έξοδος προγράμματος αθροίσματος στοιχείων πίνακα



Παράδειγμα προγράμματος ψηφοφορίας φοιτητών

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<!-- Fig. 12.6: StudentPoll.html -->
<HEAD>
<TITLE>Student Poll PRogram</TITLE>

   <SCRIPT LANGUAGE = "JavaScript">
          function start() {
                    var responses = [ 1, 2, 6, 4, 8, 5, 9, 7, 8, 10,1, 6, 3, 8, 6, 10, 3, 8, 2, 7,6, 5, 7, 6, 8, 6, 7, 5, 6, 6,
                                                                  5, 6, 7, 5, 6, 4, 8, 6, 8, 10 ];
                    var frequency = [ , 0, 0, 0, 0, 0, 0, 0, 0, 0, 0 ];
                    for ( var answer in responses )
                              ++frequency[ responses[ answer ] ];

                    document.writeln( "<TABLE BORDER = '1' WIDTH = '100%'>" );
                    document.writeln( "<TR><TD WIDTH = '100'><B>Rating</B>" + "<TD><B>Frequency</B></TR>"
);

                    for ( var rating = 1; rating < frequency.length; ++rating )
                              document.writeln( "<TR><TD>" + rating + "<TD>" +frequency[ rating ] +
"</TR>" );

                    document.writeln( "</TABLE>" );

</SCRIPT>
</HEAD><BODY ONLOAD = “start()”></BODY>
</HTML>
```

Έξοδος προγράμματος ψηφοφορίας φοιτητών



Rating	Frequency
1	2
2	2
3	2
4	2
5	5
6	11
7	5
8	7
9	1
10	3

Παράδειγμα εκτύπωσης ιστογράμματος

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<!-- Fig. 12.7: Histogram.html -->
<HEAD>
<TITLE>Histogram Printing Program</TITLE>

   <SCRIPT LANGUAGE = "JavaScript">
         function start() {
                  var theArray=[19,3,15,7,11,9,13,5,17,1];

                   document.writeln( "<TABLE BORDER = '1' WIDTH = '100%'>" );
                  document.writeln( "<TR><TD WIDTH = '100'><B>Element</B>" +
                           "<TD WIDTH = '100'><B>Value</B>" +      "<TD><B>Histogram</B></TR>" );

                  for ( var i in theArray ) {
                     document.writeln( "<TR><TD>" + i + "<TD>" + theArray[ i ] + "<TD>" );
                           // print a bar
                           for ( var j = 1; j <= theArray[ i ]; ++j )
                            document.writeln( "*" );

                           document.writeln( "</TR>" );
                  }
                   document.writeln( "</TABLE>" );
         }
</SCRIPT>
</HEAD><BODY ONLOAD = “start()”></BODY>
</HTML>
```

Η συνάρτηση δέχεται παραμέτρους με 2 τρόπους: 1)τιμή ή 2)αναφορά

Δύο τρόποι για να περάσετε τις παραμέτρους σε συναρτήσεις.

Κλήση με τιμή / Πέρασμα με τιμή

- Όταν χρησιμοποιείται για να περάσει το όρισμα, γίνεται αντιγραφή της τιμής και μεταβιβάζεται στην ονομαζόμενη συνάρτηση.
- Παίρνει περισσότερο χώρο, χρησιμοποιεί περισσότερη δύναμη, αλλά είναι πιο ασφαλές και εξαλείφει πολλά πιθανά προβλήματα.
- Χρησιμοποιείται στη JavaScript για αριθμούς ή λογικές τιμές (boolean) .

Κλήση με αναφορά / Πέρασμα με αναφορά

- Όταν χρησιμοποιείται για να περάσει όρισμα, η *θέση στη μνήμη /διεύθυνση* του ορίσματος περνάει για να καλέσει τη συνάρτηση.
- Παίρνει λιγότερο χώρο, καταναλώνει λιγότερη ενέργεια, αλλά είναι λιγότερο ασφαλές και επιτρέπει πολλά πιθανά προβλήματα.
- Χρησιμοποιείται στη JavaScript για αντικείμενα, συμπεριλαμβανομένων των πινάκων.

Συναρτήσεις και πίνακες

- Για να περάσετε έναν πίνακα σε μια συνάρτηση.
  - Αναφέρετε το όνομα του πίνακα χωρίς παρένθεση ως παράμετρο.
  - Δεν χρειάζεται να περάσετε ξεχωριστά το μέγεθος του πίνακα.

- Τα επιμέρους αριθμητικά και τα λογικά (boolean) στοιχεία πίνακα:

  - περνούν ακριβώς όπως οι απλές αριθμητικές και λογικές μεταβλητές: κλήση με τιμή
  - είναι απλά κομμάτια των δεδομένων που ονομάζονται scalars ή scalar qualities.
  - περνούν χρησιμοποιώντας το υπογεγραμμένο όνομα του στοιχείου πίνακα.

Ταξινόμηση πινάκων

- *Η ταξινόμηση* των δεδομένων αποτελεί ένα από τα σημαντικότερα σενάρια υπολογιστών.
  - Ουσιαστικά κάθε οργάνωση πρέπει να ταξινομεί τα δεδομένα σε κάποιο σύνολο.

- ***Bubble sort*** ή sinking sort τεχνική (ταξινόμιση της φυσαλλίδας ή ταξινόμηση καταβύθισης)
  - Μικρότερες τιμές αυξάνονται σταδιακά ή "αφρίζουν" το δρόμο τους προς τα πάνω.
  - Κάνει αρκετά περάσματα μέσα από τον πίνακα.
  - Σε κάθε πέρασμα διαδοχικά ζεύγη στοιχείων συγκρίνονται.
  - Εάν ένα ζευγάρι είναι σε αύξουσα σειρά, μένει όπως είναι.
  - Αν το ζεύγος είναι σε φθίνουσα σειρά, οι τιμές ανταλλάσσονται.

=> Εύκολο στον προγραμματισμό, αλλά τρέχει αργά.

- Δεν είναι καλό για την ταξινόμηση μεγάλων πινάκων.

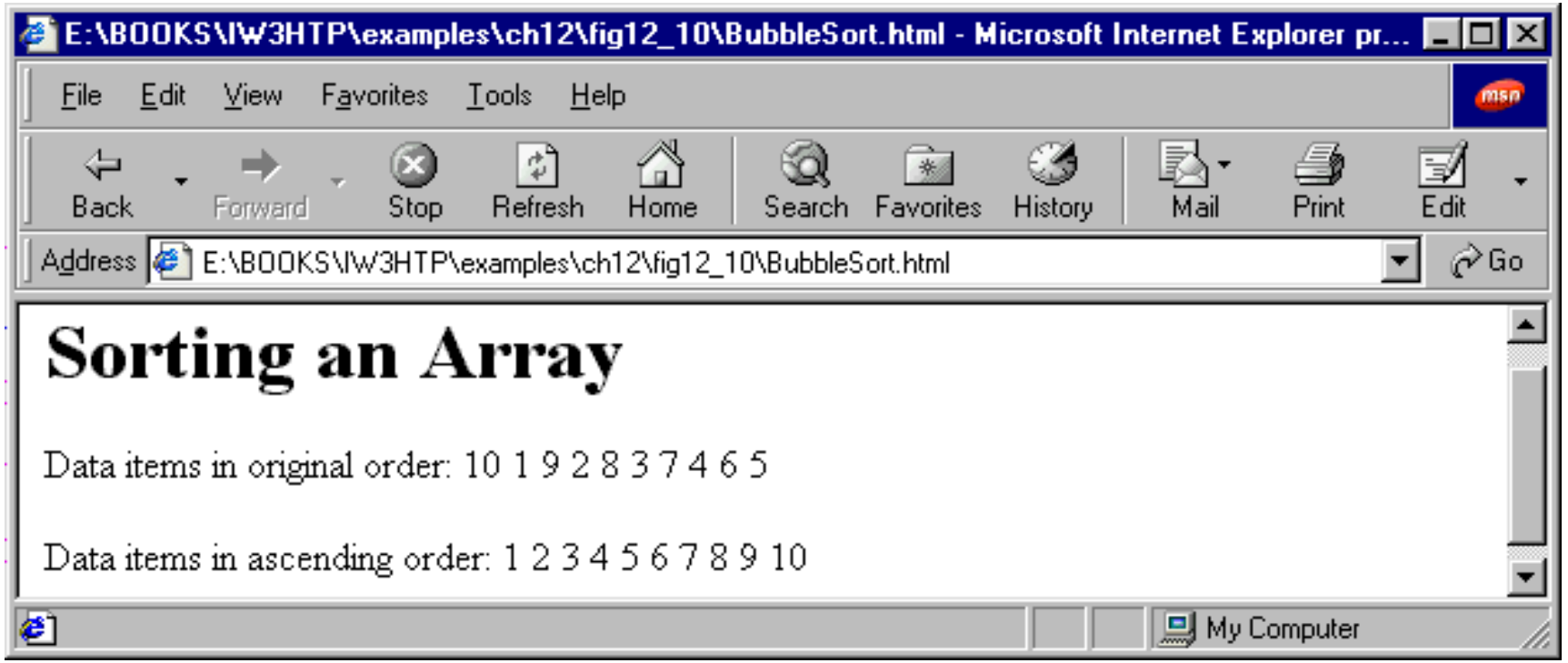
Πρόγραμμα ταξινόμησης με τον αλγόριθμο της φυσαλίδας

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
   <!-- Fig. 12.10: BubbleSort.html -->
   <HEAD>
      <TITLE>Sorting an Array with Bubble Sort</TITLE>
         <SCRIPT LANGUAGE = "JavaScript">
         function start() {
                  var a = [ 10, 1, 9, 2, 8, 3, 7, 4, 6, 5 ];
                  document.writeln( "<H1>Sorting an Array</H1>" );

                  outputArray( "Data items in original order: ", a );
                  bubbleSort( a );              // sort the array
                  outputArray( "Data items in ascending order: ", a );
         }
               // outputs "header" followed by the contents of "theArray"
               function outputArray( header, theArray ) {
                        document.writeln(
                        "<P>" + header + theArray.join( " " ) + "</P>" );
               }
                  // sort the elements of an array with bubble sort
                  function bubbleSort( theArray ) {
                           // control number of passes of theArray
                        for ( var pass = 1; pass < theArray.length; ++pass )
```

Πρόγραμμα ταξινόμησης με τον αλγόριθμο της φυσαλίδας

```
            // one pass - control's number of comparison per pass
            for ( var i = 0; i < theArray.length - 1; ++i )

            // perform one comparison
              if ( theArray[ i ] > theArray[ i + 1 ] )

                  swap( theArray, i, i + 1 );        // swap elements

}

// swap two elements of an array
function swap( theArray, first, second ) {
      var hold;   // temporary holding area for swap

      hold = theArray[ first ];

             theArray[ first ] = theArray[ second ];

             theArray[ second ] = hold;
}

</SCRIPT>
</HEAD><BODY ONLOAD = "start()"></BODY>
</HTML>
```

Έξοδος προγράμματος ταξινόμησης τυχαίου πίνακα



Αναζήτηση σε πίνακες: (α) γραμμική & (β) δυαδική

- Όταν εργάζεστε με μεγάλα ποσά δεδομένων

  - μπορεί να είναι απαραίτητο να καθοριστεί αν ένας πίνακας περιέχει μια τιμή που αντιστοιχεί σε μία συγκεκριμένη *τιμή κλειδί.*
  - *η διαδικασία αναζήτησης* εντοπίζει συγκεκριμένη τιμή σε ένα πίνακα.

- Δύο τεχνικές αναζήτησης
  - Γραμμική
  - Δυαδική

Αναζήτηση σε πίνακες: (α) γραμμική

Γραμμική αναζήτηση:

- Συγκρίνει κάθε στοιχείο σε ένα πίνακα με το κλειδί αναζήτησης.

- Πηγαίνει κατά σειρά στοιχείων σε πίνακα.
  - Αν ο πίνακας είναι αταξινόμητος, η πιθανότητα να βρεθεί στο πρώτο στοιχείο είναι ίδια με την πιθανότητα να βρεθεί στο τελευταίο στοιχείο.

- Δουλεύει καλά σε μικρούς ή αταξινόμητους πίνακες.

- Μη αποτελεσματική σε μεγάλους πίνακες.

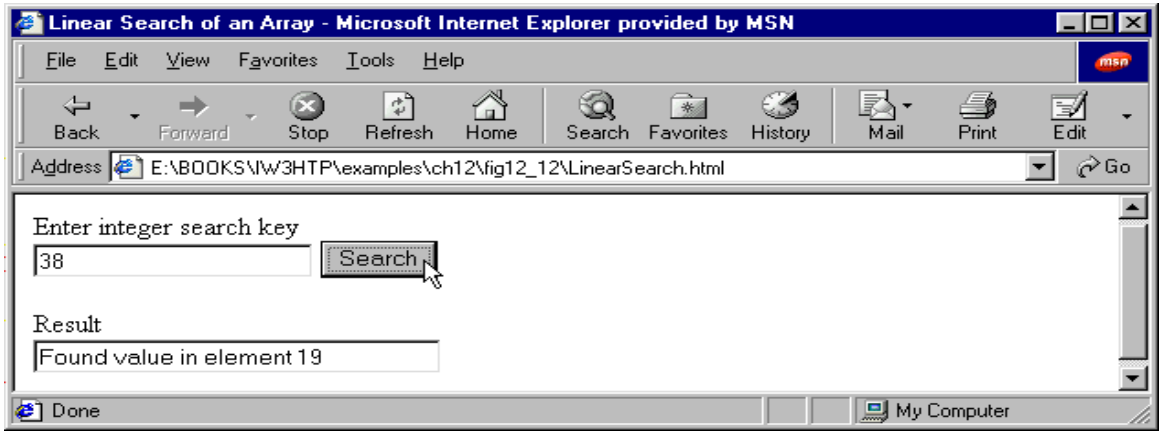
Πρόγραμμα γραμμικής αναζήτησης

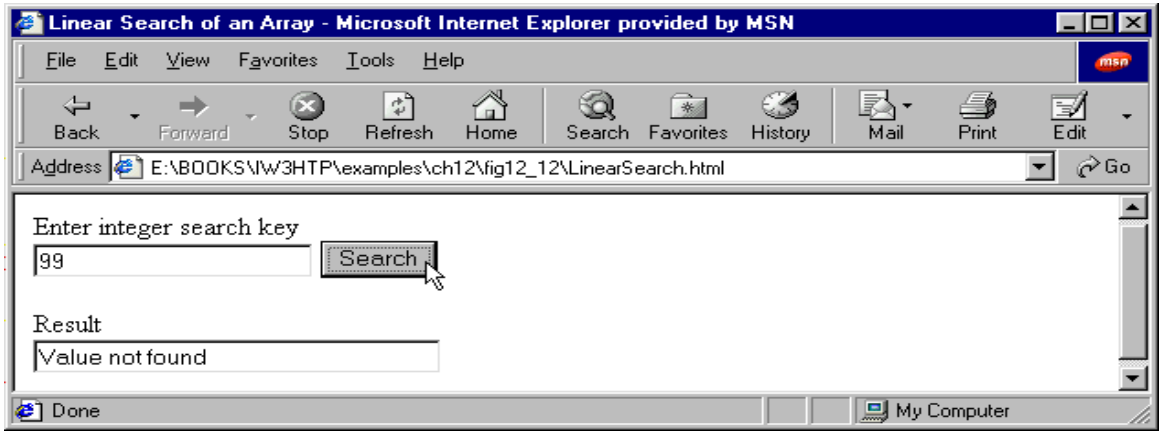
```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<!-- Fig. 12.12: LinearSerach.html -->
<HEAD>
<TITLE>Linear Search of an Array</TITLE>

   <SCRIPT LANGUAGE = "JavaScript">
          var a = new Array(100) ;
          //fill Array with even integer values from 0 to 198
          for ( var i=0; i<a.length; i++)
                    a[i] = 2*I;

          //function called when "Search" button is pressed
          function buttonPressed() {
                    var searchKe = searchForm.inputVal.value;
                    //Array a is passed to linearSearch even thought it is a global variable
                    var element = linearSearch( a, parseInt(searchKey) );

                    if (element != -1)
                              searchForm.result.value = "Found value in element" ;
                    else
                              searchForm.result.value = "Value not found" ;
          }
```

Πρόγραμμα γραμμικής αναζήτησης

```
//Search "theArray" for the specified "key" value
function linearSearch( theArray, key) {
        for (var n=0; n < theArray.length; n++)
                if(theArray[n] == key ) return n;

        return -1 ;
}

</SCRIPT>

</HEAD>
<BODY>
<FORM NAME= "searchForm" >
        <P> Enter integer search key <BR>
        <INPUT NAME = "inputVal" TYPE = "text" >
        <INPUT NAME = "search" TYPE = "button" VALUE = "Search"
                        ONCLICK="buttonPressed()"> <BR></P>

        <P> Result<BR>
        <INPUT NAME = "result" TYPE = "text" SIZE= "30" ></P>
</FORM>
</BODY>
</HTML>
```

Έξοδος προγράμματος γραμμικής αναζήτησης





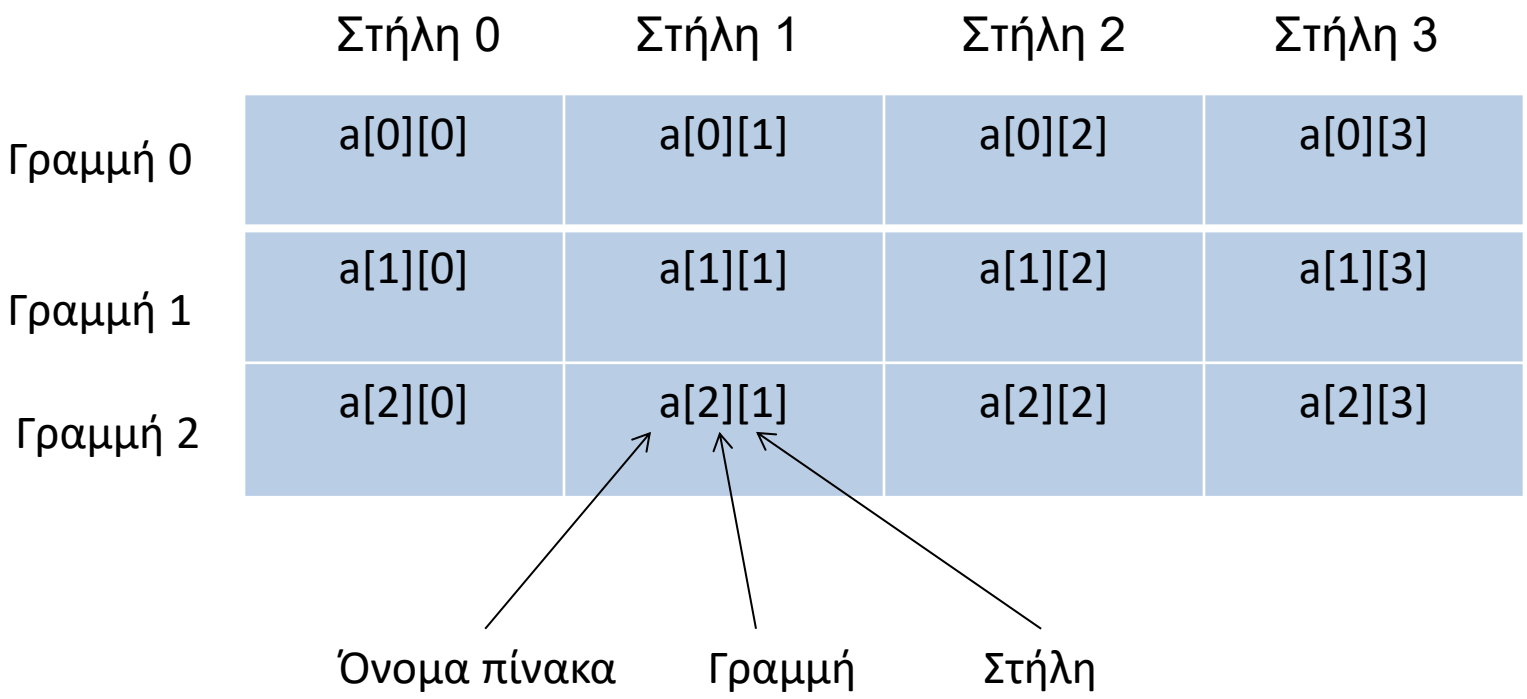
Αναζήτηση σε πίνακες: (β) δυαδική

Δυαδική αναζήτηση

- Εντοπίζει το μεσαίο στοιχείο.
- Συγκρίνει το κλειδί αναζήτησης με το μεσαίο στοιχείο.
- Εάν είναι ίσα, ο δείκτης του μεσαίου στοιχείου επιστρέφεται.
- Διαφορετικά, ο αλγόριθμος βλέπει εάν το κλειδί αναζήτησης είναι υψηλότερο ή χαμηλότερο από τη μεσαία τιμή.
- Αναζητά το πάνω ή το κάτω μισό του πίνακα (subarray) ανάλογα με τη θέση του κλειδιού αναζήτησης.
- Συνεχίζει την ίδια διαδικασία έως ότου το κλειδί αναζήτησης ισούται με τη μεσαία τιμή του subarray ή παραμένει ένα στοιχείο που δεν είναι ίσο με το κλειδί αναζήτησης.

Πίνακες πολλαπλών διαστάσεων

Δισδιάστατος πίνακας με τρεις γραμμές και τέσσερις στήλες



Πίνακες πολλαπλών διαστάσεων

- Αρχικοποίηση

  - Δηλώθηκε σαν ένας πίνακας ενιαίου σεναρίου.
  - Θα μπορούσε να δηλωθεί και να αρχικοποιηθεί με διπλό σενάριο με δύο σειρές και δύο στήλες.

