

# Programming 2

# Last Class

- What is a computer program?
- JavaScript within HTML
- Outputting messages:
  - `document.writeln("my message")`
- Comments
  - `// This is a comment!`
- Variables
  - `var x = 3`
- Arithmetic Operators
  - `*, +, -, /`

# Variables

- What names can we choose for our variables?
  - x
  - 3
  - 3musketeers
  - go4it
  - chocolate cake
  - chocolate\_cake
  - you+me
  - you'nme
  - var
  - my\_favourite\_var

# Variables

- Here are the rules:
  - **cannot** start with a numeral, e.g. 2goats
  - **can** have numbers within variable name, e.g. up2you
  - **cannot** contain a mathematical or logical operators in a variable name
  - **cannot** use punctuation marks of *any* kind, other than the underscore
  - **cannot** contain spaces
  - **cannot** use JavaScript keywords
    - for a complete list of keywords, see <http://www.javascriptkit.com/jsref/reserved.shtml>
  - **can** use keywords as part of variable names, e.g. useless\_var

# Re-assigning Variables

## Example 1:

```
var count
```

```
count = count + 1 // perfectly legal, do this all the time
```

## Example 2:

```
var x = 3
```

```
var y = 5
```

```
// Swap the values of x and y:
```

```
var z
```

```
z = y
```

```
y = x
```

```
x = z
```

# More on Arithmetic Operators

- Modulus operator %
  - $5 \% 3 = 2$  (The remainder of 5 when divided by 3)
  - $3 \% 3 =$
  - $2 \% 3 =$

# String Variables

- A **string** is a type of variable that hold sequences of characters
  - “Hello, world!” is a string
  - “Hello, world #512!” is a string

```
var str1 = "Hello"
var str2 = "Goodbye"
var str3 = "World!"
var str4 = str1 + " " + str3 // prints 'Hello World!'
document.writeln(str4)
var str5 = str1 + ", " + str2 + ". "
str5 = str5 + "You say " + str2 + "; I say "+ str1
document.writeln(str5) // prints 'Hello, Goodbye. You
// say Goodbye; I say Hello!'
```

# Line Breaks

```
var str1 = "Hello"  
var str2 = "World!"  
var str3 = str1 + "<br>" + str2  
document.writeln(str3)  
var str4 = str1 + ", Goodbye. " + "<br>"  
str4 = str4 + "You say Goodbye; I say "+ str1  
document.writeln(str4)
```

# The '+' operator

```
var x = ((2*4)+10)/6 // a numerical expression that
                    // evaluates to 3
var y = "Hello " + "World" // a string expression
                               // that evaluates to
                               // "Hello World"
var z = "Hello " + 3 // ???

// 3 gets automatically
// converted to a string
// z is "Hello 3"
```

# Input from the User

- The prompt command allows to accept input from the user by means of a pop-up. The value entered by the user is returned by the command as a string.

```
var x = prompt("Enter a number:", "") // no default value
var y = prompt("Enter another number:", "10") // default
    // value of 10
x = parseInt(x) // convert x from a string to a number
y = parseInt(y) // convert y from a string to a number
var sum = x + y
document.writeln("The sum of the two numbers is " + sum)
```

- What if don't use `parseInt()`?

# Boolean Variables

- A boolean variable can take on one of two possible values: **true** or **false**

```
var T = true
```

```
var F = false
```

- Boolean variables are produced as a result of **logical** and **comparison** operations

# Logical Operators

- The logical operators are AND, OR, NOT
- Examples:
  - true AND false = false
  - true OR false = true
  - NOT true = false

# AND (&&) Operator

- Written ‘&&’ in JavaScript

- true && true = true
- true && false = false
- false && true = false
- false && false = false

- Example:

```
var x = true
var y = false
var z = true
var a = x && y // a is set to false
var b = z && z // b is set to true
document.writeln(y && z) // 'false' is printed
document.writeln(x && z) // 'true' is printed
```

