php basics

- Hey folks, you've found the basic php short for the web development course at NC State's computer science department. I'm Ben Watson, director of the Visual Experience Lab there.
    - (php basics)
    - (web development course)
    - (Ben Watson)
    - (VXLab)
- In this short, we'll talk about php as a language.
        - (php as a computer language)
    - We're assuming that you've learned another program language well, like C or Java.
        - (you know: c, java or similar language)
- Before we start, I just wanted to let you know the sources I relied on in putting together these shorts:
    - First, php's online documentation, which is excellent
        - (php documentation)
    - Next, the book php and mysql, novice to ninja, by Yank and published by O'Reilly
        - (php novice to ninja, Yank)
    - Finally, the book Learning PHP, MySQL… by Nixon and published by O'Reilly. It is available online in nc state's library.
        - (learning php, mysql, by Nixon)
- First some basic syntax
    - all instructions end in semicolon
        - ($next += 10;)
    - all variables begin with dollar $
        - (begin all variables with $)
- We'll begin with values and types.
    - php is dynamically typed, meaning declarations are not necessary
        - (dynamically typed)
    - There are only three types in php:
        - numbers, e.g. 3 or 3.14159
        - strings, e.g. "Berners-Lee" or 'Berners-Lee'
        - booleans, e.g. true or false
    - Because of the freewheeling nature of dynamic typing, typecasting can be very useful.
        - There are both clike typecasting
            - ((int))
        - And specific functions for casting
            - ((intval))
- To create a constant, use the define variable
    - (define("MYCONST","the value"))
- As to scoping,
    - Global variables are prefaced with the keyword global
        - (global $myglobal;)
    - Local variables are declared inside a function
        - (function myfunc($myparam) {

                - $mylocal = 1;
            - })
        - Static are similar but prefixed with “static”
- Operators are pretty standard
    - +, -, *, / do what they usually do
    - . is the concatenation operator
        - If you mix strings and numbers, javascript treats the numbers like strings. For example,
            - ($outputthis = ‘the number is ‘ . $myNumber . ‘, okay?’;)
        - or a shorthand
            - ($outputthis = “the number is $myNumber, okay?”;)
    - We also have .+=, +=, -=, *=, /=, ++, and --.
- Arrays work like this.
    - To initialize:
        - ($myArray = array(1,2,3,4);)
    - Note that with dynamic typing, you can freely mix types:
        - $myArray = array(‘string’, 1, false);
    - To access values in an array:
        - ($myVar = $myArray[0];)
        - or $myVar = $myArray[$myotherVar];
    - And to put values in an array:
        - there is the standard way:
            - ($myArray[1] = $myVar;)
        - and to add to the end:
            - ($myArray[] = $myVar;)
    - There are also associative arrays, that is keyword value pairs
        - To initialize
            - ($myArray = array(
                - ‘first’ => 1,
                - ‘second’ => 2);)
        - To access
            - ($myVar = $myArray[‘first’];)
        - To put values in
            - ($myArray[‘first’] = $myVar;)
        - By the way, php includes a sort
- Comments are also pretty standard
    - // this is a comment
    - /* this is also a comment */
    - /* as are
    - these lines */
- Time for conditionals.
    - If and else work as you would expect
        - (if () {})
        - (if () {} else {})
    - there is also an elseif statement
    - These comparison operators should work largely as you expect
        - ==, !=, >, <, >=, <=
    - But there are also these two operators that also require type equivalence

        - === and !==
        - Without these, '2' equates to both '2' and 2!
        - Values from most forms are strings, so you will probably want to avoid these operators for those.
    - To combine conditions, use these operators as you normally would
        - AND, OR, XOR and !.
- As to looping, php has
    - The while loop, which is the same as c:
        - while ($myothervar < 10) {
            - each $myothervar;
            - $myothervar++;
        - }
    - and the for loop, again same as c:
        - for ($myothervar=2; $myothervar<10; $myothervar++) {
            - echo $myothervar;
        - }
    - and the do while loop (sometimes called repeat):
        - do {
            - echo $myothervar;
            - myothervar++;
        - } while $(myothervar < 10);
- On to functions.
    - They are untyped in php, which gives them a simpler syntax than C or Java:
        - function myFunc($param1,$param2) {
            - return $param1 . ", " . $param2;
        - }
    - Function names are case insensitive
    - Parameters are passed by value
        - If you want to return multiple values
            - return an array
            - pass by reference
                - just put & in the parameter list
                - you don't have to dereference these outside or inside the function
            - use globals
    - Some notes
        - Functions can be called even when declared after the call itself
        - Recursion works
    - There are many built in functions, which are always available
        - like echo or print