```
var b = [[1, 2], [3, 4, 5]]
```

- Ο μεταγλωττιστής καθορίζει τον αριθμό των στοιχείων σε σειρά / στήλη.

  - Με τον υπολογισμό του αριθμού των τιμών αρχικοποίησης στη λίστα υπο-αρχικοποιητή για αυτήν τη σειρά / στήλη.

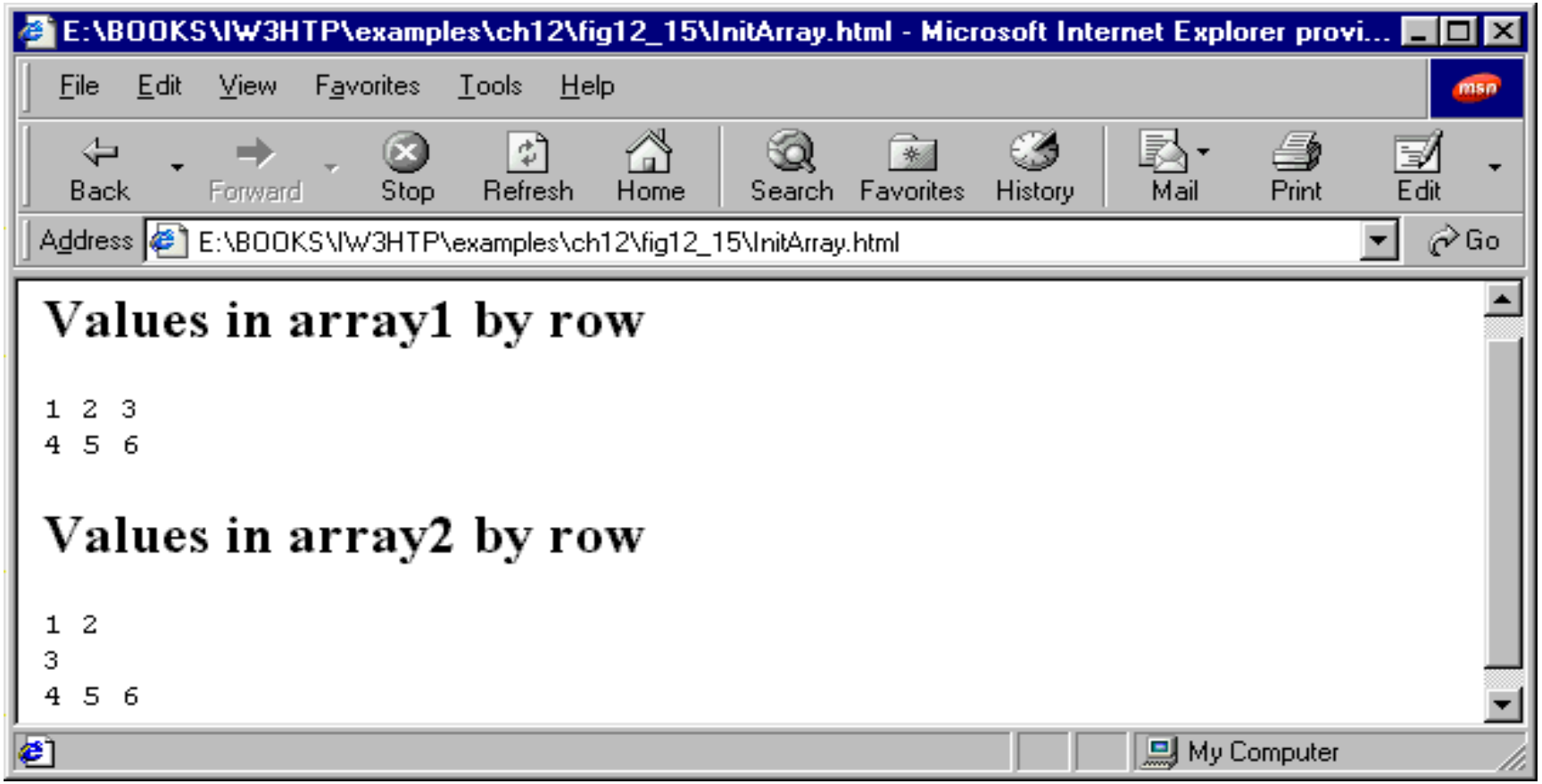
- Μπορεί να έχει διαφορετικό αριθμό στηλών σε κάθε γραμμή των βρόχων for και for / in που χρησιμοποιούνται για τη διάσχιση των πινάκων.

  - Χειριστείτε κάθε στοιχείο του πίνακα

Πρόγραμμα αρχικοποίησης πολυδιάστατου πίνακα

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0
Transitional//EN">
<HTML>
<!-- Fig. 12.15: InitArray.html -->
<HEAD>
<TITLE>Initializing Multidimensional Arrays</TITLE>

  <SCRIPT LANGUAGE = "JavaScript">
      function start() {
            var array1 = [ [1,2,3] , [4,5,6] ] ;
            var array2 = [ [1,2] , [3] , [4,5,6] ];

            outputArray("Values in array1 by row", array);
            outputArray("Values in array2 by row" , array2) ;
```

Έξοδος προγράμματος αρχικοποίησης πολυδιάστατου πίνακα



Πίνακες πολλαπλών διαστάσεων

Διαφορετικοί χειρισμοί πινάκων χρησιμοποιώντας **`for`** και **`for/in`**:

```
for ( int col = 0; col < a[ 2 ].length; ++col )
a[ 2 ][ col ] = 0;
```

ίδιο με

```
for ( var col in a[ 2 ] )
a[ 2 ][ col ] = 0
```

- Και οι δύο δηλώσεις θέτουν όλα τα στοιχεία στην τρίτη σειρά του πίνακα a στο μηδέν.

Πίνακες πολλαπλών διαστάσεων

Διαφορετικοί χειρισμοί πινάκων χρησιμοποιώντας **for** και **for/in**:

```
var total = 0;
for (var row = 0; row < a.length; ++row )
   for (var col = 0; col < a[ row ].length; ++col )
        total += a[ row ][ col ];
```

ίδιο με

```
var total = 0;
for (var row in a )
   for (var col in a[ row ] )
        total += a[ row ][ col ];
```

- Και οι δύο δηλώσεις θέτουν όλα τα στοιχεία του πίνακα, μία σειρά κάθε φορά.

Πρόγραμμα δισδιάστατου πίνακα

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<!-- Fig. 12.16: DoubleArray.html -->
<HEAD>
<TITLE>Double-subscripted Array Example</TITLE>
<SCRIPT LANGUAGE = "JavaScript">
function start() {

   var grades = [ [ 77, 68, 86, 73 ], [ 96, 87, 89, 81 ], [ 70, 90, 86, 81 ] ];

   document.writeln( "<PRE>" );
   outputArray( grades );

   document.writeln(
      "\n\nLowest grade: " + minimum( grades ) +
      "\nHighest grade: " + maximum( grades ) );

   // calculate average for each student (i.e., each row)
   for ( var i in grades )
      document.write( "\nAverage for student " + i +
         " is " + average( grades[ i ] ) );

   document.writeln( "</PRE>" );
}
```

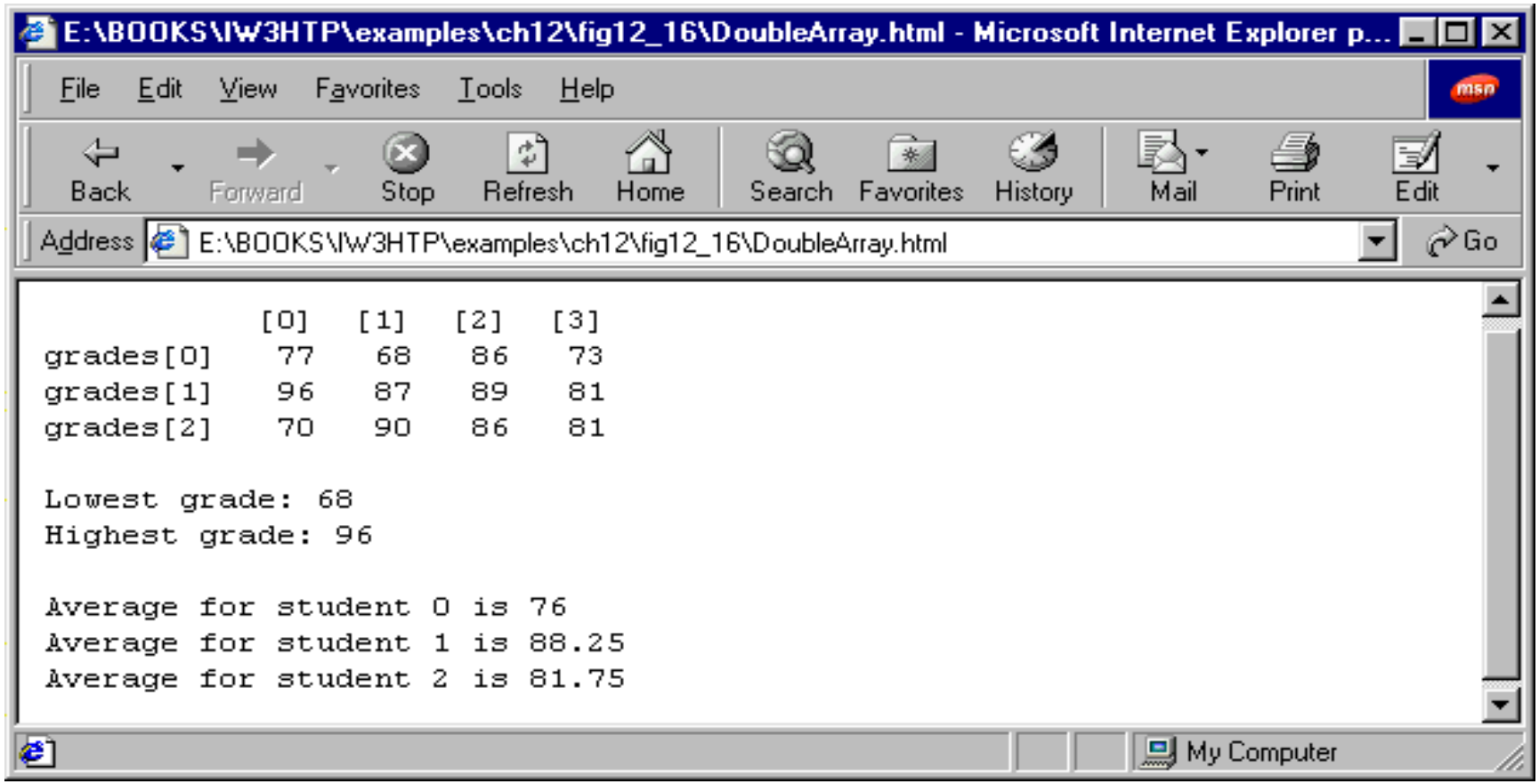
Πρόγραμμα δισδιάστατου πίνακα

```
31      function minimum( grades )
32      {
33          var lowGrade = 100;
34
35          for ( var i in grades )
36              for ( var j in grades[ i ] )
37                  if ( grades[ i ][ j ] < lowGrade )
38                      lowGrade = grades[ i ][ j ];
39
40          return lowGrade;
41      }
42
43      // find the maximum grade
44      function maximum( grades )
45      {
46          var highGrade = 0;
47
48          for ( var i in grades )
49              for ( var j in grades[ i ] )
50                  if ( grades[ i ][ j ] > highGrade )
51                      highGrade = grades[ i ][ j ];
52          return highGrade;
        }
```

Πρόγραμμα δισδιάστατου πίνακα

```
function average( setOfGrades ) {
        var total = 0 ;

        for (var i in setOfGrades )
                total += setOfGrades[i] ;

        return total / setOfGrades.length ;
}

function outputArray( grades ) {
        document.write( "          " ) ; //align heads

        for ( var i in grades[0])
                document.write( "[" +i+ "]" ) ;

        for (var i in grades) {
                document.write( "<BR>grades["+i+"] " ) ;

                for ( var  j in grades[i])
                        document.write( grades[i][j] + "     " ) ;
        }
}

</SCRIPT>
</HEAD><BODY ONLOAD = "start()" ></BODY>
</HTML>
```

Έξοδος προγράμματος δισδιάστατου πίνακα



Δηλώσεις Ελέγχου Ι

- Εισαγωγή
- Αλγόριθμοι
- Ψευδοκώδικας
- Δομές ελέγχου
- Η δομή επιλογής if
- Η δομή επιλογής if / else
- Η δομή επανάληψης while
- Αλγόριθμοι διαμόρφωσης: Μελέτη περίπτωσης 1 (Αντι-ελεγχόμενη επανάληψη)
- Σχεδίαση αλγορίθμων με την κορυφή προς τα κάτω, Βαθμονόμηση σταδιακά:
- Μελέτη περίπτωσης 2 (Επανάληψη Ελέγχου με Φρουρό)
- Σχεδίαση αλγορίθμων με την κορυφή προς τα κάτω, Βαθμονόμηση σταδιακά:
- Μελέτη περίπτωσης 3 (δομές ελέγχου)
- Εκχωρητές
- Διαχειριστές αύξησης και εκτροπής
- Σημείωση σχετικά με τον τύπο δεδομένων

Αλγοριθμικά στοιχεία της javascript

Σε αυτό το μάθημα:

- Θα κατανοήσετε τις βασικές τεχνικές επίλυσης προβλημάτων.
- Θα είστε σε θέση να αναπτύξετε αλγορίθμους μέσω της διαδικασίας από πάνω προς τα κάτω, σταδιακή βελτίωση.
- Για να μπορέσετε να χρησιμοποιήσετε τις δομές επιλογής if και if ... else για να επιλέξετε ανάμεσα σε εναλλακτικές ενέργειες.
- Θα χρησιμοποιήσετε την εντολή επανάληψης ενώ εκτελείτε δηλώσεις σε μια δέσμη ενεργειών επανειλημμένα.
- Θα κατανοήσετε την αντι-ελεγχόμενη επανάληψη και την επανάληψη του ελέγχου με φρουρό.
- Θα μπορέσετε να χρησιμοποιήσετε τους χειριστές αύξησης, μείωσης και εκχώρησης.

Εισαγωγή

- Πριν από τον προγραμματισμό ενός σεναρίου έχετε μία
  - απόλυτη κατανόηση του προβλήματος.
  - προσεκτικά σχεδιασμένη προσέγγιση για την επίλυσή του.

- Κατά τη σύνταξη ενός σεναρίου, σημαντικό είναι να
  - κατανοήστε τους τύπους δομικών στοιχείων και διαθέσιμων εργαλείων.
  - χρησιμοποιήστε αποδεδειγμένες αρχές κατασκευής του προγράμματος.

Αλγόριθμοι

- Αλγόριθμος: Διαδικασία επίλυσης προβλήματος

1. Ενέργειες για να είναι εκτελέσιμος.
2. Τάξη στην οποία εκτελούνται οι ενέργειες.

- Η σειρά των στοιχείων του αλγορίθμου είναι πολύ σημαντική.

  - Ακόμη και αν η σειρά εμφανίζεται ασήμαντη, τα σφάλματα μπορούν να έχουν εκτεταμένα αποτελέσματα.

Ψευδοκώδικας

Ψευδοκώδικας

- Τεχνητή και ανεπίσημη γλώσσα παρόμοια με την καθημερινή αγγλική γλώσσα.

- Βοηθάει τους προγραμματιστές να αναπτύξουν αλγορίθμους.

Αναγκάζει τον προγραμματιστή να "σκεφτεί" τον αλγόριθμο πριν από τη σύνθεση.

- Δεν είναι πραγματική γλώσσα προγραμματισμού υπολογιστών.

- Μετατρέπονται εύκολα σε JavaScript.

- Περιγράφει μόνο εκτελέσιμες δηλώσεις.

Δομές ελέγχου

- Διαδοχική εκτέλεση

    - Εκτέλεση δηλώσεων το ένα μετά το άλλο γραπτώς.
    - Ο κανονικός προγραμματισμός χρησιμοποιεί διαδοχική εκτέλεση.
    - Διάφορες εντολές JavaScript επιτρέπουν την εκτέλεση εντολών εκτός σειράς.
        - Μεταφορά ελέγχου

- Ο προγραμματισμός της δεκαετίας του 1960 χρησιμοποίησε τη δήλωση goto.

    - Δομημένος προγραμματισμός.
    - Ρίζα των περισσότερων δυσκολιών προγραμματισμού στη δεκαετία του '60.
    - Δεν υπάρχει στη JavaScript.

Δομές ελέγχου ΙΙ

Έρευνα του Bohm και του Jacopini

- Όλα τα προγράμματα μπορούν να γραφτούν από τρεις δομές ελέγχου.

1. Δομή αλληλουχίας
   - Κατασκευάστηκε σε JavaScript.
   - Μέθοδος προεπιλογής.

2. Δομή επιλογής

3. Δομή επανάληψης

Δομές ελέγχου III

ΔΙΑΓΡΑΜΜΑ ΡΟΗΣ

- Γραφική απεικόνιση αλγορίθμου ή τμήματος αλγορίθμου.

- Χρησιμοποιεί σύμβολα για να υποδείξει τύπους αποφάσεων των ενεργειών.

    - Σύμβολα που συνδέονται με γραμμές ροής.
        - Ορθογώνιο: οποιαδήποτε ενέργεια
        - Οβάλ: αρχή / τέλος αλγορίθμου
        - Ρόμβος: απόφαση

Δομές ελέγχου IV

3 Τύποι δομών επιλογής

- if
  - Δομή μονής επιλογής.
    - Επιλέγει ή αγνοεί μια μεμονωμένη ενέργεια ή μια ομάδα ενεργειών.
- if / else
  - Δομή διπλής επιλογής
    - Επιλέγει μεταξύ δύο ενεργειών ή ομάδων ενεργειών.
- Switch
  - Δομή πολλαπλών επιλογών
    - Επιλέγει μεταξύ πολλών ενεργειών ή ομάδων ενεργειών.

Δομές ελέγχου V

- 4 τύποι δομών επανάληψης
  - *while*
  - *do/while*
  - *for*
  - *for/in*
- Δύο τρόποι να συνδυάζεις δομές
  - *Δομή ελέγχου στοίβας (stacking)*
    - *Δομές μονής εισόδου και εξόδου*
  - *Εμφωλευμένη δομή ελέγχου*

Δομές VI

Όλα τα ονόματα δομών ελέγχου είναι λέξεις-κλειδιά.

Δεσμευμένο από τη γλώσσα για την υλοποίηση των λειτουργιών.

Δεν επιτρέπεται να χρησιμοποιούνται ως ονόματα μεταβλητών.

Λέξεις κλειδιά που χρησιμοποιούνται στη Javascript				
break	case	catch	continue	default
delete	do	else	finally	for
function	if	In	instanceof	new
return	switch	this	throw	try
typeof	var	void	while	with

Δομές VI

Λέξεις κλειδιά δεσμευμένες αλλά δεν χρησιμοποιούνται αυτή τη στιγμή στη Javascript				
abstract	boolean	byte	char , int	class
const	debugger	double	enum	export
extends	final	float	goto	implements
import	volatile	inteface	long	native
package	private	synchronized	throws	trasient

Η δομή επιλογής if

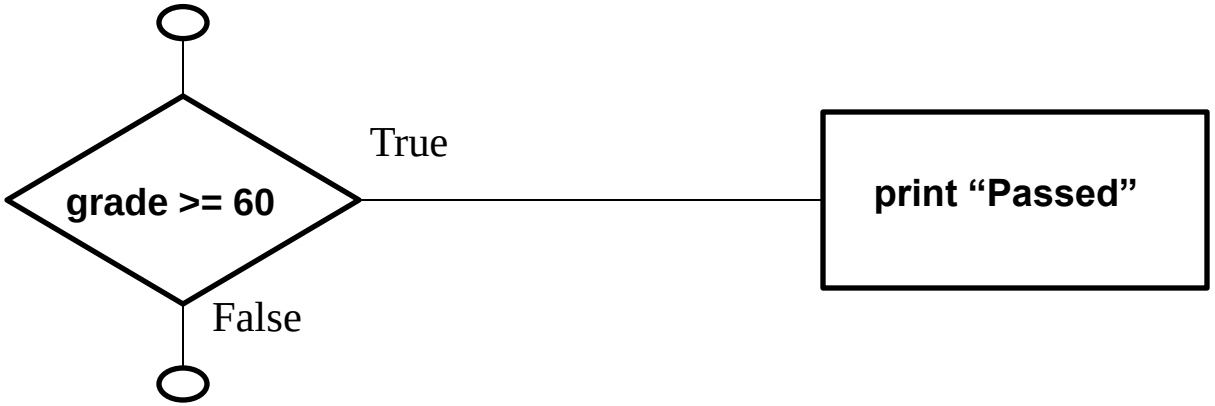
Ψευδοκώδικας:

```
If student's grade is greater than or equal to 60
        Print "Passed"
```

Δήλωση Javascript:

```
if (grade >= 60)
        document.writeln("Passed") ;
```

ΔΙΑΓΡΑΜΜΑ ΡΟΗΣ



Η δομή επιλογής if II

- Οι συνθήκες που αξιολογούνται ως true.
  - Αληθής συνθήκη.
  - Μη μηδενική αριθμητική τιμή.
  - Συμβολοσειρά που περιέχει τουλάχιστον ένα χαρακτήρα.

- Οι συνθήκες που αξιολογούνται ως false.
  - Ψευδής συνθήκη.
  - Αριθμητική τιμή ίση με 0.
  - Άδεια συμβολοσειρά.
  - Μεταβλητή χωρίς καθορισμένη τιμή.

Η δομή επιλογής if/else

- Ψευδοκώδικας:

If student's grade is greater than or equal to 60

Print "Passed"

else

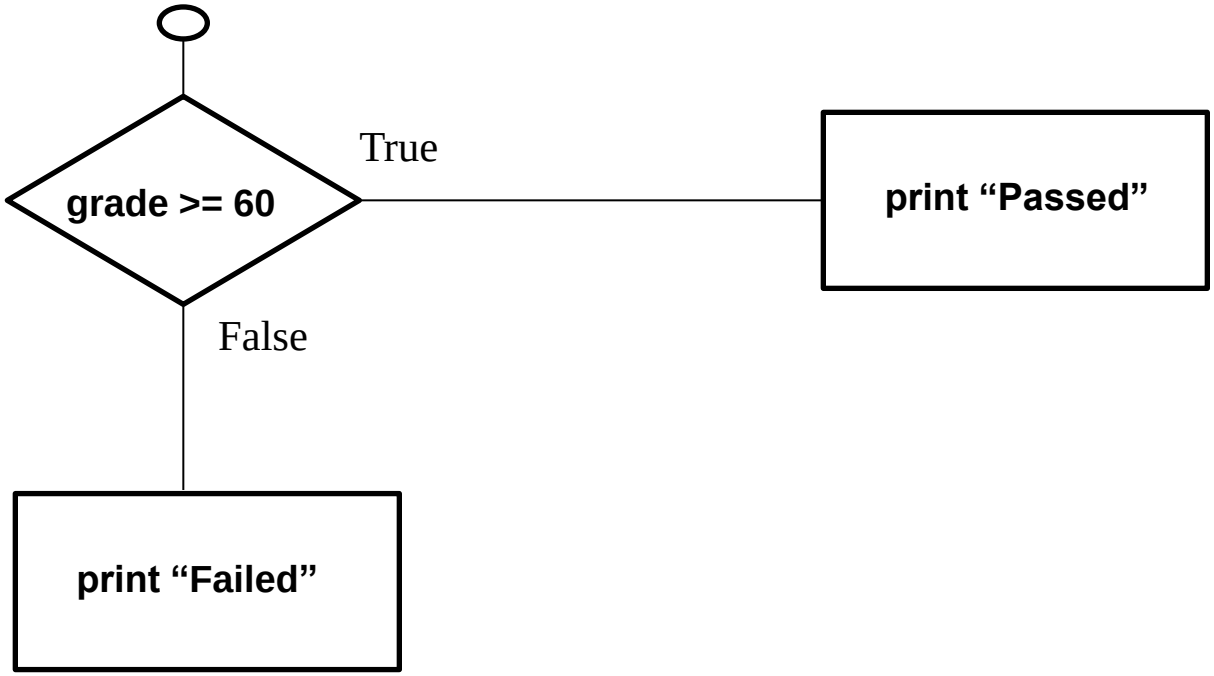
Print "Failed"

- Δήλωση JavaScript:

```
if ( grade >= 60 )
   document.writeln( "Passed" );
else
   document.writeln( "Failed" );
```

Η δομή επιλογής if/else II

ΔΙΑΓΡΑΜΜΑ ΡΟΗΣ



Η δομή επιλογής if/else III

- Υποθετικός χειριστής (? :)

  - Μοναδικός τριμερής χειριστής της JavaScript.
  - Λαμβάνει τρεις τελεστές.

1. Λογική έκφραση (Boolean).
2. Τιμή για υποθετική έκφραση αν είναι αληθής.
3. Τιμή για υποθετική έκφραση αν είναι ψευδής.

Παράδειγμα

```
document.writeln ( studentGrade> = 60; “Passed": “Failed");
```

Ίδια λειτουργία με την προηγούμενη δήλωση if / else.

Η δομή επιλογής if/else IV

Εμφωλευμένη δομή if/else:

- Ψευδοκώδικας:

```
if student's grade is greater than or equal to 90
     Print "A"
else
     if Student's grade is greater than or equal to 80
           Print "B"
     else
         if student's grade is greater than or equal to 70
               Print "C"
         else
              if student's grade is greater than or equal to 60
                    Print "D"
              else
                    Print "F"
```

Η δομή επιλογής if/else V

Εμφωλευμένη δομή if/else II:

- Ψευδοκώδικας:

```
if ( studentGrade >= 90 )
 document.writeln( "A" );
else
    if ( studentGrade >= 80 )
       document.writeln( "B" );
    else
       if ( studentGrade >= 70 )
          document.writeln( "C" );
       else
          if ( studentGrade >= 60 )
             document.writeln( "D" );
          else
             document.writeln( "F" );
```

Η δομή επιλογής if/else VI

Εμφωλευμένη δομή if/else III:

- Παρόμοια δήλωση Javascript:

```
if ( studentGrade >= 90 )
   document.writeln( "A" );
else if ( studentGrade >= 80 )
   document.writeln( "B" );
else if ( studentGrade >= 70 )
   document.writeln( "C" );
else if ( studentGrade >= 60 )
   document.writeln( "D" );
else
   document.writeln( "F" );
```

Η δομή επιλογής if/else VII

- Διερμηνέας Javascript

  - Συνδέει τη δήλωση else με προηγούμενη δήλωση if εκτός αν περιβάλλεται από άγκιστρα ({}).

Παράδειγμα:

```
if ( x > 5 )

if ( y > 5 )

   document.writeln( “x and y are > 5” );

else

document.writeln( “x is <= 5” );
```

Η δομή επιλογής if/else VIII

Λόγω της εσοχής, φαίνεται ότι η δήλωση else ισχύει για τη δεύτερη εντολή if.
Ο διερμηνέας της JavaScript διαβάζει ως εξής:

```
   if ( x > 5 )
if ( y > 5 )
   document.writeln( “x and y are > 5” );
    else
 document.writeln( “x is <= 5” );
```

Η δομή επιλογής if/else IX

Για να διαβάσει ο διερμηνέας του JavaScript τη δομή όπως εσείς σκοπεύατε, χρησιμοποιήστε άγκιστρα ({}).

```
if ( x > 5 ) {
      if ( y > 5 )
            document.writeln( “x and y are > 5” );
}
else
      document.writeln( “x is <= 5” );
```

Η δήλωση else τώρα εφαρμόζεται στην πρώτη δήλωση if.

Η δομή επιλογής if/else Χ

Σύνθεση Δήλωση:

- Δήλωση που περιέχεται μέσα σε άγκιστρα ({}).
- Δεν τελειώνει με ένα ελληνικό ερωτηματικό.
  - Όλες οι δηλώσεις μέσα πρέπει να τελειώνουν με ελληνικά ερωτηματικά.

Παράδειγμα:

```
if ( grade >= 60 )
                document.writeln( "Passed" );
else {
                document.writeln( "Failed<BR>" );
                document.writeln( "You must take the course again." );
}
```

- Ο διερμηνέας του JavaScript εκτελεί και τις δύο εντολές writeln μέσα στα άγκιστρα αν η συνθήκη if είναι ψευδής.
- Χωρίς άγκιστρα, η τελευταία εντολή writeln έξω από την if / else δομή θα εκτελείται πάντα.

Η δομή επανάληψης while

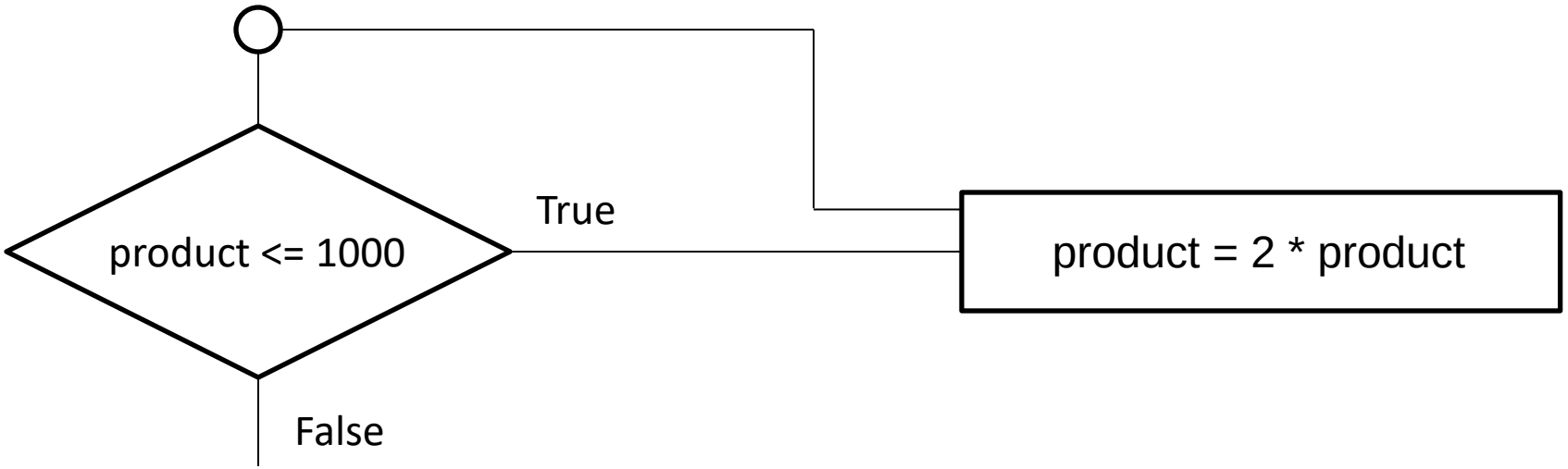
- Ψευδοκώδικας:

While product is less than 1000
    multiply product by 2

- Javascript:

```
var product = 2 ;
While ( product <= 1000)
    product = 2 * product ;
```

- Διάγραμμα ροής:



Αλγόριθμοι διαμόρφωσης: Μελέτη περίπτωσης 1 (Counter-Controlled Repetition)

Counter-Controlled Repetition:

Χρησιμοποιεί δομή επανάληψης

- Ελέγχει εάν η μεταβλητή counter έχει φτάσει την τιμή στόχο χρησιμοποιώντας σχετικές συνθήκες.
- Η μεταβλητή counter αυξάνεται ή μειώνεται ένα καθορισμένο ποσό σε κάθε βρόχο.
- Η δομή τελειώνει όταν η κατάσταση γίνεται ψευδής (δηλαδή: ο μετρητής φτάνει στην τιμή στόχο).

Χρησιμοποιείται

- Με ή χωρίς είσοδο χρήστη.
- Όταν υπάρχει ένας γνωστός αριθμός βρόχων.

Πρόγραμμα υπολογισμού μέσου όρου

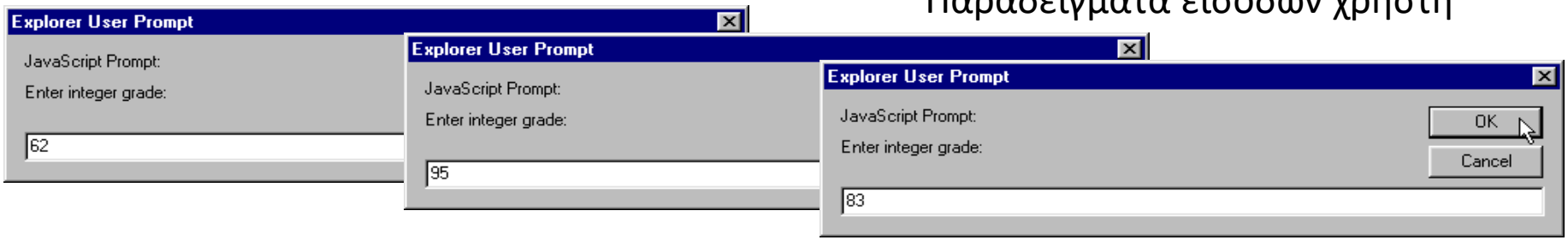
```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<!-- Fig. 9.7: average.html -->
<HEAD>
<TITLE>Class Average Program</TITLE>

   <SCRIPT LANGUAGE = "JavaScript">
          var total, gradeCounter, gradeValue, average, grade;
          //Initialization Phase
          total = 0 ;
          gradeCounter = 1 ;

          //Processing Phase
          while( gradeCounter <= 10) {
                    grade = window.prompt("Enter integer grade:", "0");
                    gradeValue = parseInt( grade );
                    total = total + gradeValue;
                    gradeCounter = gradeCounter + 1;
          }
```

Πρόγραμμα υπολογισμού μέσου όρου

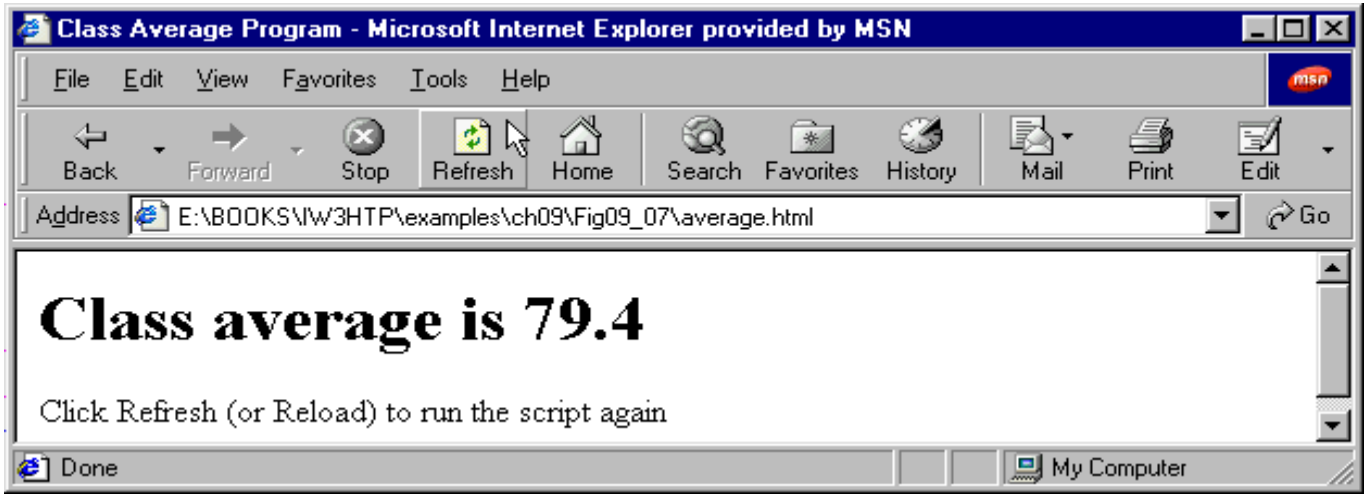
```
        //Termination Phase
        average = total / 10 ; //calculte the average

        document.writeln("<H1>Class average is " +average+ "<H1>" ) ;

</SCRIPT>

</HEAD>
<BODY>
Click Refresh ( or Reload) to run the script again
</BODY>
</HTML>
```

Έξοδος προγράμματος μέσου όρου

Παραδείγματα εισόδων χρήστη



Έξοδος προγράμματος με είσοδο χρήστη: 100, 88, 55, 68, 77, 83, 95, 73, 62



Αλγόριθμοι διαμόρφωσης: με Top-Down, Σταδιακή Βελτίωση: Μελέτη περίπτωσης 2

Επανάληψη ελεγχόμενη από φρουρό:

- Χρησιμοποιεί δομή επανάληψης
  - Ελέγχει εάν η μεταβλητή counter έχει οριστεί σε τιμή φρουρού, χρησιμοποιώντας όρους ισότητας.
  - Όταν ο χρήστης εισάγει συμβολοσειρά ίση με την τιμή του φρουρού, η κατάσταση θα είναι ψευδής.

- Χρησιμοποιείται όταν
  - Η είσοδος του χρήστη ενσωματώνεται στη δομή.
  - Τελικός αριθμός βρόχων άγνωστος - αόριστη επανάληψη.

- Η πρώτη είσοδος χρήστη θα πρέπει να πραγματοποιηθεί πριν από την έναρξη της δομής.
  - Βεβαιωθείτε ότι έχετε υπόψη την πιθανότητα ο χρήστης να εισάγει αρχικά την τιμή του φρουρού.

- Η τιμή φρουρού που επιλέγεται έτσι ώστε να μην συγχέεται με αποδεκτή τιμή εισόδου.
  → -1 είναι μια κοινή τιμή φρουρού.

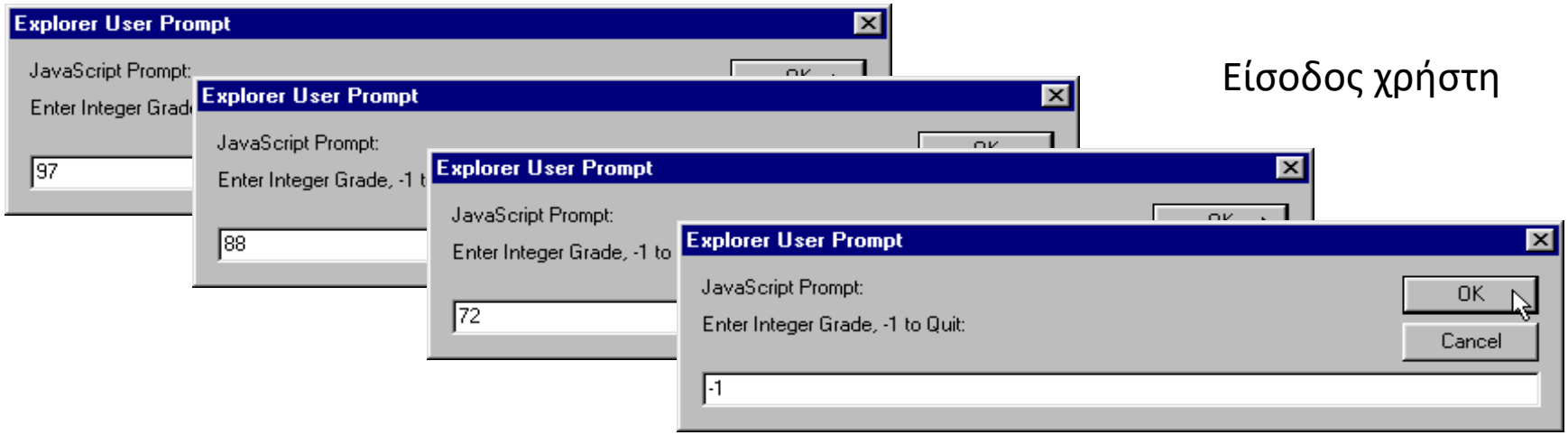
Πρόγραμμα υπολογισμού μέσου όρου II

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<!-- Fig. 9.9: average2.html -->
<HEAD>
<TITLE>Class Average Program: Sentinel-controlled Repetition</TITLE>

  <SCRIPT LANGUAGE = "JavaScript">
        var total, gradeCounter, gradeValue, average, grade;
        //Initialization Phase
        total = 0 ;
        gradeCounter = 0 ;
        //Processing Phase prompt for input and read
        grade = window.prompt("Enter Integer Grade, -1 to Quit.", "0");
        gradeValue = parseInt(grade) ;
        while( gradeValue != -1) {
                total = total + gradeValue;
        gradeCounter = gradeCounter + 1;
                grade = window.prompt("Enter integer grade, -1 to Quit", "0");
                gradeValue = parseInt( grade );
        }
```