# OR (||) Operator

- Written '||' in JavaScript

- true || true = true
- true || false = true
- false || true = true
- false || false = false

- Example:

```
var x = true
var y = false
var z = false
var a = x || y // a is set to true
var b = z || z // b is set to false
var c = y || y // c is set to false
document.writeln(x || z) // 'true' is printed
document.writeln(y || z) // 'false' is printed
```

# NOT (!) Operator

- Written ‘!’ in JavaScript

- ! true = false

- ! false = true

- Example:

```
var x = true
```

```
var y = false
```

```
var z = false
```

```
var a = !x // a is set to false
```

```
var b = !y // b is set to true
```

```
var c = !z // c is set to true
```

```
document.writeln(c) // 'true' is printed
```

```
document.writeln(a) // 'false' is printed
```

```
document.writeln(!a) // 'true' is printed
```

# Comparison Operators

- The *comparison operators* compare two values (numbers, or variables).
- There are several comparison operators:
  - $<$ : less than
  - $<=$ : less than or equal to
  - $==$ : equal to
  - $>$ : greater than
  - $>=$ : greater than or equal to
  - $!=$ : not equal to

# Comparison Operators

- **Examples:**

```
var x = 17 > 5 // x is set to true
```

```
var y = x // y is set to true
```

```
var a = 3
```

```
var z = x && (a >= 8) // z is set to false
```

```
var b = (a == 3) // b is set to true
```

```
var c = (z != true) // c is set to true
```

- **How would you use conditional and logic operators?**

# Conditional Statements

- Execute a block of code only *if* a certain statement is true

- Example:

```
var x = prompt("Enter a number between 1 and 100", "")
x = parseInt(x) // convert the input from a string to a
                // number

if (x > 50)
{
    document.writeln("Your number is greater than 50!")
    // this statement is only executed if the
    // boolean expression 'x > 50' is true
}
document.writeln("Thank you for entering that number.")
```

- This block of code is called an *if* block

# Conditional Statements

- If there is an *else* block after the *if* block, then the *else* block is executed if the *if* condition is evaluated to false.

```
if(some expression is true)
{
    do something
}
else
{
    do something else
}
```

Either something or something else happens, but NOT both!!!

# Conditional Statements

```
var x = prompt("Enter a number between 1 and 100", "")
x = parseInt(x) // convert the input from a string to a
                // number

if (x > 50)
{
    document.writeln("Your number is greater than 50!")
    // this statement is only executed if the
    // boolean expression 'x > 50' is true
} else
{
    document.writeln("Your number is smaller or equal to
    50.")
}
document.writeln("Thank you for entering that number.")
```

# Conditional Statements

- An *else if* block allows another condition to be checked if the first one (*if* condition) was evaluated to false.

```
if(some expression is true)
{
    do something
}
else if(some expression is true)
{
    do something else
}
```

Either something, something else or nothing happens, but never more than one thing

# Conditional Statements

```
var x = prompt("Enter a number between 1 and 100", "")
x = parseInt(x) // convert the input from a string to a
                // number
if (x < 1 || x > 100)
{
    document.writeln("You have entered an invalid number!")
} else if (x > 50)
{
    document.writeln("Your number is greater than 50!")
    // this statement is only executed if the
    // boolean expression 'x > 50' is true and if the
    // previous expression is false
} else
{
    document.writeln("Your number is smaller or equal to
    50.")
}
document.writeln("Thank you for entering that number.")
```

# Conditional Statements

- The conditional block can itself contain conditional statements

```
var x = prompt("Enter a number between 1 and 100", "")
x = parseInt(x) // convert the input from a string to a
  // number
if (x > 50)
{
  document.writeln("Your number is greater than 50!")
  if (x < 75)
  {
    document.writeln("Your number is less than 75!")
  }
}
```

- How would you check if the number is greater than or equal to 75?

