Chapter 1 Fast track to PHP Programming

What is programming? Have you ever been given a list of things to do or looked at instructions? Well that is what programming is, only the list of things to do or instructions are written in a way that a computer can understand them. We will be exploring the PHP [https://en.wikipedia.org/wiki/PHP] programming language. For us the main point to remember is what PHP does for us. In a nutshell PHP creates HTML code. What is a fast track? It means we will cover only what we need just before we need it. The goal is to gain a general understanding not a complete understanding. The complete understanding comes with time.

Computers only do four things:

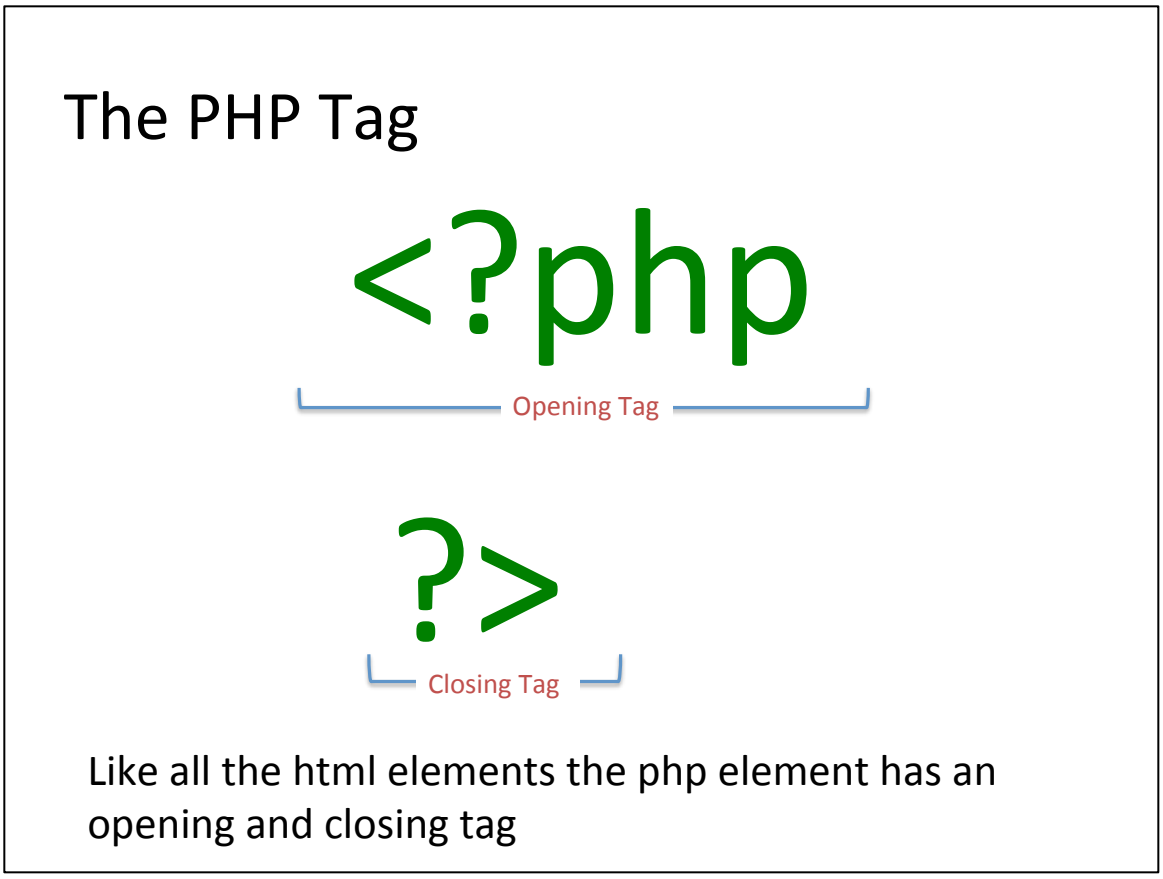
1. **Input** – Accept input into the computer via keyboard, mouse, touch screen, voice commands.
2. **Process** – A program will process the Input it has been given in some fashion from simple addition to determining your current location based on the gps coordinates it receives.
3. **Output** – The computer provides us with output, generally to our computer screen, printed on paper or speakers to name a few.
4. **Storage** – The computer allows us to save information, documents, pictures even your contact list are all examples of storage.