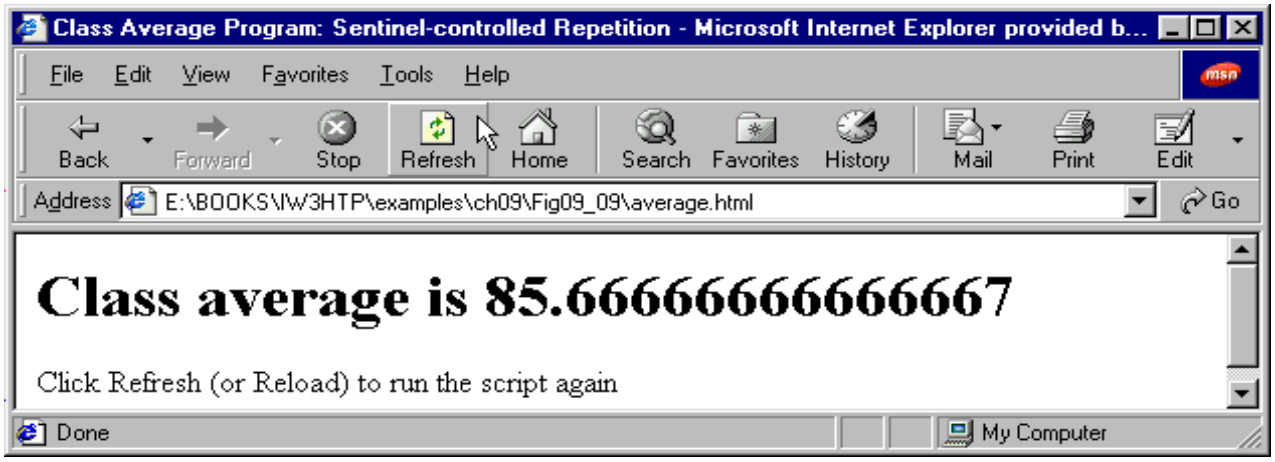
Πρόγραμμα υπολογισμού μέσου όρου II

```
        if (gradeCounter != 0) {
                average = total / gradeCounter ;

                //display average of exam grades
                document.writeln("<H1>Class average is "+average+"</H1>") ;
        else
                document.writeln("<P>No grades were entered</P>");
</SCRIPT>

</HEAD>
<BODY>
<P> Click Refresh ( or Reload) to run the script again </P>
</BODY>
</HTML>
```

Έξοδος προγράμματος μέσου όρου II



Είσοδος χρήστη



Έξοδος προγράμματος

Αλγόριθμοι διαμόρφωσης: με Top-Down, Σταδιακή Βελτίωση: Μελέτη περίπτωσης 3

- Ένθετες δομές ελέγχου
  - Οι δομές ελέγχου μπορούν να τοποθετηθούν μέσα σε άλλες δομές ελέγχου.
  - Μπορεί να εκτελεστεί όσες φορές χρειάζεται.
  - Μπορεί να επιτύχει τους στόχους του προγράμματος γρηγορότερα και με λιγότερες επιπλοκές.
  - Να είστε βέβαιος για να
    - Χαρτογραφήστε τον αλγόριθμό σας με ψευδοκώδικα ή / και διάγραμμα ροής πριν από τον προγραμματισμό.
    - Εισάγετε σχόλια στο πρόγραμμα για να βοηθήσετε την αποσφαλμάτωση.

- Αρχικοποίηση μεταβλητής
  - Οι τιμές μπορούν να αντιστοιχιστούν σε μεταβλητές στη δήλωση αρχικοποίησης.
  - Αν η μεταβλητή δεν εισαχθεί, θα αρχικοποιηθεί αυτόματα από τη JavaScript.

Πρόγραμμα ανάλυσης βαθμών φοιτητή

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<!-- Fig. 9.11: analysis.html -->
<HEAD>
<TITLE>Analysis of Examination Results</TITLE>

   <SCRIPT LANGUAGE = "JavaScript">
          var passes = 0, failures = 0, student = 1, result ;

          //process 10 students; counter-controlled loop
          while( student <= 10) {
                    result = window.prompt("Enter result(1=pass 2=failure)");
                    if ( result == "1")
                              passes= passes + 1;
                    else
                              failures = failures + 1;

                    student = student + 1 ;
          }
```

Πρόγραμμα ανάλυσης βαθμών φοιτητή

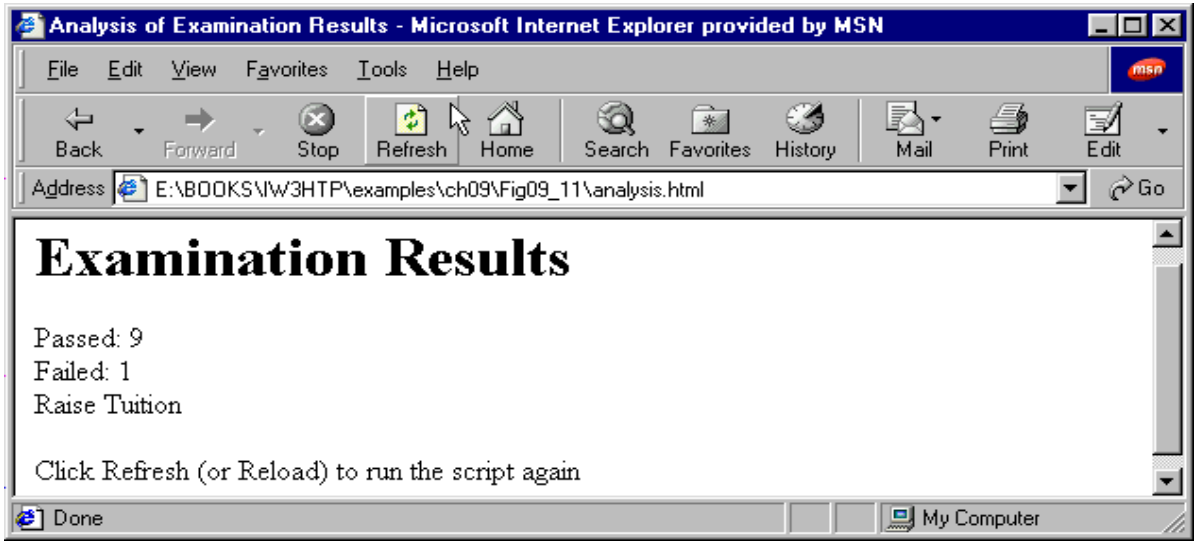
```
            //termination phase
            document.writeln("<H1>Examination Results</H1>");
            document.writeln("Passes: " +passes+ "<BR>Failed: "+failures ) ;

            if (passes > 8 )
                        document.writeln("<BR>Raise Tuition" ) ;

</SCRIPT>

</HEAD>
<BODY>
<P> Click Refresh ( or Reload) to run the script again </P>
</BODY>
</HTML>
```

Πρώτη εκτέλεση προγράμματος ανάλυσης βαθμών φοιτητή

Είσοδος χρήστη:

Εισάγει τη συμβολοσειρά «1» εννιά φορές.
Εισάγει τη συμβολοσειρά «2» μία φορά.

Έξοδος προγράμματος:

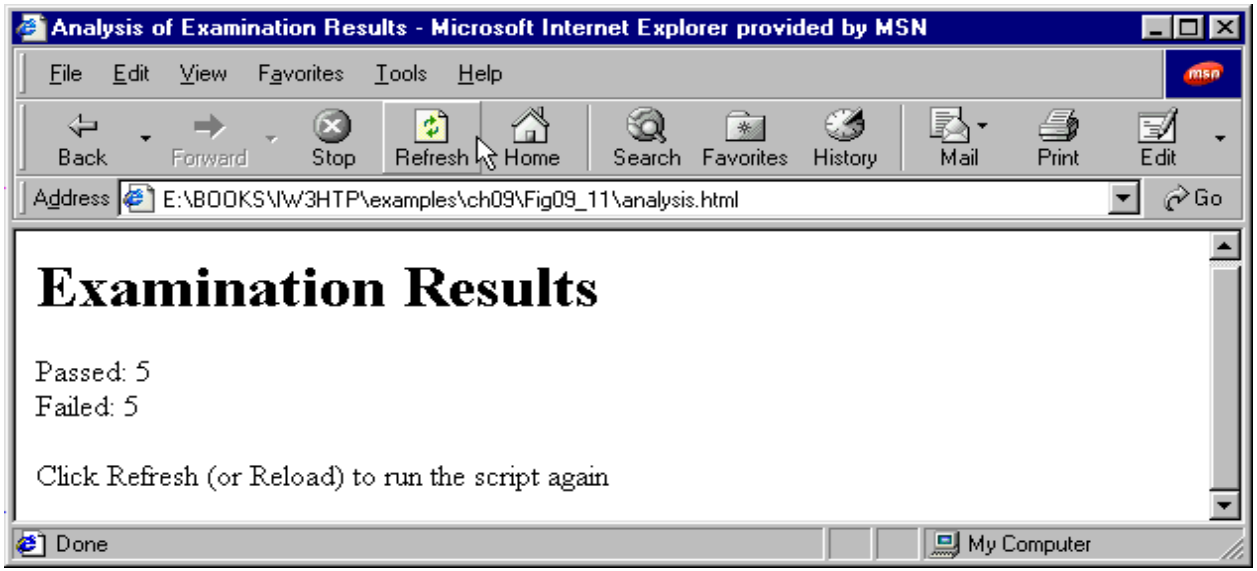


Δεύτερη εκτέλεση προγράμματος ανάλυσης βαθμών φοιτητή

Είσοδος χρήστη:

Εισάγει τη συμβολοσειρά «1» πέντε φορές.
Εισάγει τη συμβολοσειρά «2» πέντε φορές.

Έξοδος προγράμματος:



Εκχωρητές

- Οι εργασίες εκχώρησης με τα ίδια αποτελέσματα μπορούν να γραφτούν με διαφορετικούς τρόπους.

Παράδειγμα 1:

```
c = c +3 ;
```

Παράδειγμα 2:

```
c += 3 ;
```

- Και οι δύο τρόποι προσθέτουν τον αριθμό 3 στην τιμή της μεταβλητής c.
- Το παράδειγμα 2 εκτελείται γρηγορότερα.
  - Μικρή διαφορά για μεμονωμένες λειτουργίες.
  - Σημαντική σε μεγάλο αριθμό λειτουργιών.

Εκχωρητές II

Αριθμητικοί εκχωρητές

Εκχωρητής	Αρχική τιμή μεταβλητής	Παράδειγμα έκφρασης	Εξήγηση	Εκχωρεί
+=	c = 3	c += 7	c = c + 7	10 στη c
-=	d = 5	d -= 4	d = d -4	1 στη d
*=	e = 4	e *= 5	e = e *5	20 στην e
/=	f = 6	f /= 3	f = f / 3	2 στην f
%=	g = 12	g %= 9	g = g % 9	3 στη g

Τελεστές αύξησης και μείωσης

- Τελεστής αύξησης (++)
  Παράδειγμα:
  `c++;` είναι ίδιο με το `c += 1;` ίδιο με το `c = c + 1;`

- Τελεστής μείωσης(--)
  Παράδειγμα:
  `c--;` είναι ίδιο με το `c -= 1;` ίδιο με το `c = c - 1;`

- Γρηγορότερος τελεστής –
  Εξοικονομεί χρόνο στις πολλές επαναλήψεις.

- Μπορεί να προ-αυξηθεί / μειωθεί ή να μετά-αυξηθεί / μειωθεί.
  - Διαφέρει μόνο όταν η μεταβλητή εμφανίζεται στο πλαίσιο μεγαλύτερης έκφρασης.

Τελεστές αύξησης και μείωσης II

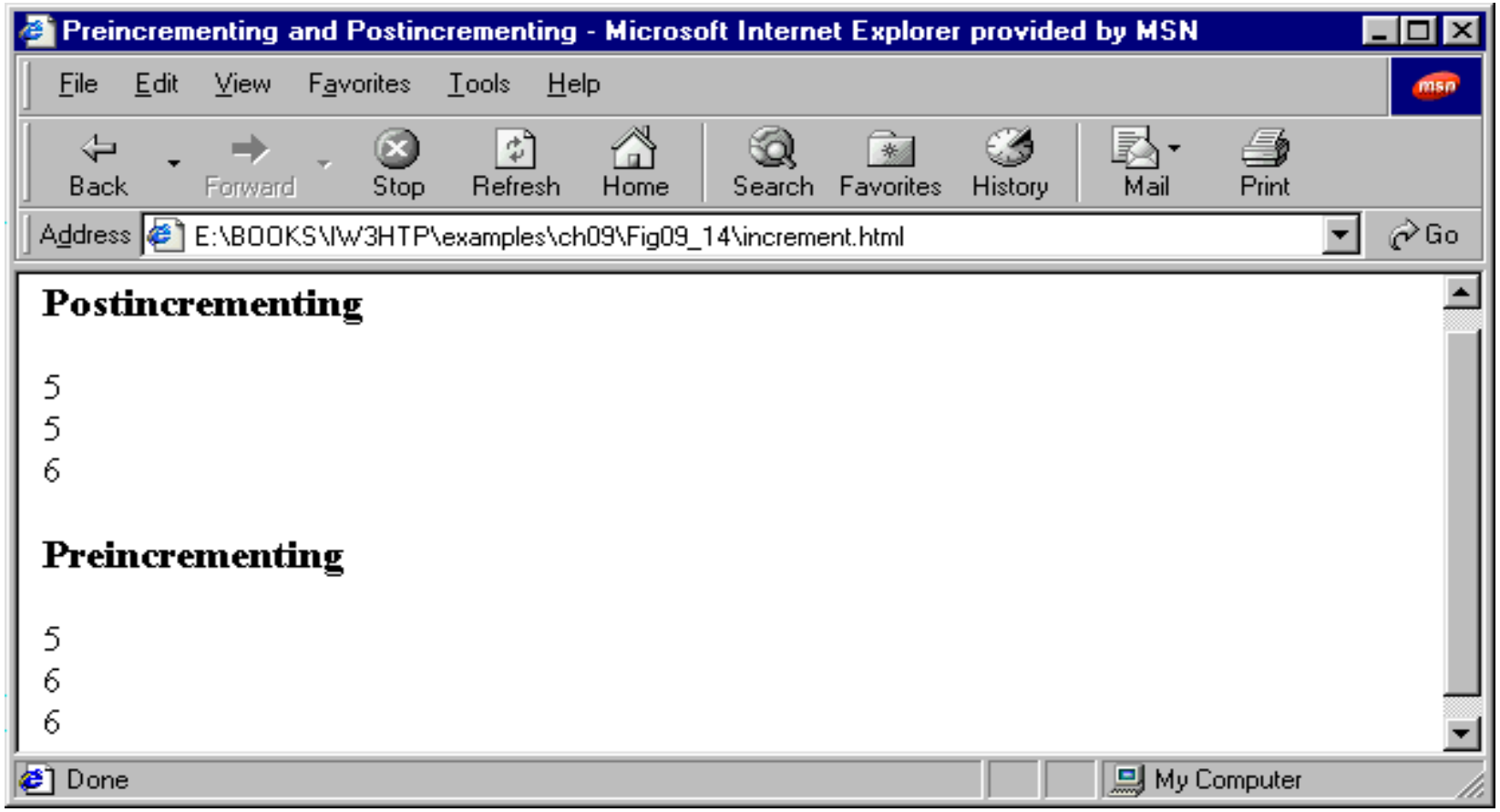
Τελεστές αύξησης και μείωσης

Τελεστής	Κλήση	Παράδειγμα έκφρασης	Εξήγηση
++	προ-αύξηση	++a	Αυξάνει το a κατά 1, μετά χρησιμοποιεί τη νέα τιμή του a στην έκφραση στην οποία βρίσκεται η a.
++	μετά-αύξηση	a++	Χρησιμοποιεί την τρέχουσα τιμή του a στην έκφραση που βρίσκεται, μετά μειώνει την τιμή του a κατά 1.
--	προ-μείωση	--b	Μειώνει το b κατά 1, μετά χρησιμοποιεί τη νέα τιμή του b στην έκφραση στην οποία βρίσκεται η b.
--	μετά-μείωση	b--	Χρησιμοποιεί την τρέχουσα τιμή του b στην έκφραση που βρίσκεται, μετά μειώνει την τιμή του b κατά 1.

Πρόγραμμα προ-αύξησης και μετά-αύξησης τιμής μεταβλητής

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<!-- Fig. 9.14: increment.html -->
<HEAD>
<TITLE>Preincrementing and Postincrementing</TITLE>

   <SCRIPT LANGUAGE = "JavaScript">
         var c;
         c = 5;

         document.writeln( "<H3>Postincrementing</H3>" );
         document.writeln(c); //print 5
         document.writeln("<BR> + c++); //print 5 then increment
         document.writeln("<BR>" +c); //print 6

         c = 5;
         document.writeln( "<H3>Preincrementing</H3>" );
         document.writeln(c); //print 5
         document.writeln("<BR> + ++c); // increment then print 6
         document.writeln("<BR>" +c); //print 6


</SCRIPT>
</HEAD><BODY></BODY>
</HTML>
```

Έξοδος προγράμματος προ-αύξησης και μετά-αύξησης



Τελεστές αύξησης και μείωσης III

Τα ακόλουθα επιστρέφουν τα ίδια αποτελέσματα:

- Δηλώσεις εκχώρησης

```
passes  =  passes + 1;
```

- Δηλώσεις εκχώρησης

```
passes += 1;
```

- Τελεστής προ-αύξησης

```
++passes;
```

- Τελεστής μετά-αύξησης

```
passes++;
```

Τελεστές αύξησης και μείωσης

Τελεστής	Συσχετισμός	Τύπος
++ --	δεξιά προς τα αριστερά	Μοναδιαίος
* / %	από αριστερά προς τα δεξιά	Πολλαπλασιαστικός
+ -	από αριστερά προς τα δεξιά	Προσθετικός
< <= > >=	από αριστερά προς τα δεξιά	Σχεσιακός
== !=	από αριστερά προς τα δεξιά	Ισότητας
? :	δεξιά προς τα αριστερά	Υποθετικός
= += -= *= /= %=	δεξιά προς τα αριστερά	Εκχώρησης
Την προτεραιότητα και το συσχετισμό των τελεστών που συζητήθηκαν μέχρι στιγμής.		

Μια σημείωση στους τύπους δεδομένων

- JavaScript – πρόχειρα γραμμένη γλώσσα.
  - Δεν απαιτεί από τη μεταβλητή να έχει τύπο πριν από τη χρήση στο πρόγραμμα (σε αντίθεση με άλλες γλώσσες).
  - Η μεταβλητή μπορεί να περιέχει μια τιμή οποιουδήποτε τύπου δεδομένων.
  - Η JavaScript συχνά μετατρέπει αυτόματα τις τιμές διαφορετικών τύπων.

- Όταν δηλώνετε μεταβλητές.
  - Εάν δεν δίνεται τιμή, η μεταβλητή έχει απροσδιόριστη τιμή.
  - Για να υποδείξετε ότι η μεταβλητή δεν έχει τιμή, αναθέστε την σε μηδενική.

2ο μέρος – Δηλώσεις ελέγχου

Περίγραμμα

- Εισαγωγή
- Τα βασικά στοιχεία της επανάληψης ελεγχόμενη με μετρητή
- Η δομή επανάληψης
- Παραδείγματα
- Η πολλαπλή δομή επιλογής switch
- Δομή επανάληψης do/while
- Το break και το continue
- Οι δηλώσεις break και continue
- Λογικοί τελεστές
- Περίληψη Δομημένου Προγραμματισμού

Εισαγωγή

- Πριν από τον προγραμματισμό ενός σεναρίου να έχετε:

  - απόλυτη κατανόηση του προβλήματος.
  - προσεκτικά σχεδιασμένη προσέγγιση για την επίλυσή του.

- Κατά τη σύνταξη ενός σεναρίου, σημαντικό είναι να

  - κατανοήσετε τους τύπους των δομικών στοιχείων και τα διαθέσιμα εργαλεία.
  - χρησιμοποιήσετε αποδεδειγμένες αρχές κατασκευής του προγράμματος.