# Conditional Chains

- You can have as many *else if* blocks per conditional statement as you want, as long as you start with an *if* block
- *else* blocks are always optional. Each *if, else if, else if, ... , else if, else* sequence is called a *conditional chain*.
- **IMPORTANT:** Only **one** of the blocks in a conditional chain is executed: the first block whose condition evaluates to true is executed. The rest, even if their conditions also evaluate to true, are not executed.

# alert and confirm

- **alert** pops up a window with a message:  

```
alert("This is a very important message")
```
- **confirm** pops up a window with a message that you either confirm (click 'OK') or don't confirm (click 'Cancel')  

```
var x = confirm("You are to be my slave")  
if (x == true) alert("You belong to me!")  
else alert("I'll get you next time!")
```

# Loops

- Say you wanted to repeatedly print out “Hello, World!” 5 times to the screen. How would you do it?
- Say you wanted to do it 50 times?
- Say you wanted to do it  $n$  times, where  $n$  is given by the user?

# Loops

```
var i = 1
while (i <= 50)
{
    document.writeln("Hello, World!")
    document.writeln(i + "<br>")
    i = i+1
}
```

- Loops are like conditionals except that **the code inside a loop is repeated, over and over, until the condition becomes false**
- How many times does the above program execute?

# Loops

```
var i = 1
while (i <= 5)
{
  document.writeln("Hello, World!")
  document.writeln(i + "<br>")
  i = i+2
}
```

- How many times is “Hello, World” printed?  
How can you count?

# Loops

```
var i = 1
var count = 0
while (i <= 5)
{
    document.writeln("Hello, World!")
    document.writeln(i + "<br>")
    i = i+2
    count = count + 1
}
document.writeln("<br><br>\nHello,
    World\" printed " + count + " times")
```

# Loops

```
var i = 5
while (i > 0)
{
  document.writeln("Hello, World!")
  document.writeln(i + "<br>")
  i = i-2
}
```

- How many times is “Hello, World” printed?
- Answer: 3

# Loops

- The following program calculates the sum of the numbers from 1 to  $n$ .

```
var n = prompt("Enter a number", "")
n = parseInt(n)
var i = 1
var sum = 0
while (i <= n)
{
    sum = sum + i
    i = i+1
}
document.writeln("The sum of the numbers from 1 to
"+ n + " is "+sum)
```

# Loops

- What does this program do?

```
var n = prompt("Enter a number", "")
```

```
n = parseInt(n)
```

```
var i = 1
```

```
var sum = 0
```

```
while (i <= n)
```

```
{
```

```
    sum = sum + i
```

```
}
```

```
document.writeln("The sum of the numbers from 1 to  
"+ n + " is "+sum)
```

# Loops

- What does this program do?

```
var n = prompt("Enter a number", "")
n = parseInt(n)
var i = 1
var sum = 0
while (i > n)
{
    sum = sum + i
    i = i+1
}
document.writeln("The sum of the numbers from 1 to
"+ n + " is "+sum)
```

# Nested Loops

- You can have a loop(s) inside another loop. What does the following program print out?

```
var i = 1
var j = 1
while (i <= 10)
{
  j = 1
  while (j <= 10)
  {
    document.writeln(i*j)
    j = j+1
  }
  i = i+1
  document.writeln("<br>")
}
```

The 10x10  
multiplication table!

# Functions

- **Functions** allow you to write reusable code
- When a function is invoked, the execution temporarily “jumps” to the body of the function to execute the statements in the function, then comes back to the statement that was being executed
- **Example:**

```
function greeting() // this block is a function
                    // declaration
{
    alert("Hello World!")
} // code inside the function is only executed
  // when the function is invoked/called

greeting() // invoke the function once
greeting() // invoke the function a second time
```

# Functions

- Purposes of functions:
  - **Modular** program design: divide your task into modules:
    - Each function is said to perform a certain operation
    - Can assume each function works correctly while programming the rest
  - **Re-use** of code