Even more amazing is computers do all these things by understanding if an electrical switch is turned on or off! From a basic sense it is the series of switches that make up what is called a byte and it is the pattern of how they are turned on and off dictates what they mean. For example Zero is used to represent off and One is used to represent on. So this:

0100 0001
[https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0ahUKEwiDuJ-R_4bUAhVi_4MKHWWEDY0QFggoMAA&url=https%3A%2F%2Fwww.jpl.nasa.gov%2Fedu%2Fpdfs%2Fphases_chart.pdf&usg=AFQjCNGoOlgxYfkqpkDiRox9ETB8AzrMQw&sig2=a3JDZ1s_An9ufSjZSRUAAw]

represents the capital letter A when using ASCII text
[https://en.wikipedia.org/wiki/Wikipedia:ASCII]. The computer looks up in a table what that represents then display the character it is told to. Pretty amazing how many electric switches are turned on and off in just a simple text message!

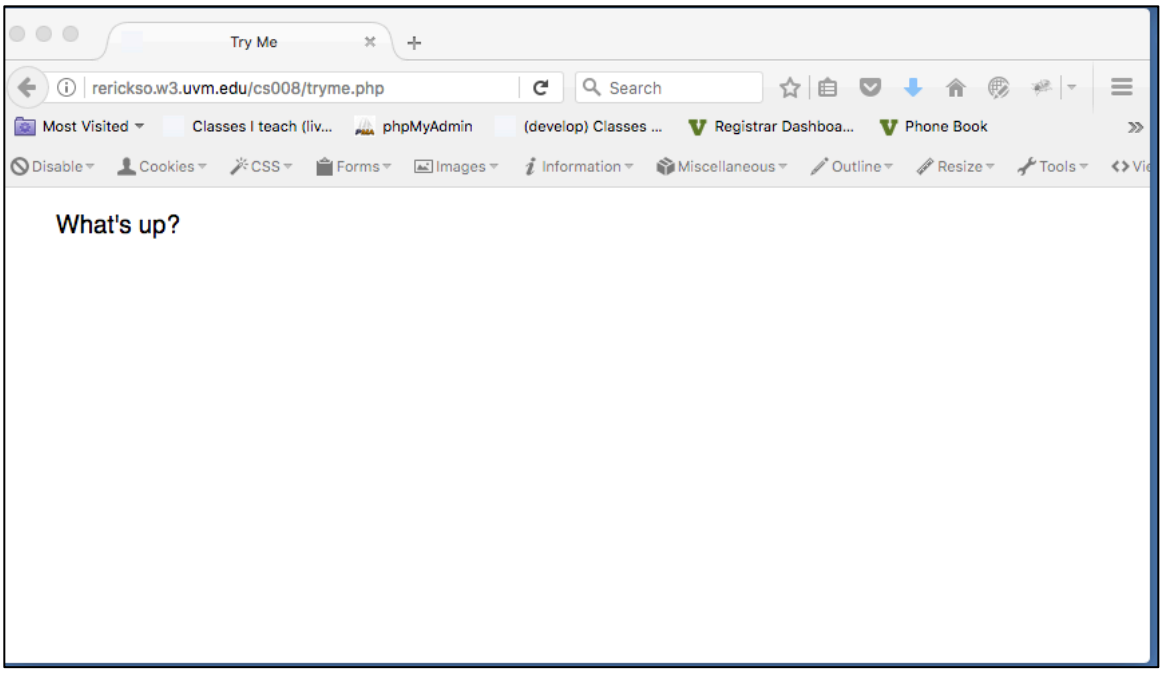
So everything a computer does is because someone has written a series of instructions to tell the computer what to do. PHP is considered a high level language where we don't have to worry about the ones and zeros but instead we learn the syntax and structure of the programming language.