Τα βασικά στοιχεία της επανάληψης ελεγχόμενη με μετρητή

- Η επανάληψης ελεγχόμενη με μετρητή απαιτεί:

1. Όνομα μεταβλητής ελέγχου (ή μετρητής βρόχου).
2. Αρχική τιμή της μεταβλητής ελέγχου.
3. Αύξηση (ή μείωση) της μεταβλητής ελέγχου ανά βρόχο.
4. Συνθήκη που ελέγχει την τελική τιμή της μεταβλητής ελέγχου.

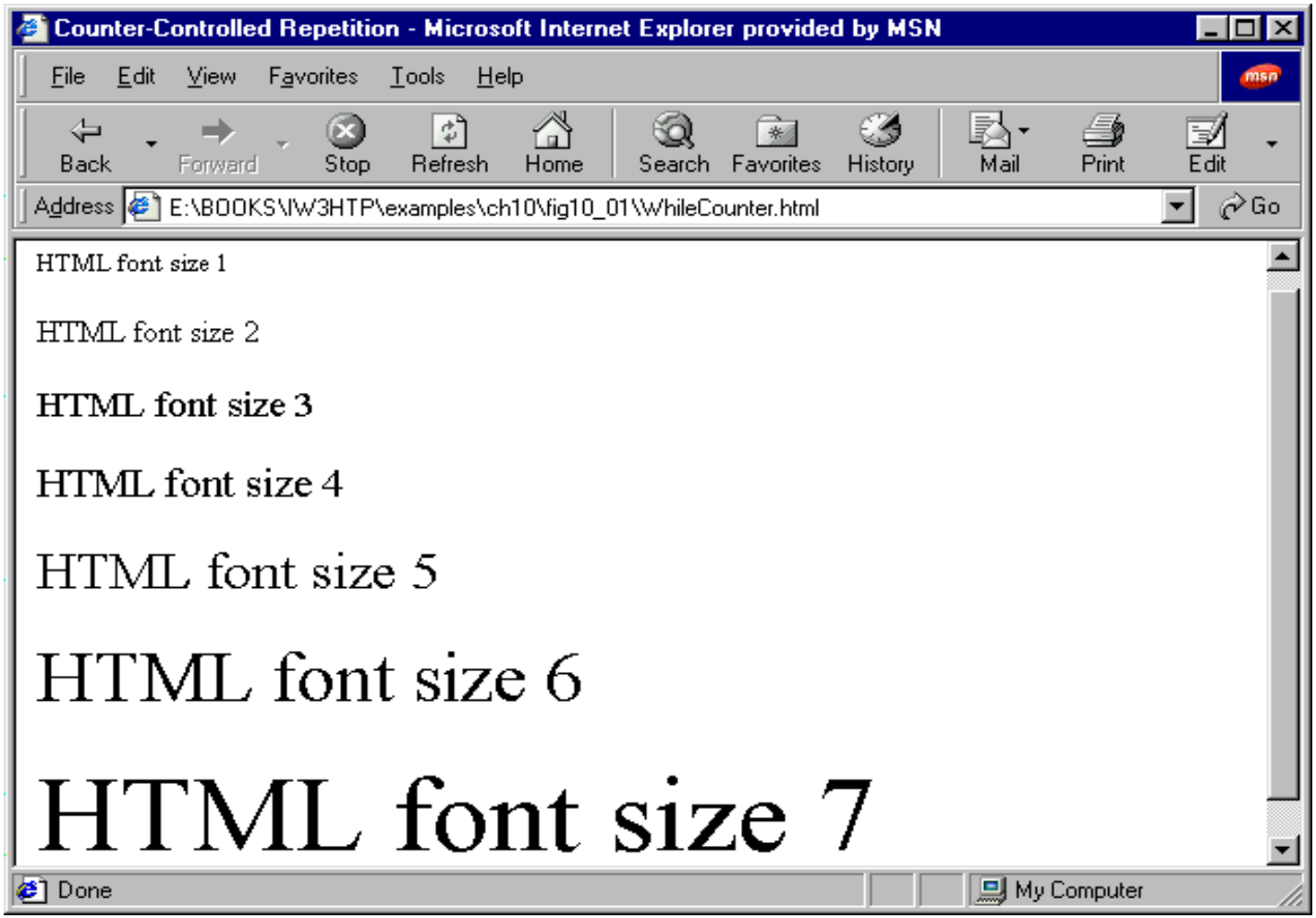
- Αναγνώριση προγράμματος:

- Δηλώσεις εσοχής στο σώμα κάθε δομής ελέγχου.
- Κενή γραμμή πριν και μετά από κάθε μεγάλη δομή ελέγχου.
- Αποφύγετε περισσότερα από τρία επίπεδα εμφώλευσης.

Πρόγραμμα επανάληψης με τη δομή while

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0
Transitional//EN">
<HTML>
<!-- Fig. 10.1: WhileCounter.html -->
<HEAD>
<TITLE>Counter-Controlled Repetition</TITLE>

  <SCRIPT LANGUAGE = "JavaScript">
        var counter = 1; //initialization

        while ( counter <= 7 ) {    //repetition condition
               document.writeln("<P><FONT SIZE = "+counter+">
                      HTML font size "+counter+ "</FONT></P>");
               ++counter;   //increment
        }
</SCRIPT>
</HEAD><BODY></BODY>
</HTML>
```

Έξοδος σεναρίου χρήσης της δομής επανάληψης while



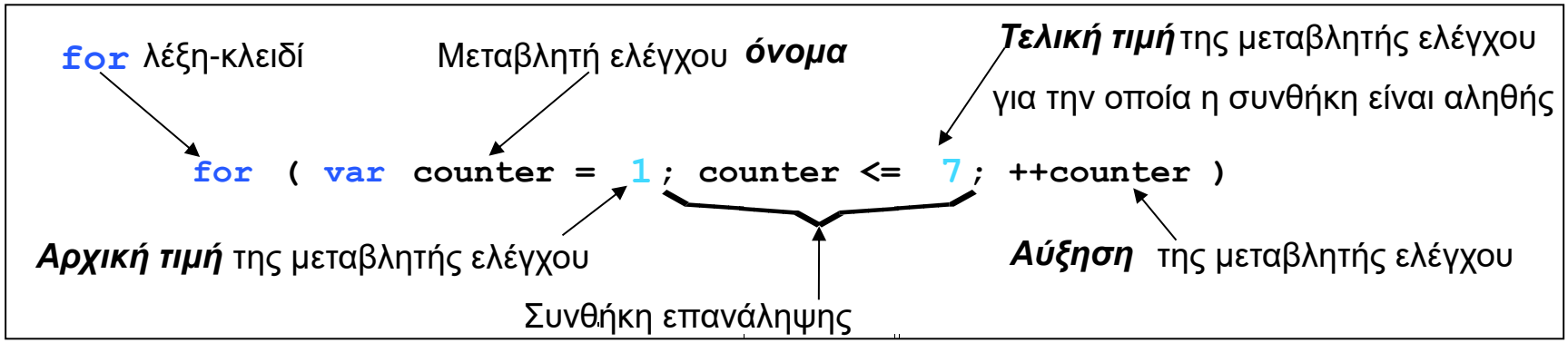
Η δομή επανάληψης for

- *Δομή επανάληψης for* :
  - Χειρίζεται όλες τις λεπτομέρειες της επανάληψης.

- Δήλωση JavaScript :

```
for ( var counter = 1 ; counter <= 7 ; counter ++ )
      document.writeln( "<P> <FONT SIZE =" + counter
+ ">HTML Font size " + counter + "</FONT>" );
```

Η δομή επανάληψης for II



Εικόνα: Συστατικά μιας τυπικής κεφαλίδας της δομής for.

Η δομή επανάληψης for III

Ισοδύναμες δομές

- **for** structure:

```
for ( initialization; loopContinuationTest ; increment )
        statement;
```

- **while** structure:

```
initialization;
while ( loopContinuationTest ) {
              statement;
              increment;
}
```

Η δομή επανάληψης for IV

- Τρεις εκφράσεις για τη δομή for είναι προαιρετικές.

- Εάν η παράμετρος loopContinuationTest παραλείπεται, η JavaScript υποθέτει ότι η συνθήκη ισχύει (true).
  - Οδηγεί σε άπειρο βρόχο.
- Μπορεί να παραλείψει την έκφραση αρχικοποίησης εάν η μεταβλητή αρχικοποιείται σε άλλο σημείο του προγράμματος.
- Μπορεί να παραλείψει τη δήλωση αύξησης εάν εμφανιστεί αύξηση μέσα στη δομή.

- Αν η συνθήκη συνέχισης του βρόχου αρχικά ήταν ψευδής (false), το σώμα της δομής for δεν εκτελέστηκε.

- Καθυστέρηση βρόχου
- η δομή for είναι κενή προσθέτοντας το ερωτηματικό.
- ο βρόχος εξακολουθεί να λειτουργεί συγκεκριμένες φορές.
- είναι χρήσιμη για την επιβράδυνση των προγραμμάτων, αλλά υπάρχουν πιο αποδοτικές τεχνικές (Κεφάλαιο 15).

Παραδείγματα χρήσης της δομής for

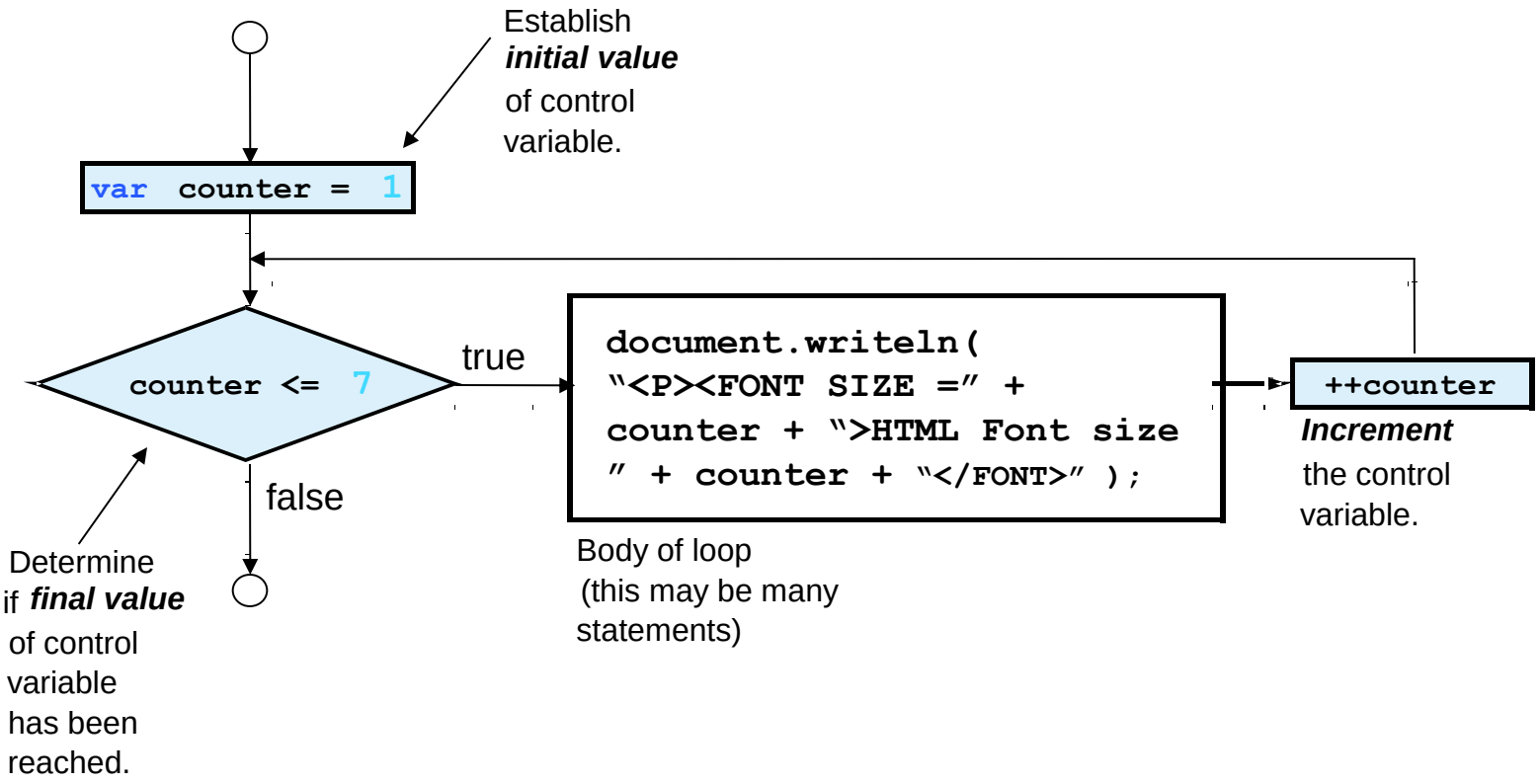
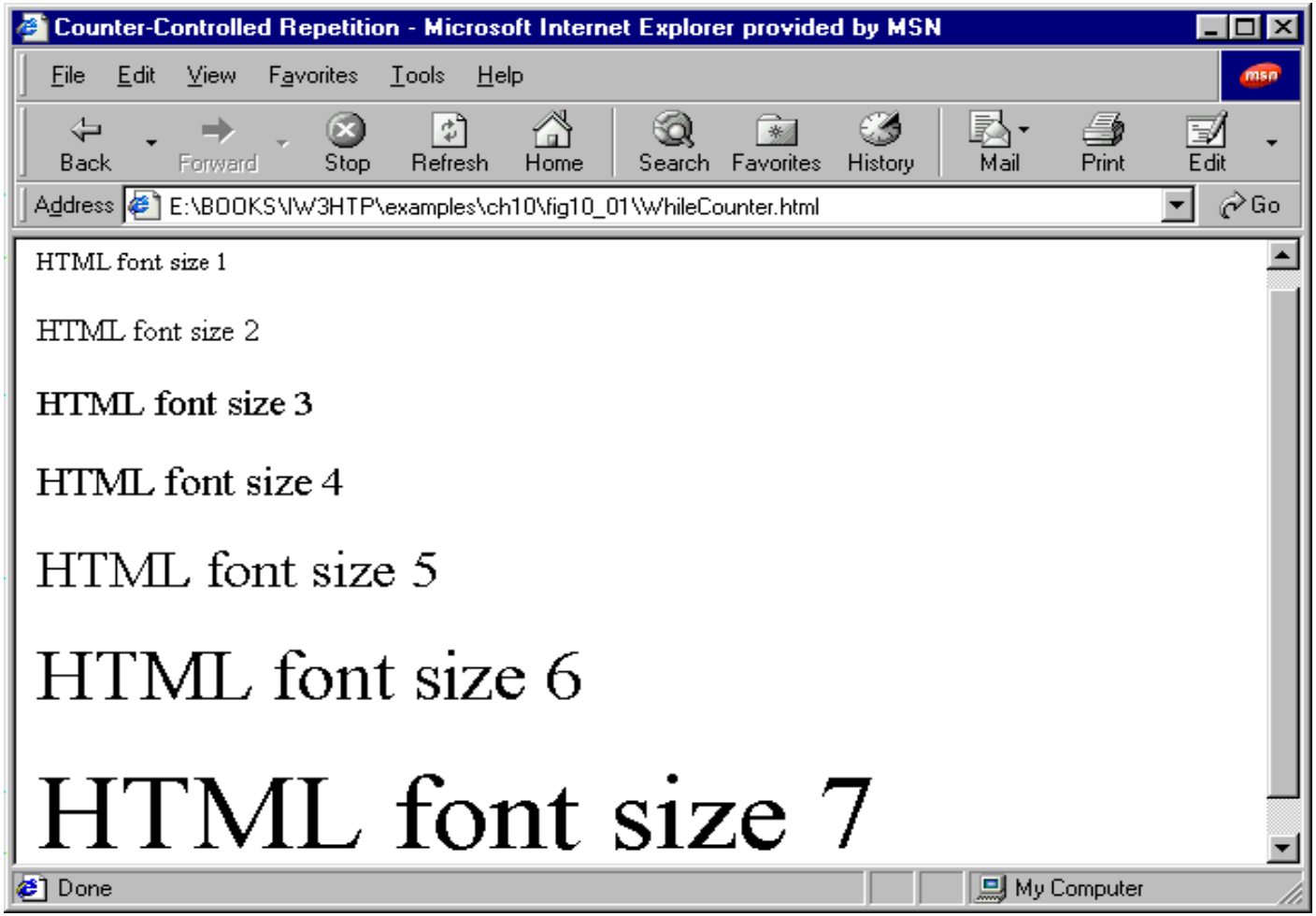


Fig. Flowcharting a typical for repetition structure.

Πρόγραμμα επανάληψης με τη δομή for

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0
Transitional//EN">
<HTML>
<!-- Fig. 10.2: ForCounter.html -->
<HEAD>
<TITLE>Counter-Controlled Repetition</TITLE>

  <SCRIPT LANGUAGE = "JavaScript">

        for( var counter=1; counter <= 7; ++counter ) {
              document.writeln("<P><FONT SIZE = "+counter+">
                    HTML font size "+counter+ </FONT></P>");
        }

</SCRIPT>
</HEAD><BODY></BODY>
</HTML>
```

Έξοδος σεναρίου χρήσης της δομής επανάληψης for



Παραδείγματα χρήσης της δομής for II

- Διαφορετικές μέθοδοι για ποίκιλλες μεταβλητές ελέγχου στη δομή for.
- Παραδείγματα:

Μεταβλητή ελέγχου: 1 έως 100, αυξάνοντας κατά 1.

```
for ( var i = 1; i <= 100; ++i );
```

Μεταβλητή ελέγχου: 100 έως 1, αυξάνοντας κατά -1 (μείωση κατά 1).

```
for ( var i = 100; i >= 1; --i );
```

Μεταβλητή ελέγχου: 7 έως 77, με βήμα 7.

```
for ( var i = 7; i <= 77; i += 7 );
```

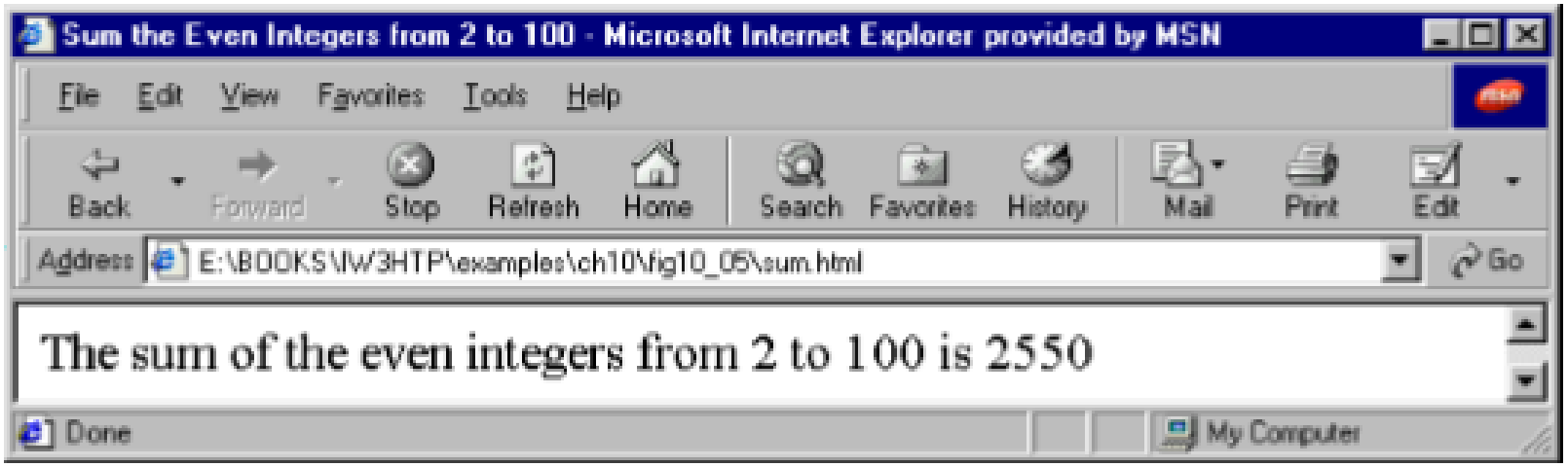
Μεταβλητή ελέγχου σε ακολουθία τιμών: 99, 88, 77, 66, 55, 44, 33, 22, 11, 0

```
for ( var k = 99; k >= 0; k -= 11 );
```

Πρόγραμμα υπολογισμού αθροίσματος με τη δομή επανάληψης for

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<!-- Fig. 10.5: Sum.html -->
<HEAD>
<TITLE>Sum the Even Integers from 2 to 100</TITLE>

