

IT452 Advanced Web and Internet Systems

Set 1: Review of Key Concepts

Exercise #1 – Correct any invalid XHTML syntax

```
<?xml version = "1.0" encoding=utf-8 ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
  "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<!-- An example file
<!-- Our first Web page --&gt;
&lt;html xmlns = "http://www.w3.org/1999/xhtml"&gt;

&lt;body&gt;
  &lt;h1&gt; Welcome to &lt;b&gt; IT350! &lt;/h1&gt; &lt;/b&gt;
  &lt;h2&gt; Today's Agenda &lt;/h2&gt;
  &lt;li&gt; XHTML
  &lt;li&gt; JavaScript
&lt;/body&gt;</pre>
```

Exercise #2 -- What's the output?

```
var a, b, c;  
  
a = 1;  
b = 2;  
c = 3;  
  
d = a + b * c;  
  
window.alert("<h1>Begin</h1>");  
  
if (d < 20)  
    window.alert("d is okay: "+d);  
else  
    window.alert("d is too high!:"+ d);  
d = d - 3;  
  
document.writeln("<h1>Done. Final d = "+d+"</h1>");
```

Exercise #3 -- What's the output?

```
var x, y, z;  
  
x = 7;  
y = 9;  
z = "abc";  
  
window.alert(x+y+z);  
  
window.alert(z+y+x);  
  
if (x)  
    window.alert("x true");  
  
x = "seven";  
  
window.alert(x+y+z);
```

JavaScript Scope Rules

- Variables declared inside a function:
 - Explicitly (with var)
 - Implicitly (just used)
 - Parameters

(Look at FIRST USE inside a function to decide which applies)
- Variables declared outside a function:
 - Explicitly
 - Implicitly

Exercise #3 – Write a function that takes an array and returns the minimum of the array

Exercise #4 – What's the output?

```
function fun1 (x) {
    x = x + 3;
    y = y + 4;
    document.writeln("<br/> FUN1: "+x+", "+y);
}

function fun2 () {
    var y;
    x = x + 10;
    y = y + 20;
    document.writeln("<br/> FUN2: "+x+", "+y);
}

x = 1;
y = 2;

document.writeln("<br/> MAIN #1: "+x+", "+y);
fun1(x);
document.writeln("<br/> MAIN #2: "+x+", "+y);
fun1(y);
document.writeln("<br/> MAIN #3: "+x+", "+y);
fun2();
document.writeln("<br/> MAIN #4: "+x+", "+y);
```

Exercise #5 – Change this code to make the <p> element have a bigger font when you move the mouse over it.

```
<html xmlns = "http://www.w3.org/1999/xhtml">
<head>
    <title>Bigger</title>
    <script type = "text/javascript">

        </script>
    </head>
<body>

    <p>
        Welcome to my page!
    </p>

</body>
</html>
```

Exercise #6 – Modify so that clicking on the button changes target of <a> element to “dog.html”

```
<html xmlns = "http://www.w3.org/1999/xhtml">
  <head>
    <title>Change Link</title>
    <script type = "text/javascript">

      </script>
    </head>
  <body>
    <a href="cat.html" >
      See some animals!
    </a>

    <form action=""> <br/>
      <input type="button" value="Change animal" />
    </form>
  </body> </html>
```

Perl Stuff

“Scalar” variables:

```
$x = 3;
$y = "Hello";
```

“Array” variables:

```
@list = (3, 7, "dog", "cat");
@list2 = @list1;      # copies whole array!
```

A single element of an array is a “scalar:

```
print "Second item is: $list[1]";  # Don't use @
```

Get array length by treating whole array as scalar:

```
$lengthOfList2 = @list2;
```

File operations

```
open ( MYFILE, "input.txt" );
open ( MYFILE, ">output.txt" );
open ( MYFILE, ">>LOG.txt" );
```

Perl Basics

```
use CGI qw( :standard );
print( header() );

$x = 2 + 3;
$y = $x * 4;

if ($x == 5.0) {
    print ("x is five");
}

for ($i = 0; $i < 3; $i++) {
    $squared = $i * $i;
    print ("<br> $i = $i, squared is $squared");
}

$pet1 = "dog";
$pet2 = "ll" . "ama";

# Single quotes vs. double quotes
print ("<br/>I have a $pet1 and a $pet2.");
print ('<br/>I have a $pet1 and a $pet2.');

$compl = ($pet1 eq "dog");
print ("<br/> compl: $compl");
```

Perl Function Calls (“subroutines”)

```
use CGI qw( :standard );
print( header() );

# Prints "hello", takes no arguments
sub hello {
    print "\n<br/> Hello.";
}

# Takes two arguments, return their product
sub multiply {
    my($valA, $valB) = @_;
    return $valA * $valB;
}

my($x) = 2;
hello();
print "\n<br/> $x * 7 = " . multiply($x,7);
hello();
hello(72145);

print(end_html());
```

Function Calls and Arrays

```
# Takes an array as argument, returns minimum value
sub findMin {
    my(@array) = @_;
    my $min = $array[0];
    my $ii;
    my $len = @array;
    for ($ii=0; $ii < $len; $ii++) {
        if ($array[$ii] < $min) {
            $min = $array[$ii];
        }
    }
    return $min;
}

# Defines new global array, @array1
# AND returns a new array with 4 elements.
sub makeArray() {
    @array1 = (89, 23, 90);
    my @array2 = (34, 5.4, 123, 2.01);
    return @array2;
}

@test1 = makeArray();
@test2 = (89, 23, 40, -17);
print "\nMin1 is: " . findMin(@test1);
print "\nMin2 is: " . findMin(@test2);
print "\nMin3 is: " . findMin(@array1);
print "\nMin4 is: " . findMin(@array2);
```