# Functions

- Functions can have **parameters**. Use these parameters to *customize* function activity.
- When using a function with parameters, need to specify in brackets the parameters

```
function bdaysong(name)
{
    alert("Happy Birthday to you!")
    alert("Happy Birthday to you!")
    alert("Happy Birthday, dear " + name + "!")
    alert("Happy Birthday to you!")
}
bdaysong("Fred")
```

- "Fred" is assigned to the variable name in bdaysong

# Functions

- What functions do you know that have parameters?

```
alert("This is an important message")  
prompt("Enter a number", "5")  
writeln("Hello, world!")  
confirm("Do you wish to continue?")
```

# Functions

- prompt and confirm *return* values

```
var x = prompt("Enter a number", "")
var keep_going = confirm("Keep going?")
```

- This function computes the sum of two numbers and *returns* the answer.

```
function sum(x, y)
{
    return x+y
}
```

```
var a = sum(3,4) // a is assigned the value 7
document.writeln(a) // prints out 7
```

# Functions

- This function calculates the sum of numbers from 1 to n.

```
function sumoneton (n)
```

```
{
```

```
  var sum = 0
```

```
  var i = 1
```

```
  while (i <= n)
```

```
  {
```

```
    sum = sum + i
```

```
    i = i + 1
```

```
  }
```

```
  return sum
```

```
}
```

```
var a = sumoneton(100) // sets a to 1 + 2 + ... + 100
```

```
var b = sumoneton(150) // sets a to 1 + 2 + ... + 150
```

- Before we knew about functions, how would we do these computations?

# String Manipulation

- Strings (e.g. “Mary had 3 little lambs”) are important in programming. Very often, you may want to find out
  - the length of a string
  - the character in the string at a certain position
  - convert all the characters in a string to UPPER or lower case
- Nice programmers made it *easy* for us to perform these operations on strings

# String Manipulation

- By placing the ‘.’ character after a string followed by special words reserved for strings, can perform operations, or find out properties of the string
- String length:

```
var str = "Yahoo!!!"
```

```
var len = str.length
```

```
document.writeln(len) // displays 8
```

```
str = "Hello, how are you?"
```

```
len = str.length
```

```
document.writeln(len) // displays 19
```

# String Manipulation

- You can find the character in a string at position  $i$ , using `.charAt(i)`
- The first character of the string is at position **0**

```
var str = "Hello, World!"
var len = str.length
var a = str.charAt(0) // sets a to "H"
var b = str.charAt(1) // sets b to "e"
var c = str.charAt(5) // sets c to ","
var d = str.charAt(len-1) // sets d to "!"
document.writeln(a+b+c+d) // displays "He,!"
```

# String Manipulation

- Following a string *a* with `.toUpperCase()` makes another string *b* that has the same letters as *a* but uses only capital letters
- Analogously for `.toLowerCase()`

```
var str = "Hello, World!"
var str1 = str.toUpperCase()
document.writeln(str1) // displays HELLO,
// WORLD!
var str2 = str.toLowerCase()
document.writeln(str2) // displays hello,
// world!
document.writeln(str2.toUpperCase()) //???
```

# String Manipulation

- This program writes the letters in a string one per line

```
var str = "Hello, World!"
var len = str.length
var i = 0
var c
while (i < len)
{
    c = str.charAt(i)
    document.writeln(c + "<br>")
    i = i + 1
}
```

# String Manipulation

- This function returns the reverse of a string.

```
function revstr(s)
{
  var c
  var newstr = "" // !!!
  var i = s.length - 1
  while (i >= 0)
  {
    c = s.charAt(i)
    newstr = newstr + c
    i = i - 1
  }
  return newstr
}
document.writeln(revstr("Hello World!"))
// displays !dlroW olleH
```

# String Manipulation

- This function also returns the reverse of a string.