We put PHP instructions inside the PHP element and the end result will be HTML. For example (meaning a useless example other than to demonstrate the point) we could have PHP say hello (old fashioned first program would be Hello World) or in the case of the example below "What's up?".

```
<!DOCTYPE html>
<html>
    <head>
        <title>Try Me</title>
        <meta charset="utf-8">
        <meta name="description" content="just a testing file">
    </head>
    <body>
<?php

print '<p>What\'s up?</p>';

?>
    </body>
</html>
```

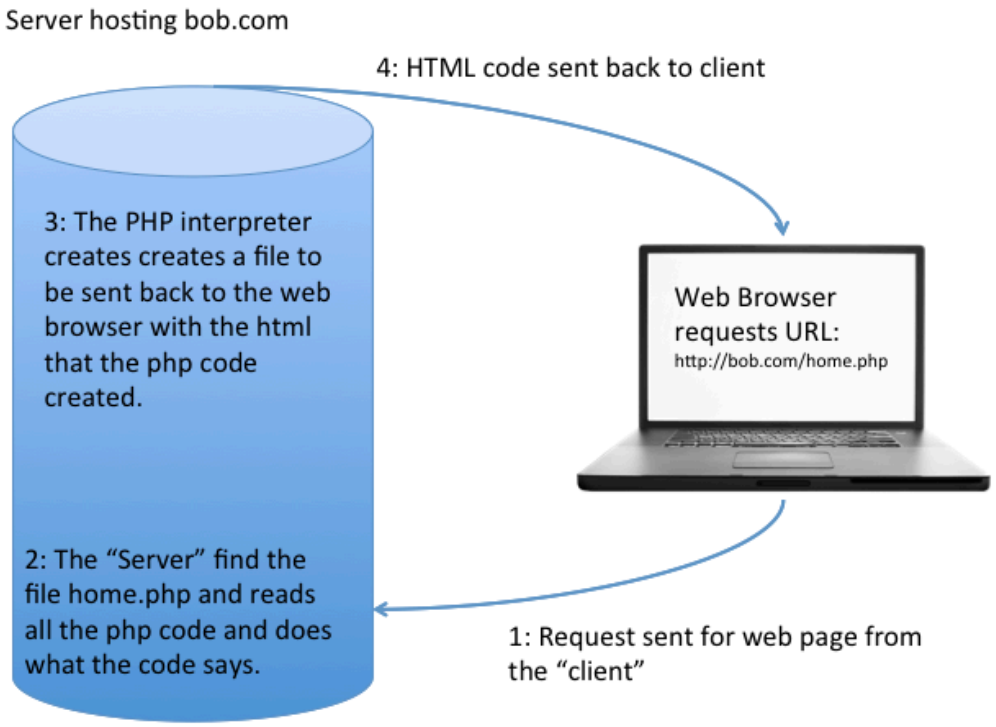
This is how it will look in the browser:



and more importantly if we look at the browsers source code this is what we see.