  <SCRIPT LANGUAGE = "JavaScript">
        var sum=0;
        for( var number=2; number <= 100; number += 2 )
              sum+=number;

        document.writeln("<BIG>The sum of the even integers " +
              "from 2 to 100 is " +sum+ "</BiG>");

</SCRIPT>
</HEAD><BODY></BODY>
</HTML>
```

Έξοδος υπολογισμού αθροίσματος με τη δομή επανάληψης for



Παραδείγματα χρήσης της δομής for III

Μαθηματικό Αντικείμενο

```
Math.pow (x, y);
```

Υπολογίζει το x που ανεβαίνει στην $y^{η}$ δύναμη

```
Math.round ();
```

Στρογγυλοποιεί την τιμή που έχει εισαχθεί στον πλησιέστερο ακέραιο αριθμό.
Για να εξάγετε έναν αριθμό με το δεύτερο δεκαδικό ψηφίο, χρησιμοποιήστε τον τύπο:

```
Math.round (amount * 100) / 100
```

Παράδειγμα:

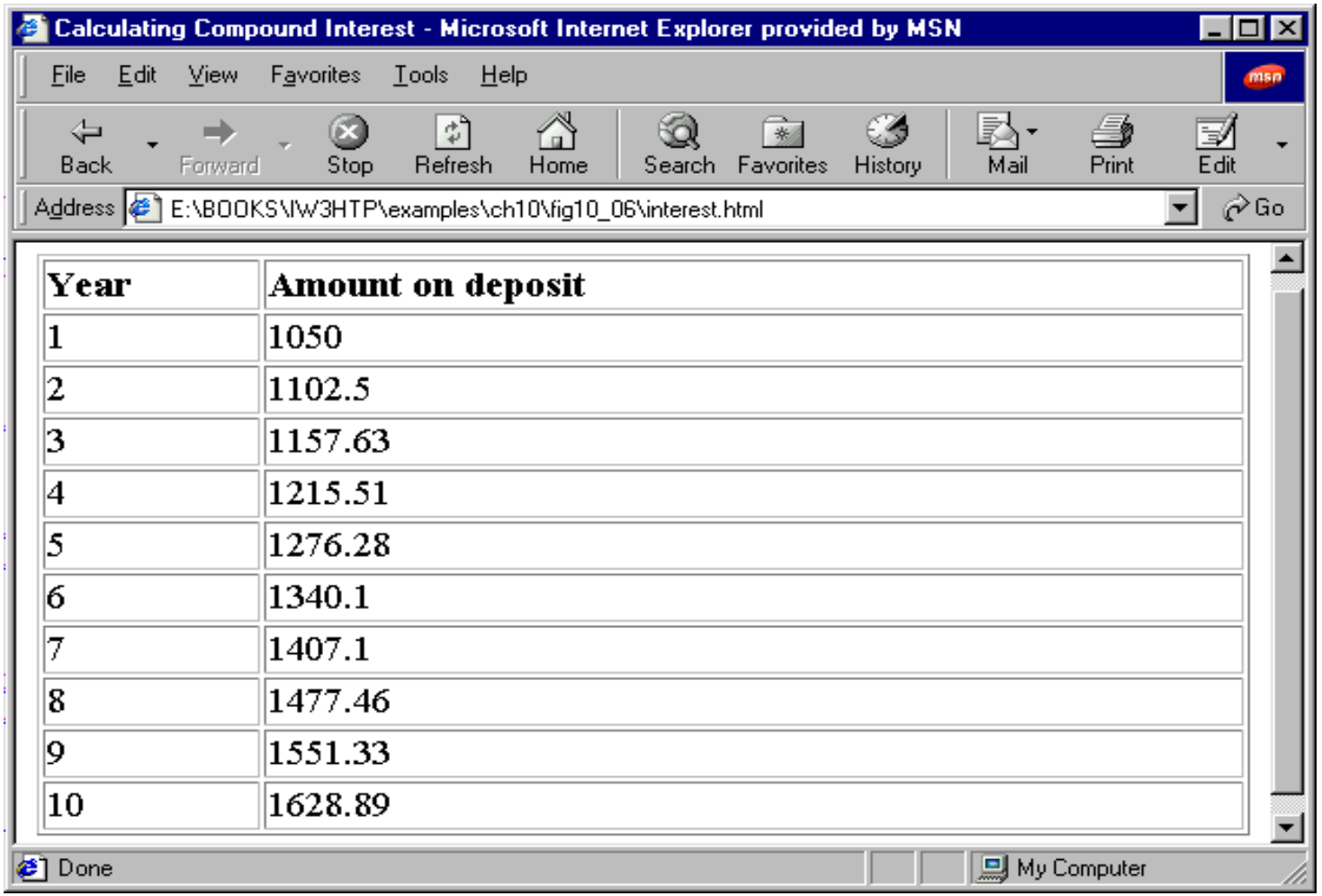
```
Math.round (3.1415 * 100) / 100 = 314/100 = 3.14
```

Η JavaScript παριστάνει όλους τους αριθμούς ως αριθμούς κινητής υποδιαστολής. Όταν οι αριθμοί κινητής υποδιαστολής στρογγυλοποιούνται, το αποτέλεσμα μπορεί να μην είναι εντελώς σωστό (ειδικά όταν χρησιμοποιείται σε εξισώσεις με άλλες στρογγυλεμένες τιμές).

Παράδειγμα

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<!-- Fig. 10.6: interest.html -->
<HEAD>
<TITLE>Calculating Compound Interest</TITLE>

   <SCRIPT LANGUAGE = "JavaScript">
          var amount, principal = 1000.0, rate= .05;

          document.writeln("<TABLE BORDER = "100") ;
          document.writeln("<TR><TD WIDTH="100%"/>) ;
          document.writeln("<TD><B>Amount on deposit</B></TD>") ;

          for( var year=1; year <= 10; ++year ){
                    amount = principal * Math.pow(1.0 +rate, year) ;
                    document.writeln("<TR><TD>" +year+ "<TR><TD>" +
                              Math.round(amount * 100) / 100 + "</TD></TR>" );
          }
          document.writeln("</TABLE>") ;

</SCRIPT>
</HEAD><BODY></BODY>
</HTML>
```

Έξοδος προηγούμενου σεναρίου



Year	Amount on deposit
1	1050
2	1102.5
3	1157.63
4	1215.51
5	1276.28
6	1340.1
7	1407.1
8	1477.46
9	1551.33
10	1628.89

Η δομή πολλαπλής επιλογής switch

- Δομή επιλογής switch

Περιέχει πολλά σκέλη.
Οι ενέργειες που εκτελούνται εξαρτώνται από τη μεταβλητή τιμή.
Λειτουργεί καλά ταξινομώντας τις εισόδους των χρηστών.

- Δήλωση break

Πηγαίνει στο τέλος της δομής switch.
Θα πρέπει να είναι στο τέλος κάθε σκέλους (case) της δομής.
Αν παραλειφθεί, η JavaScript θα συνεχίσει να ελέγχει την είσοδο των χρηστών από τις περιπτώσεις (cases).

Η δομή πολλαπλής επιλογής switch II

- default case (περίπτωση προεπιλογής)

Εκτελείται εάν η μεταβλητή δεν αντιστοιχεί σε καμία από τις περιπτώσεις.

- Ορθές πρακτικές:

- Ελέγξτε αν έχει εισαχθεί έγκυρη τιμή από τον χρήστη.
- Προσπερνά όλες τις περιπτώσεις της δομής.

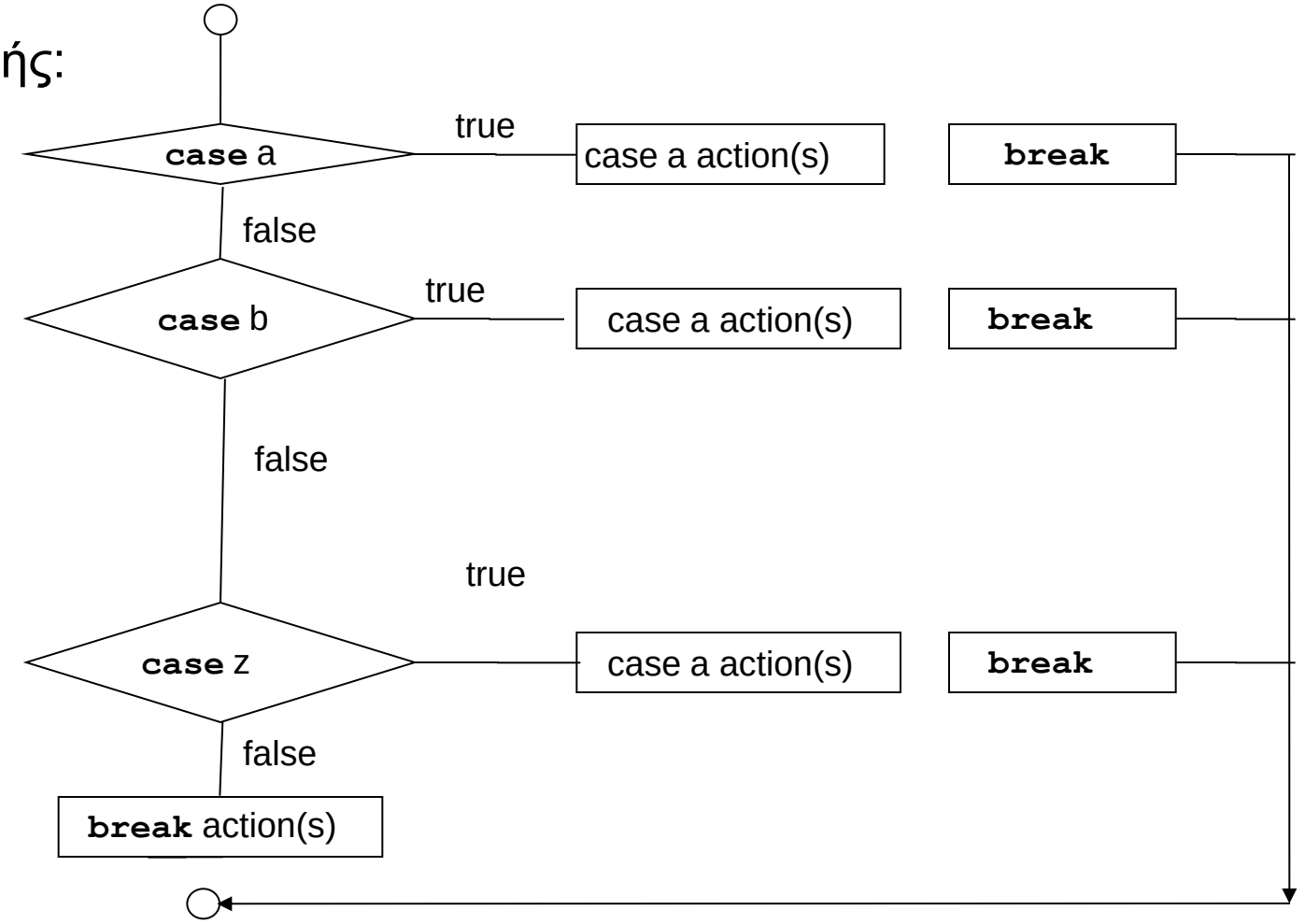
Η δομή πολλαπλής επιλογής switch III

Δήλωση Javascript:

```
var choice;
   choice = window.prompt();

   switch ( choice ) {
       case “a”:
           actions
           break;
       case “b”:
           actions
           break;
       case “z”:
          actions
          break;
       default:
           actions
   }
```

Η δομή πολλαπλής επιλογής switch IV

- Διάγραμμα Ροής:



Παράδειγμα case

```
<script type="text/javascript">
   //You will receive a different greeting based
   //on what day it is. Note that Sunday=0,
   //Monday=1, Tuesday=2, etc.

   var d=new Date();
   theDay=d.getDay();
   switch (theDay)
   {
   case 5:
   document.write("Finally Friday");
   break;
   case 6:
   document.write("Super Saturday");
   break;
   case 0:
   document.write("Sleepy Sunday");
   break;
   default:
   document.write("I'm looking forward to this weekend!");
   }
   </script>
```

Παράδειγμα χρήσης πολλαπλής επιλογής switch

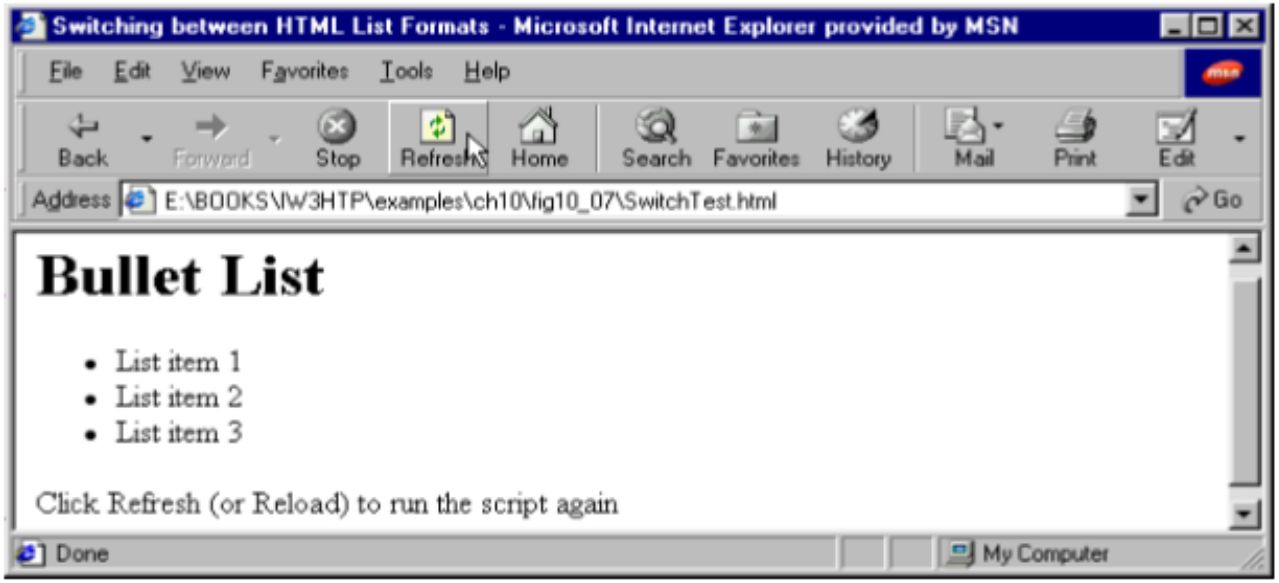
```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<!-- Fig. 10.7: SwitchTest.html -->
<HEAD>
<TITLE>Switching between HTML List Formats</TITLE>

   <SCRIPT LANGUAGE = "JavaScript">
         var choice, startTag, endTag, validInput=true, listType;

         choice = window.prompt("Select a list style:\n" + "1 (bullet), 2
         (numbered), 3 (lettered)", "1") ;
         switch( choice) {
                  case "1":
                           startTag = "<UL>" ;
                           endTag = "</UL>" ;
                           listType = "<H1>Bullet List</H1>" ;
                           break ;
                  case "2":
                           startTag = "<OL>" ;
                           endTag = "</OL>" ;
                           listType = "<H1>Ordered List: Numbered</H1>" ;
                           break ;
```

Παράδειγμα χρήσης πολλαπλής επιλογής switch

```
                case "3":
                        startTag = "<OL TYPE = 'A'>" ;
                        endTag = "</OL>" ;
                        listType = "<H1>Ordered List: Lettered</H1>" ;
                        break ;
                 default:
                        validInput = false ;
        }

        if (validInput == true) {
                document.writeln(listType + startTag ) ;

                for ( var i=1; i<=3; i++)
                        document.writeln("List item " +i);
                document.writeln(endTag) ;
        }
        else
                document.writeln("Invalid choice: " + choice) ;

</SCRIPT>
</HEAD><BODY><P> Click Refresh (or Reload) to run the script again </P></BODY>
</HTML>
```

Πρώτη εκτέλεση προγράμματος χρήσης πολλαπλής επιλογής switch

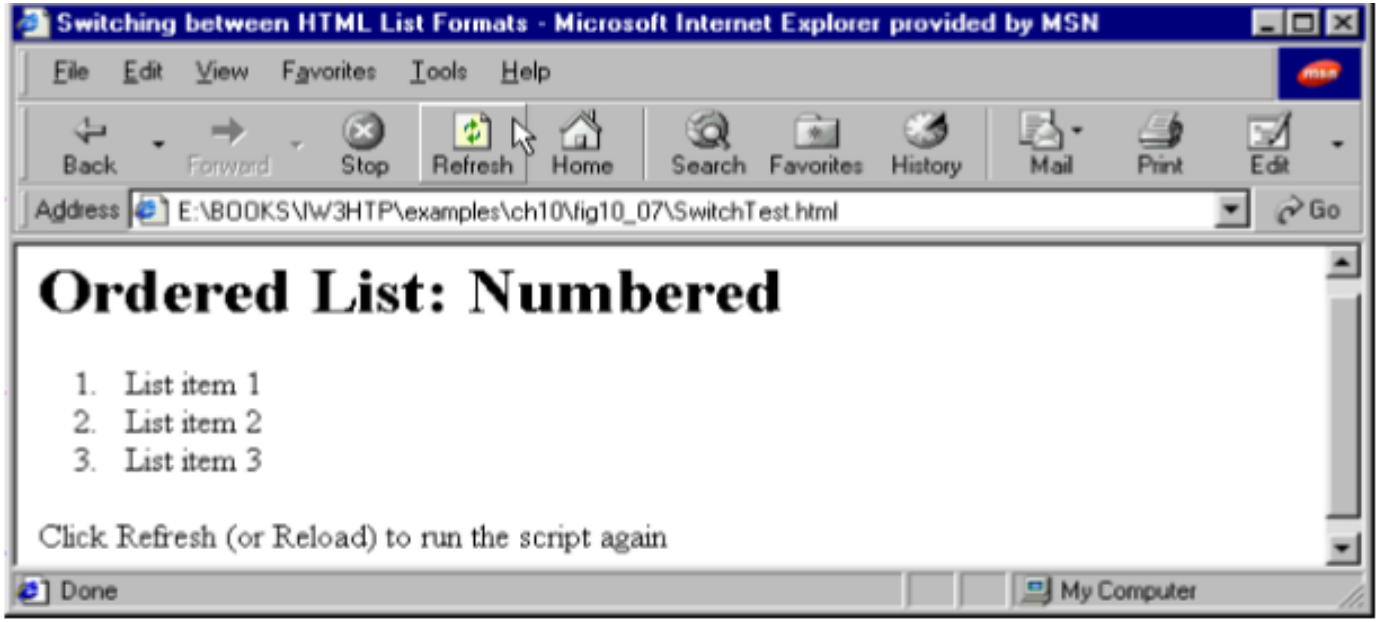
Είσοδος χρήστη: 1



Έξοδος προγράμματος

Δεύτερη εκτέλεση προγράμματος χρήσης πολλαπλής επιλογής switch

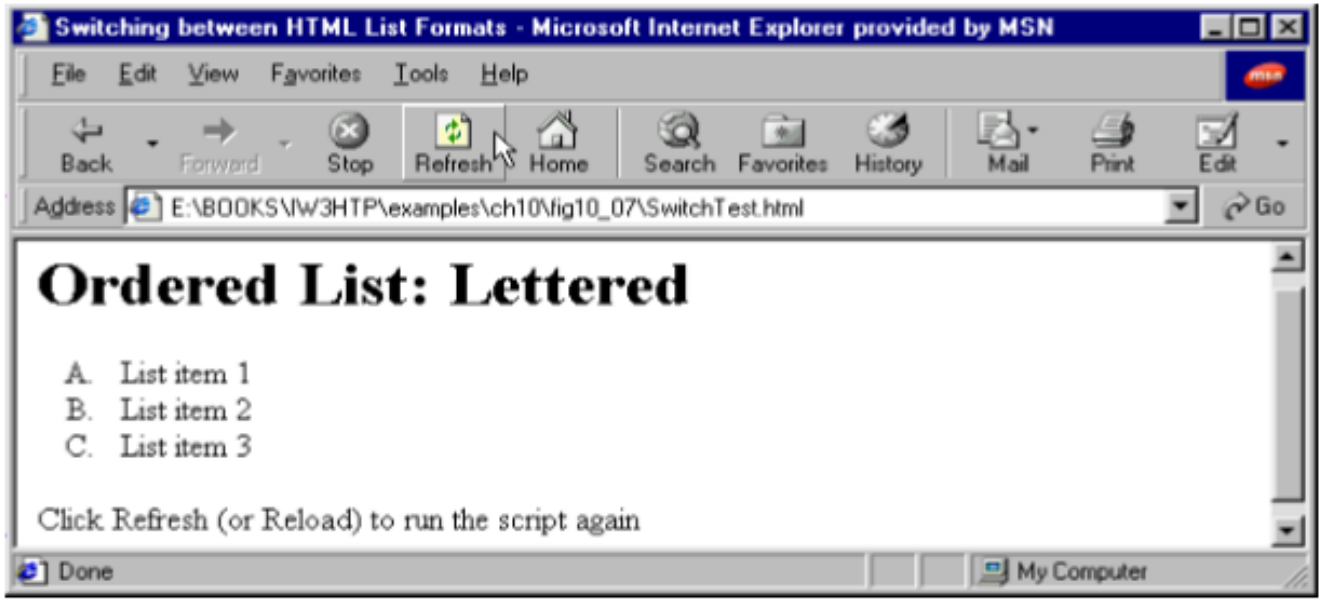
Είσοδος χρήστη: 2



Έξοδος προγράμματος

Τρίτη εκτέλεση προγράμματος χρήσης πολλαπλής επιλογής switch

Είσοδος χρήστη: 3



Έξοδος προγράμματος

Η δομή επανάληψης do/while

- Παρόμοια με τη δομή ελέγχου while
- Διαφορά

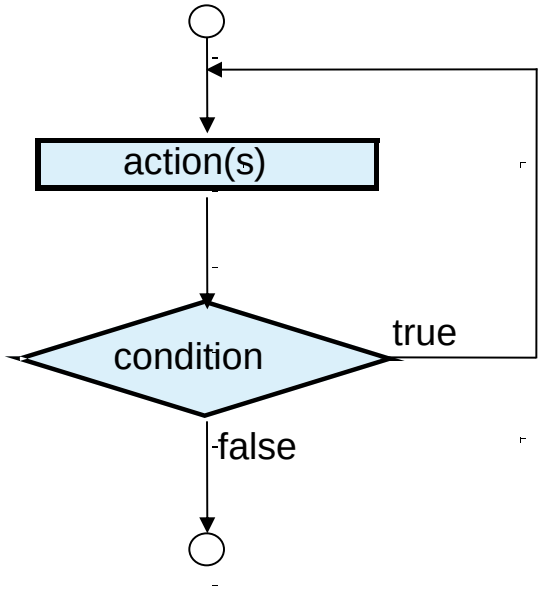
  while: η δομή εκτελείται μόνο αν η συνθήκη είναι αρχικά αληθης.

```
while ( condition ) {
    statement
}
```

  do/while: η δομή εκτελείται πάντα τουλάχιστον μία φορά.

```
do {
    statement
} while ( condition );
```