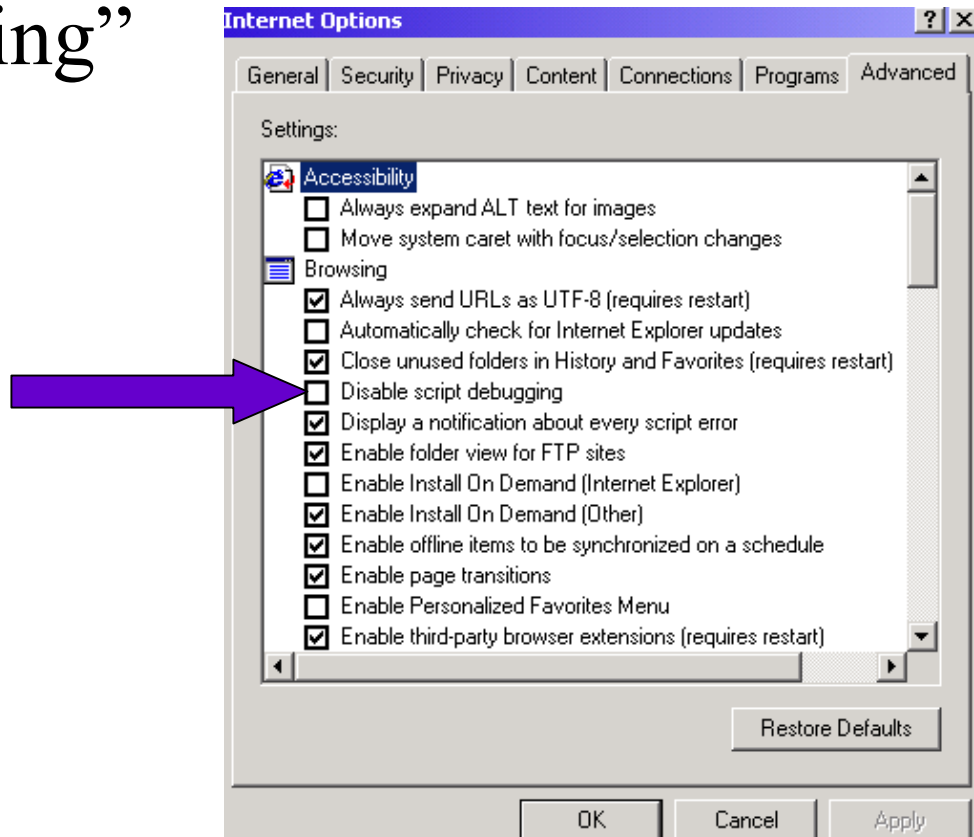
```
function revstr(s)
{
    var c
    var newstr = "" // !!!
    var i = 0
    while (i <= s.length - 1)
    {
        c = s.charAt(i)
        newstr = c + newstr
        i = i + 1
    }
    return newstr
}
document.writeln(revstr("Hello World!"))
// displays !dlroW olleH
```

# Tips for writing your code

- If you copy and paste code, make sure that all the quotations are indeed “”.
- Debugging ideas:
  - Use `document.writeln()` statements to write the value of your variables to make sure that they store the correct value
  - Comment out sections of code that you don't want to be executed
  - Build your code *incrementally*

# Enable JavaScript Debugging in Internet Explorer

- Tools → Internet Options → Advanced and then *uncheck* the box that says “Disable script debugging”



Don't need to know  
for exam!

# Guessing Game

Need to know  
for exam!

```
var rand = Math.random() // gives a random number between 0 and 1
var num = Math.round(rand*100) // num is an integer between 0 and 100
var max_guesses = 10 // maximum number of guesses allowed
var num_guesses = 1
var win = false
while (num_guesses <= max_guesses && !win)
{
    var guess = prompt("Pick a number between 0 and 100", "")
    guess = parseInt(guess)
    if (guess > num) { alert(guess + " is too high!") }
    else if (guess < num) { alert(guess + " is too low!") }
    else if (guess == num) { win = true }
    num_guesses = num_guesses+1
}
if (win) { document.writeln("You win! :)") }
else { document.writeln("You lose :(") }
```