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <title>Try Me</title>
5          <meta charset="utf-8">
6          <meta name="description" content="just a testing file">
7      </head>
8      <body>
9  <p>What's up?</p>    </body>
10 </html>
```

So it may be a really simple example and not very practical but there is a lot going on. Let's start with the PHP word **print** which is built into PHP and just causes whatever comes after it to become *part of the HTML code*. Ok I need to take a step back and describe what happens when you request a PHP page in the URL.

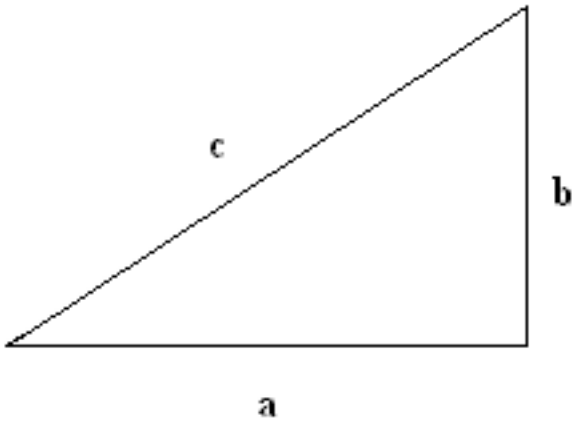


Let's follow the steps:

1. You type a url into your browser window, in the above example http://bob.com/home.php. The browser then goes to the server that hosts that web site.
2. The server then looks for the file home.php (just like you look for files on your computer). When it finds the file it notices it has a .php extension and says woohoo a PHP file (ok maybe it does not actually say that). The server then reads the entire contents of the file executing each line of PHP code one at a time.
3. The PHP interpreter creates a file (still called home.php) that has nothing but HTML code in it that looks just like the browsers source code above. So it executes each PHP instruction that you have given it.
4. The server takes the output file home.php that has nothing but HTML in it and sends it back to the "clients" computer (your computer) so that your web browser can display it.

Ok pretty neat but let's face it, if that is all PHP can do we might as well just write the HTML directly and save ourselves the trouble. I am hoping though that little example makes sense and of course we are going to have more detailed instructions but we have to learn to crawl before we can run.

All programs need values or input that we have to pass into the program in some fashion. Let's just start with variables. Let us start with math and the **Pythagorean theorem [http://www.basic-mathematics.com/pythagorean-theorem.html]** which states:



$$c^2 = a^2 + b^2$$

notice I wrote it backwards since that is how we assign a value in programming versus the math class way of $a^2 + b^2 = c^2$. Scared yet? Don't worry. In Building Trades I learned this as the 3,4,5 triangle to see if the building's corner is square.

[https://www.google.com/maps/search/ocean+county+vocational+school/@39.9836687,-74.1876238,771a,35y,270h,39.28t/data=!3m1!1e3] Technically I helped build the wall from the corner to the door but we had to be square or the door would not work (glad to see its still standing from 1979)

The way I learned it was that if you measure 4 feet on one side (a) and put a pencil mark there. Next measure 3 feet on the other side (b) and put a pencil mark there. Now measure the distance between the two pencil points as long as your building is square it will measure 5 feet. Ok so let's program 3 and 4 into the formula in PHP to see if c truly is five as my shop teacher taught me.

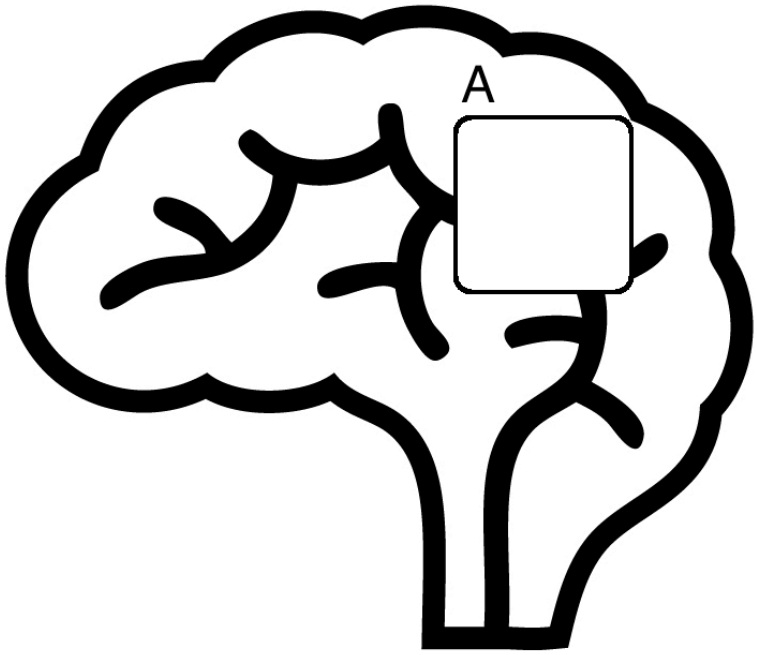
```
<!DOCTYPE html>
<html>
    <head>
        <title>Try Me</title>
        <meta charset="utf-8">
        <meta name="description" content="just a testing file">
    </head>
    <body>
<?php