Η δομή επανάληψης do/while

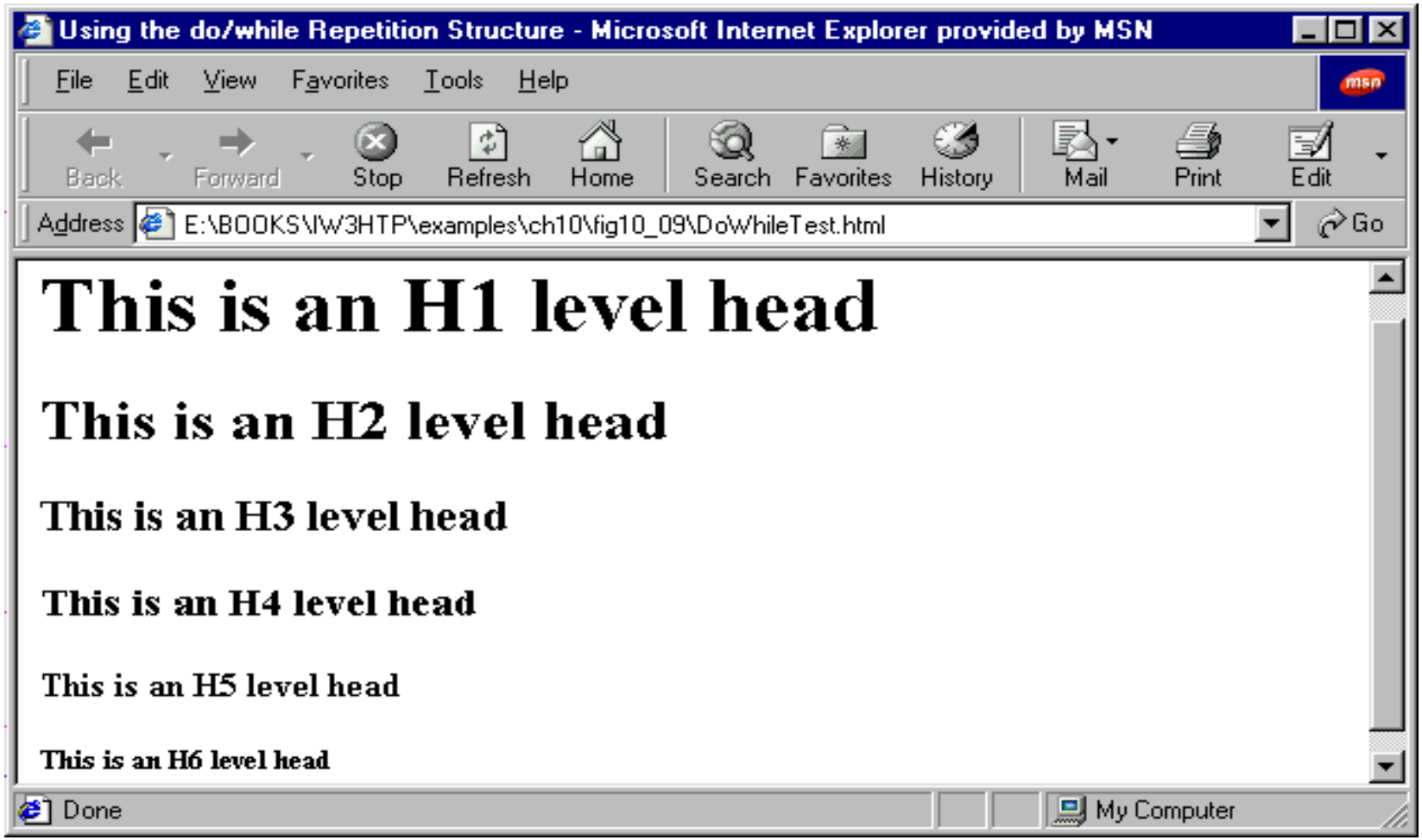


Εικόνα. Διάγραμμα ροής της do/while δομής επανάληψης.

Παράδειγμα χρήσης της δομής do/while

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<!-- Fig. 10.9: DoWhileTest.html -->
<HEAD>
<TITLE>Using the do/while Repetition Structure</TITLE>

   <SCRIPT LANGUAGE = "JavaScript">
         var counter = 1;

         do(
                  document.writeln("<H" +counter+">This is an H"+counter+
                           "level head" +"</H" +counter+ ">") ;
                  ++counter;
         }while(counter <= 6 ) ;

</SCRIPT>
</HEAD><BODY></BODY>
</HTML>
```

Έξοδος σεναρίου χρήσης της δομής do/while



Οι δηλώσεις break και continue

- Αλλάζει τη ροή του ελέγχου

```
break;
```

- Έξοδος από τη δομή.

```
continue;
```

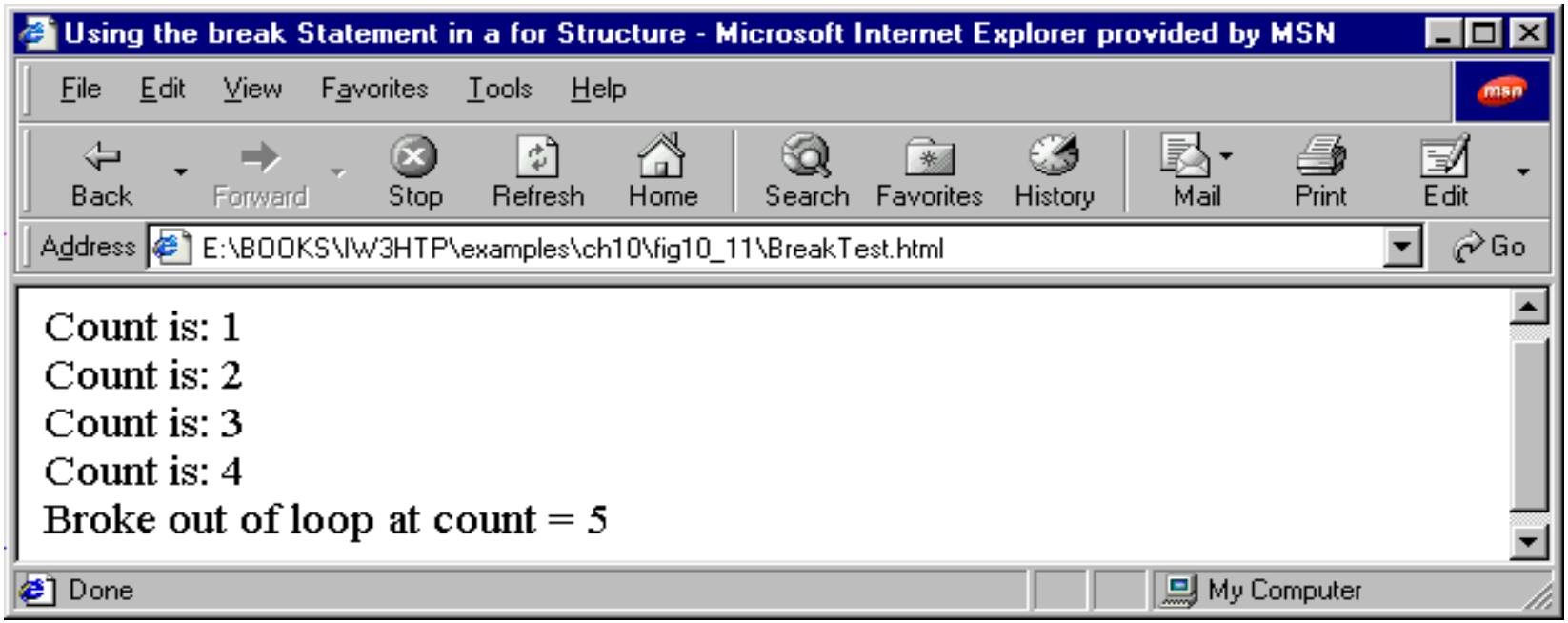
- Παραλείπει τις υπόλοιπες δηλώσεις στη δομή και συνεχίζει με την επόμενη επανάληψη (loop).

- Όταν χρησιμοποιείται σωστά
  - Εκτελείται ταχύτερα από τις αντίστοιχες δομημένες τεχνικές.

Παράδειγμα χρήσης της εντολής break

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<!-- Fig. 10.11: BreakTest.html -->
<HEAD>
<TITLE>Using the break Statement in a for Structure</TITLE>

   <SCRIPT LANGUAGE = "JavaScript">

         for( var count=1; count <= 10; ++count ){
                  if(count == 5)
                           break; //break loop only if count == 5
                  document.writeln("Count is: " +count+ "<BR>" );
         }
         document.writeln("Broke out of loop at count = " +count) ;

</SCRIPT>
</HEAD><BODY></BODY>
</HTML>
```

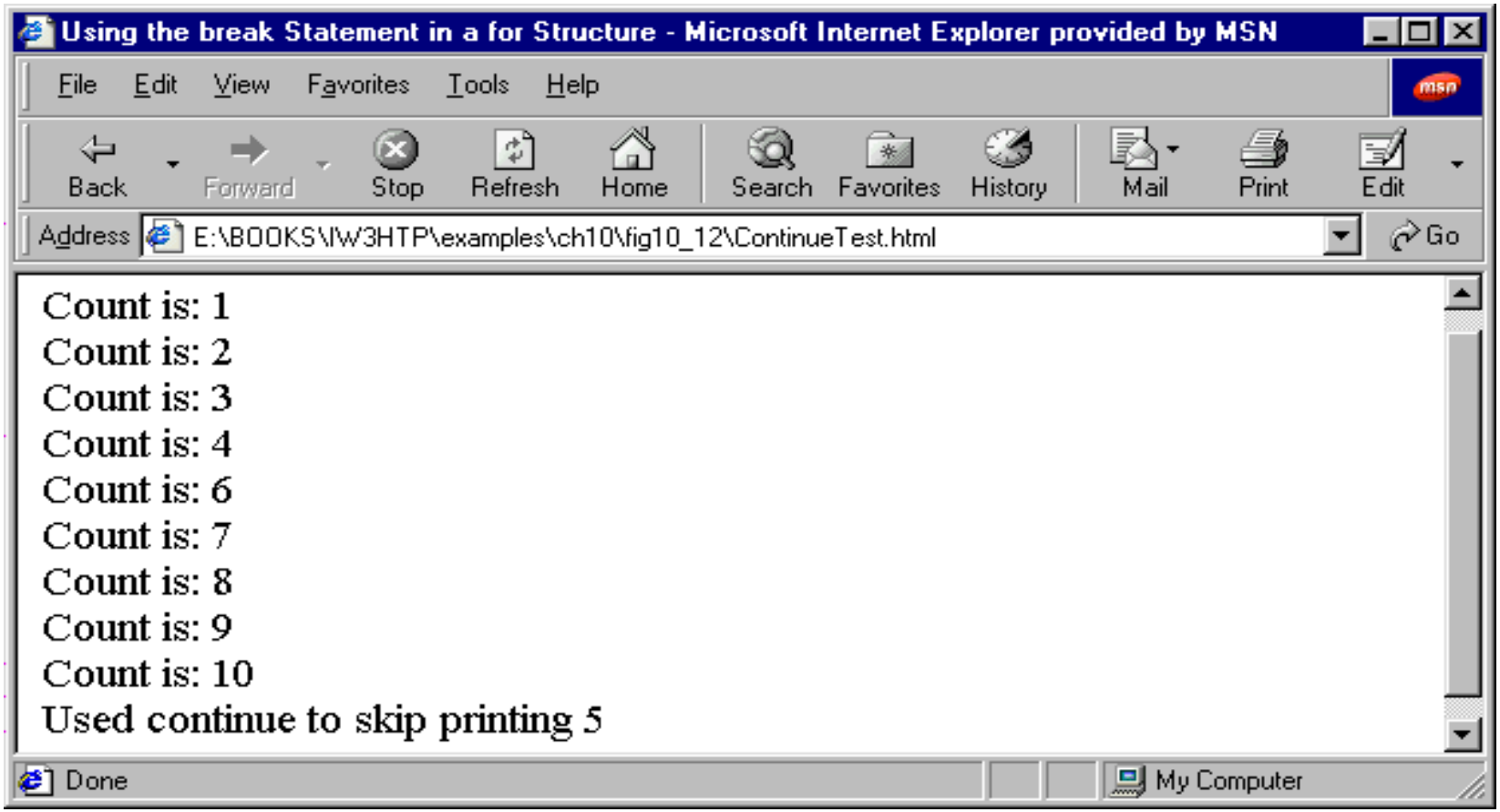
Έξοδος σεναρίου χρήσης break



Παράδειγμα χρήσης της εντολής break

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<!-- Fig. 10.12: ContinueTest.html -->
<HEAD>
<TITLE>Using the continue Statement in a for Structure</TITLE>

   <SCRIPT LANGUAGE = "JavaScript">

         for( var count=1; count <= 10; ++count ){
                  if(count == 5)
                           coninue; //skip remaining code in loop
                                    //only if count == 5

                  document.writeln("Count is: " +count+ "<BR>" );
         }
         document.writeln("Used coninue to skip printing 5") ;

</SCRIPT>
</HEAD><BODY></BODY>
</HTML>
```

Έξοδος σεναρίου



Οι δηλώσεις break και continue

- Δήλωση break

«Ξεφεύγει» από την άμεση ενσωμάτωση της δομής ελέγχου επανάληψης.

- Για να ξεφύγετε από ένθετες δομές.

  - Χρησιμοποιήστε τις labeled δηλώσεις break.
  - Ξεκινά με μια ετικέτα (αναγνωριστικό που ακολουθείται από τελεία).
  - Εγκλείστε τις δομές που πρέπει να κάνουν break με άγκιστρα ({}).
  - Ονομάζεται labeled compound statement.
  - Κατά την εκτέλεση της εντολής break, ακολουθήστε τη μορφή:

```
break label ;
```

Οι δηλώσεις labeled break και continue II

- Χρησιμοποιήστε τις labeled δηλώσεις continue.

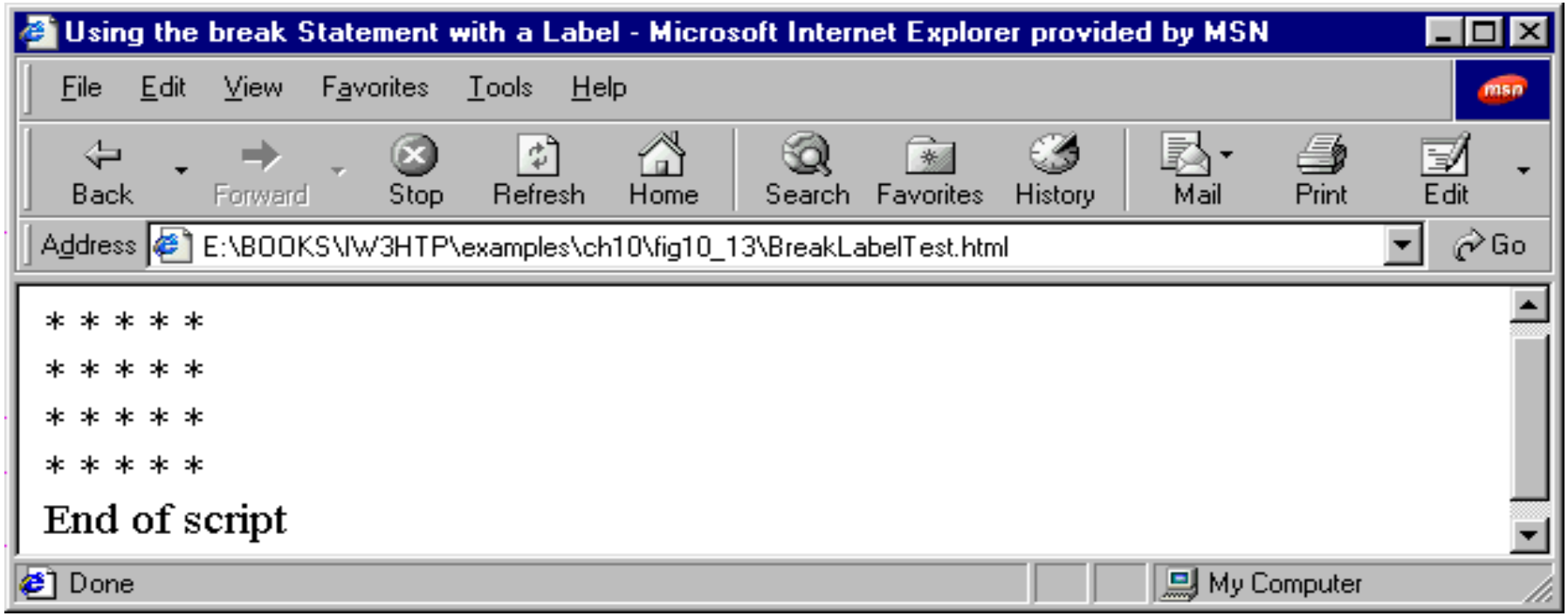
  - Ακολουθεί τους ίδιους συντακτικούς κανόνες.
  - Μετά την εκτέλεση, συνεχίζεται με την επόμενη επανάληψη της labeled δομής επανάληψης.

- Ορθή πρακτική είναι να εισάγετε τη δήλωση εξόδου για να ελέγξετε εάν η labeled δήλωση εκτελείται σωστά.

Παράδειγμα χρήσης της labeled break

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<!-- Fig. 10.13: BreakLabelTest.html -->
<HEAD>
<TITLE>Using the break Statement with a Label</TITLE>

   <SCRIPT LANGUAGE = "JavaScript">
      stop: {
          for( var row=1; row <= 5; ++row ){
                    for( var column=1; column <= 5; ++column ){
                              if(row == 5)
                                        break stop; //jump to end of stop block

                              document.writeln("* " );
                    }
                    document.writeln("<BR>") ;
          }
          document.writeln("This line should not print") ;
      }
      document.writeln("End of script");
</SCRIPT>
</HEAD><BODY></BODY>
</HTML>
```