$a = 4;
$b = 3;
$c = sqrt(($a * $a) + ($b * $b));

print '<p>The ' . $b . ' ' . $a . ' ' . $c . ' triangle is correct!</p>';

?>
    </body>
</html>
```

and we upload our file to the server and this is what shows up in the browser!

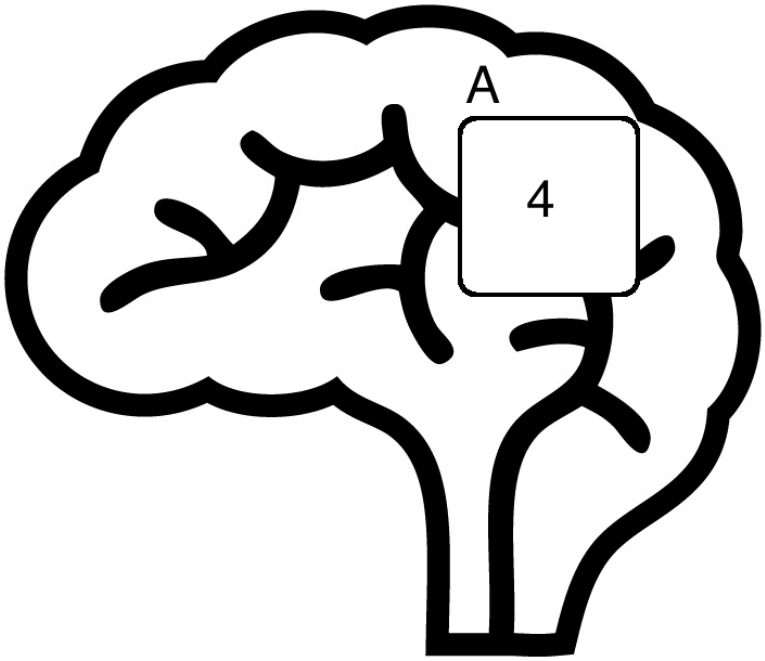
Ok I have never used the square root method [http://php.net/manual/en/function.sqrt.php] built into PHP so I had to google it. Clear as mud? Well let's try to clear things up a bit. The PHP line:

```
$a = 4;
```

is called an assignment statement to be official. The letter a is called a variable. All variables in PHP begin with a $ sign (pretty much the only language I know of that uses such a convention). What happens when when a programming language creates a variable is that it makes a box in the computers memory and gives that box the name you told it to.



Once PHP has the box it will take whatever is to the right of the equal sign and put that value in the box, be it a number, word, image whatever. In our example the number 4



The computer will do this with the variable b as well. It does the same thing for the variable c only it will calculate the value first following the standard rules for math (PEMDAS) [http://www.coolmath.com/prealgebra/05-order-of-operations/05-order-of-operations-parenthesis-PEMDAS-01]

Now let us explore that PHP method called print which will print to the file whatever comes after it. I have a lot of things to explain. I have used single quotes. You can use double quotes however to be consistent I try to use single quotes for PHP and double quotes for HTML. Notice in my first example I printed

```
print '<p>What\'s up?</p>';
```

Ok the outside single quotes make sense as the print method will print whatever is inside the quotes. Since I wanted to print a single quote inside of single quotes I had to 'escape' the quote with a \ backslash which means ignore the quote and just print it. Just to make life confusing I could have used double quotes on the outside and it would produce the same HTML output.