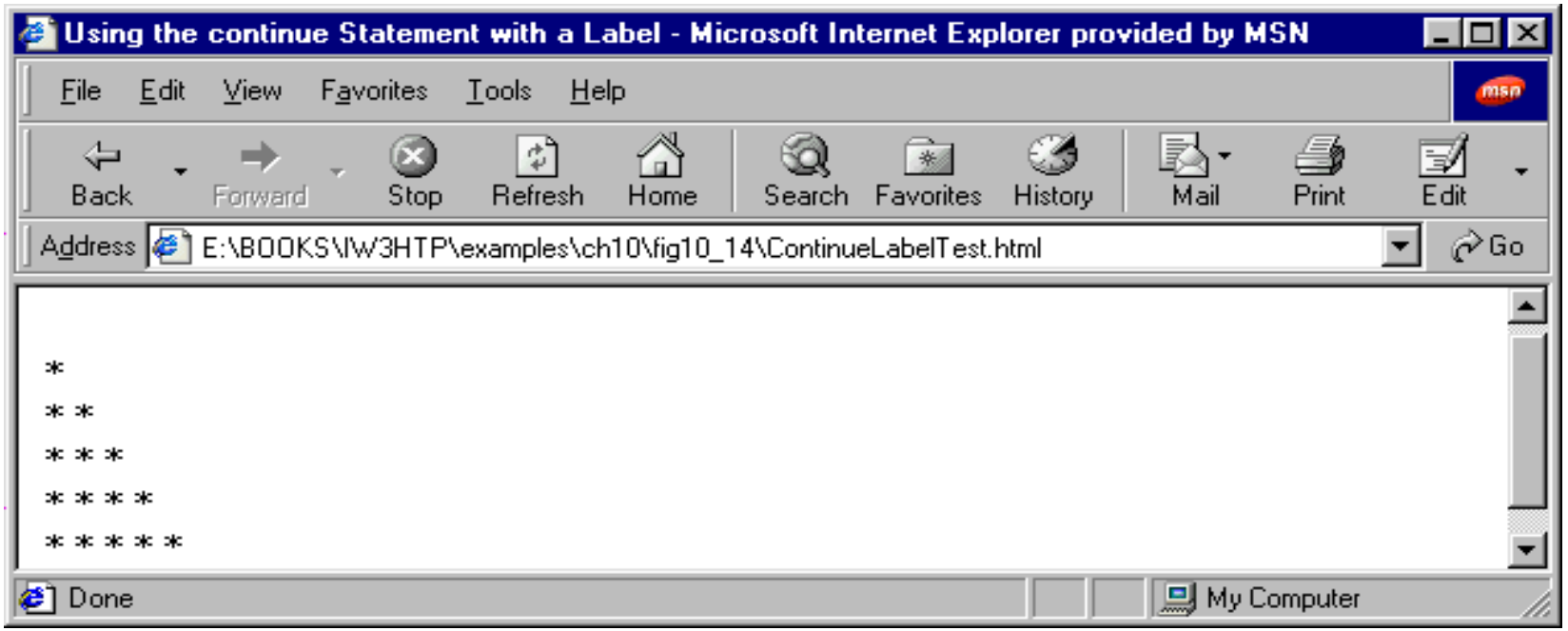
Έξοδος προγράμματος χρήσης labeled break



Παράδειγμα χρήσης της labeled continue

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<!-- Fig. 10.14: ContinueLabelTest.html -->
<HEAD>
<TITLE>Using the continue Statement with a Label</TITLE>

   <SCRIPT LANGUAGE = "JavaScript">
      nextRow:
          for( var row=1; row <= 5; ++row ){
                    document.writeln("<BR>") ;

                    for( var column=1; column <= 10; ++column ){
                              if(colum > row)
                                        continue nextRow;

                              document.writeln("* " );
                    }
           }

</SCRIPT>
</HEAD><BODY></BODY>
</HTML>
```

Έξοδος προγράμματος χρήσης labeled continue



Λογικοί τελεστές

- Λογικοί τελεστές

  - Χρησιμοποιούνται για τον σχηματισμό πιο σύνθετων συνθηκών συνδυάζοντας απλές συνθήκες.

- Οι λογικοί τελεστές είναι οι εξής:

  && (λογικό AND)
  || (λογικό OR)
  != (λογικό NOT ή λογική άρνηση)

Λογικοί τελεστές ΙΙ

&& (λογικό AND)

- Όλες οι δηλώσεις που χρησιμοποιούνται στους τελεστές && σε μια συνθήκη πρέπει να είναι αληθής για να είναι αληθής η συνθήκη.

έκφραση1	έκφραση2	έκφραση1 && έκφραση2
false	false	false
false	true	false
true	false	false
true	true	true

Λογικοί τελεστές III

|| (λογικό OR)

- Οποιαδήποτε από τις δηλώσεις που χρησιμοποιούνται στους τελεστές || σε μια συνθήκη πρέπει να είναι αληθής για να είναι αληθής η συνθήκη.

έκφραση1	έκφραση2	έκφραση1 && έκφραση2
false	false	false
false	true	true
true	false	true
true	true	true

Λογικοί τελεστές IV

! (λογικό NOT ή λογική άρνηση)

- Το ‘!’ μπροστά από τη συνθήκη αντιστρέφει την έννοια της συνθήκης.

  Μία τιμή **true** γίνεται **false**.
  Μία τιμή **false** γίνεται **true**.

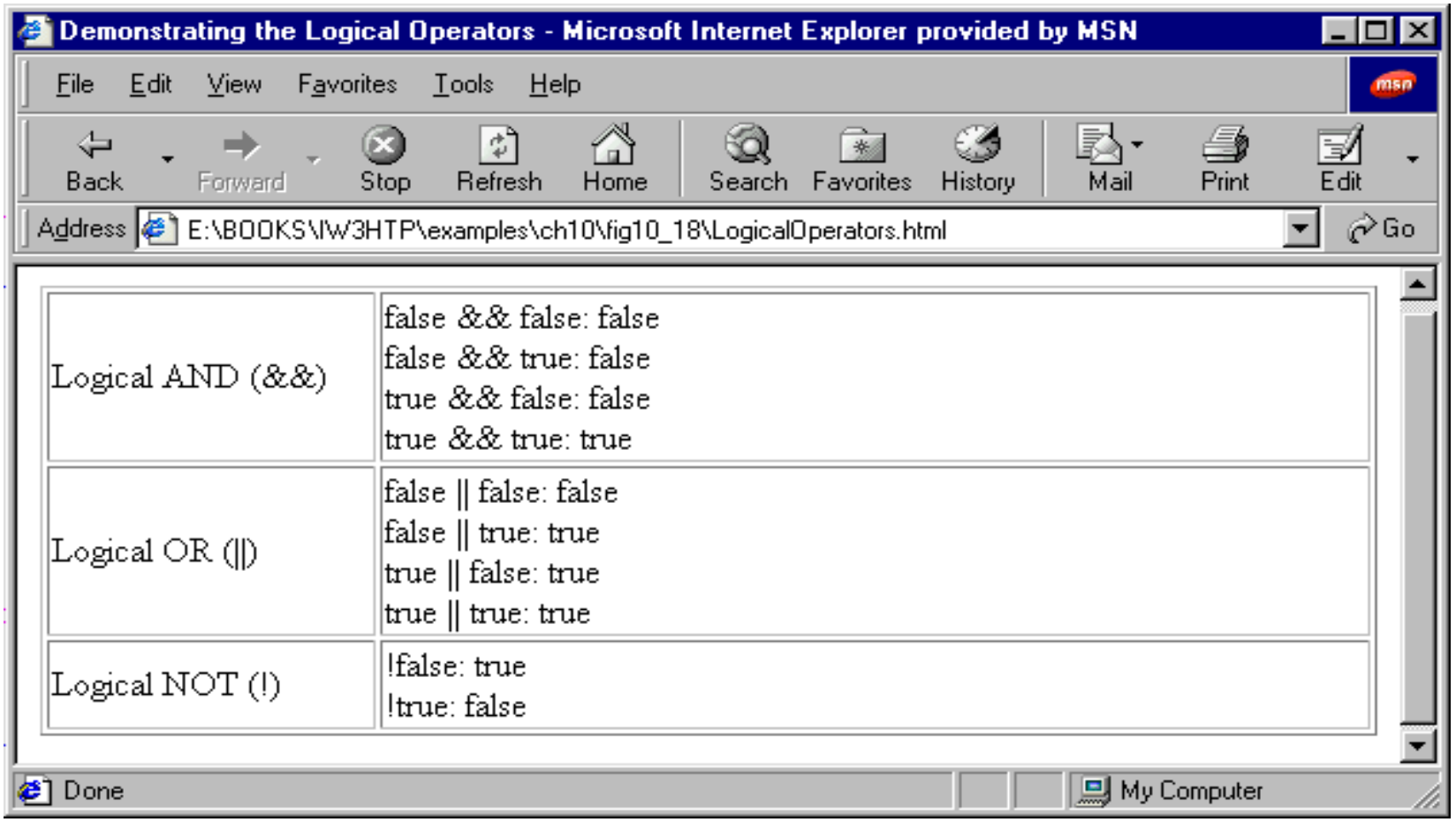
Παράδειγμα χρήσης λογικών τελεστών

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">
<HTML>
<!-- Fig. 10.18: LogicalOperators.html -->
<HEAD>
<TITLE>Demonstrating the Logical Operators</TITLE>

   <SCRIPT LANGUAGE = "JavaScript">
          document.writeln("<TABLE BORDER= '1' WIDTH= '100%'>") ;

          document.writeln(
                    "<TR><TD WIDTH = '25%'> :Logical AND (&&)</TD>" +
                    "<TD> false && false: " + ( false && false ) +
                    "<BR> false && true: " + ( false && true ) +
                    "<BR> true && false: " + ( true && false ) +
                    "<BR> true && true: " + ( true && true ) + "</TD>");

           document.writeln(
                    "<TR><TD WIDTH = '25%'> :Logical OR (||)</TD>" +
                    "<TD> false || false: " + ( false || false ) +
                    "<BR> false || true: " + ( false || true ) +
                    "<BR> true || false: " + ( true || false ) +
                    "<BR> true || true: " + ( true || true ) + "</TD>");
```

Παράδειγμα χρήσης λογικών τελεστών

```
        document.writeln(
                  "<TR><TD WIDTH = '25%'> :Logical NOT (!)</TD>" +
                  "<TD> !false: " + ( !false ) +
                  "<BR> !true: " + ( !true ) + "</TD>");

        document.writeln( "</TABLE>") ;


</SCRIPT>

</HEAD><BODY></BODY>
</HTML>
```

Έξοδος προγράμματος χρήσης λογικών τελεστών



Περίληψη δομημένου προγραμματισμού

- Κανόνες για τη διαμόρφωση δομημένων προγραμμάτων

Ξεκινήστε με το “απλούστερο διάγραμμα ροής”.

Κάθε ορθογώνιο (ενέργεια) μπορεί να αντικατασταθεί από δύο ορθογώνια (ενέργειες) σε σειρά.

Κάθε ορθογώνιο (ενέργεια) μπορεί να αντικατασταθεί από οποιαδήποτε δομή ελέγχου (ακολουθία, if, if / else, switch, do / while ή for)

Οι κανόνες 2 και 3 μπορούν να εφαρμόζονται όσο συχνά θέλετε και με οποιαδήποτε σειρά.

Περίληψη δομημένου προγραμματισμού II

Δομημένη προσέγγιση: 7 κομμάτια μονής εισόδου / μονής εξόδου.

Δομές ελέγχου επιλογής:

- `if` (μονή επιλογή)
- `if/else` (διπλή επιλογή)
- `switch` (πολλαπλή επιλογή)

Δομές ελέγχου επανάληψης:

- `while`
- `do/while`
- `for`
- `for/in` (Κεφάλαιο12)

Περίληψη δομημένου προγραμματισμού III

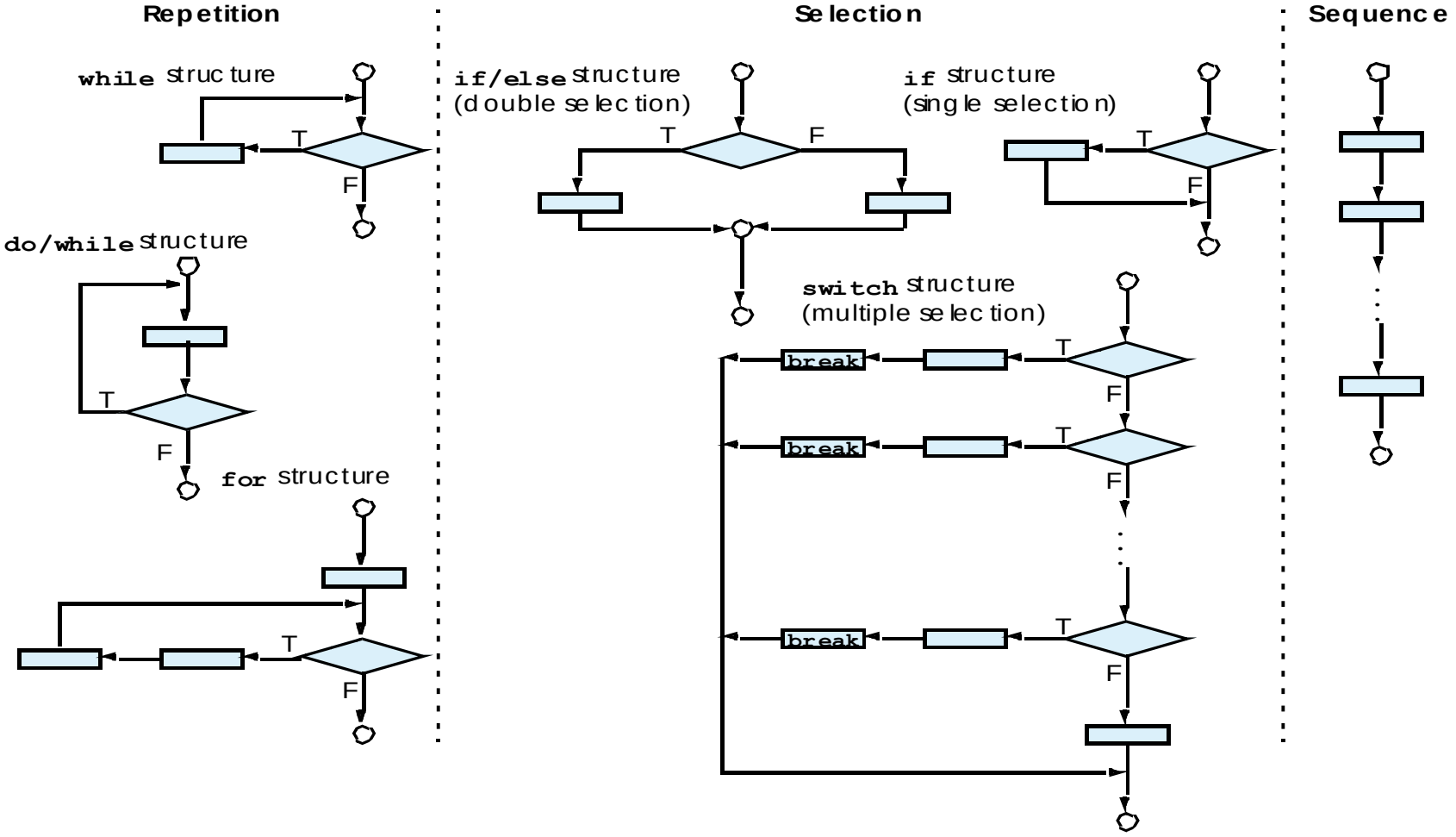
- Οποιαδήποτε μορφή ελέγχου στη JavaScript μπορεί να εκφραστεί μέσω

  - δομής if (επιλογή)
  - δομής while (επανάληψη)

- Οι δομές ελέγχου συνδυάζονται με δύο τρόπους

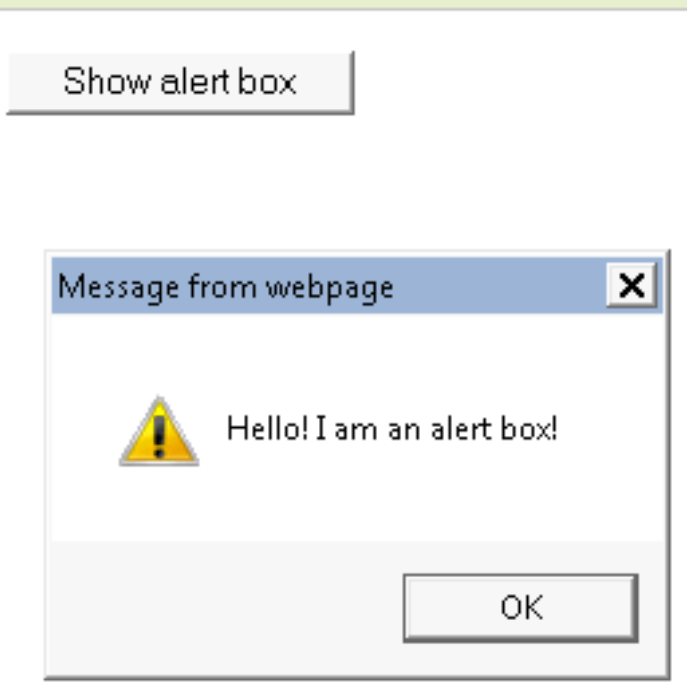
  - Τοποθέτηση σε στοίβα
  - Εμφώλευση

Περίληψη δομημένου προγραμματισμού IV



Alert Box

```
<html>
<head>
<script type="text/javascript">
function show_alert()
{
alert("Hello! I am an alert box!");
}
</script>
</head>
<body>

<input type="button" onclick="show_alert
()" value="Show alert box" />

</body>
</html>
```

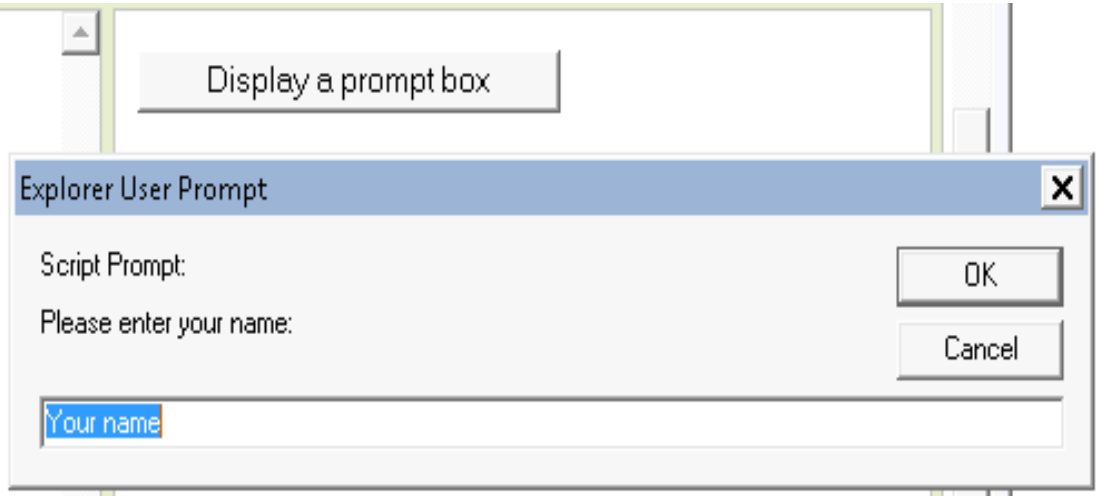


Confirm Box

```
<html>
<head>
<script type="text/javascript">
function show_confirm()
{
var r=confirm("Press a button!");
if (r==true)
  {
  alert("You pressed OK!");
  }
else
  {
  alert("You pressed Cancel!");
  }
}
</script>
</head>
<body>

<input type="button" onclick="show_confirm
()" value="Show a confirm box" />

</body>
</html>
```

Prompt Box

```
<html>
<head>
<script type="text/javascript">
function disp_prompt()
{
var fname=prompt("Please enter your
name:","Your name")
document.getElementById
("msg").innerHTML="Greetings " + fname
}
</script>
</head>
<body>

<input type="button" onclick="disp_prompt
()" value="Display a prompt box" />
<br /><br />

<div id="msg"></div>

</body>
</html>
```



Συνάρτηση και επιστρεφόμενη τιμή

```
<html>
<head>
<script type="text/javascript">
function product(a,b)
{
return a*b;
}
</script>
</head>

<body>
<script type="text/javascript">
document.write(product(4,3));
</script>

<p>The script in the body section calls a
function with two parameters (4 and 3).</p>
<p>The function will return the product of
these two parameters.</p>
</body>
</html>
```

12

The script in the body section calls a function with two parameters (4 and 3).

The function will return the product of these two parameters.

For … in

```
<html>
<body>
<script type="text/javascript">
var x;
var mycars = new Array();
mycars[0] = "Saab";
mycars[1] = "Volvo";
mycars[2] = "BMW";

for (x in mycars)
{
document.write(mycars[x] + "<br />");
}
</script>
</body>
</html>
```

Saab
Volvo
BMW