```
print "<p>What's up?</p>";
```

I don't need to escape the single quote in this case. Either way works just remember that whatever you start and end with cannot be on the inside. To really mess with your brain I could do this:

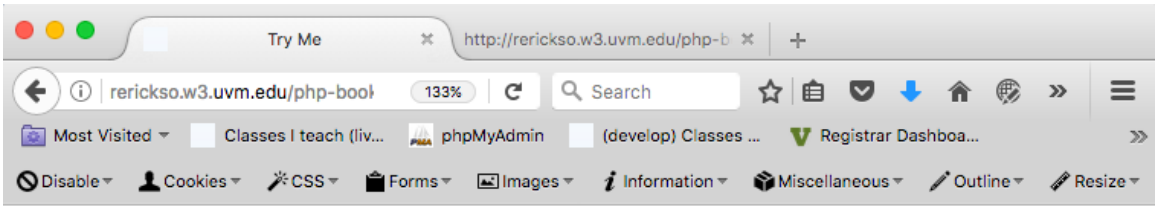
```
print '<p>What' . """ . 's up?</p>';
```

The inner block with five quotes is really a double quote, singe quote, double quote but it follows the rules I described above. Programming can be funky at times especially when it comes to quotes.

What is up with the periods by the way? In PHP the periods are used to add text together which we call concatenation. Lets look at an example.

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <title>Try Me</title>
5          <meta charset="utf-8">
6          <meta name="description" content="just a testing file">
7      </head>
8      <body>
9  <?php
10
11 $a = 4;
12 $b = 3;
13 print '<p>' . ($a . $b) . '</p>';
14 print '<p>' . ($a + $b) . '</p>';
15 ?>
16     </body>
   </html>
```

Line 13 will concatenate 4 and 3 together as text so it will display as 43. Line 14 will add 4 and 3 as numbers so it will display 7.



43

7

Let's look at the source code so we can see what PHP created.

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <title>Try Me</title>
5          <meta charset="utf-8">
6          <meta name="description" content="just a testing file">
7      </head>
8      <body>
9  <p>43</p><p>7</p>    </body>
10 </html>
```

Notice that all the PHP created HTML is on line 9 even though in the code they are on separate lines. The reason is that we never told PHP to put a line break in. Computers only do what they are told and we never told PHP to start a new line. You don't need to as new lines are for humans but if you wanted to we can do that by printing a \n. Of course this makes the use of quotes even more confusing. Here I used single and double quotes.

```
print '<p>' . ($a . $b) . '</p>' . '\n';
print '<p>' . ($a + $b) . '</p>' . "\n";
```

Let's look at the results.

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <title>Try Me</title>
5          <meta charset="utf-8">
6          <meta name="description" content="just a testing file">
7      </head>
8      <body>
9  <p>43</p>\n<p>7</p>
10     </body>
11 </html>
```

The end of line mark needs to be printed in double quotes or it does not work as intended. Notice the closing body element has moved to the next line but the \n just printed on our page. There is not much point in wondering why it (printed \n when in single quotes) is like this but it is what it is. It is a good example of when we may not understand why it is like that but we understand what it does and we can accept that.

You can use a built in PHP **constant** called PHP_EOL which will not require quotes and which is just the best thing to do.

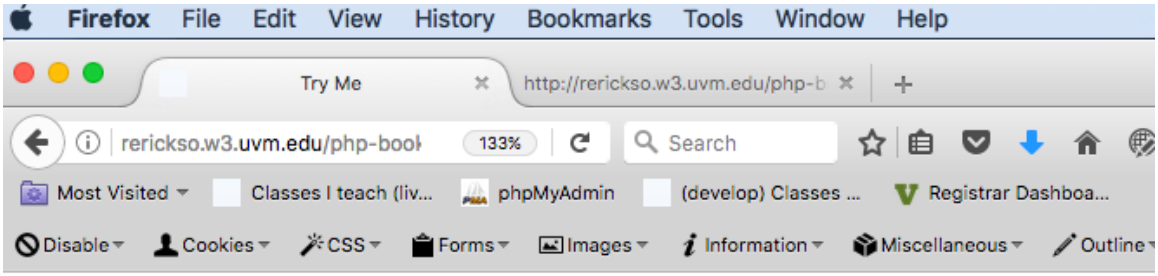
```
print '<p>' . ($a + $b) . '</p>' . PHP_EOL;
```

I only mentioned all this as a way of explanation and introduction. The only time I have PHP put in the end of line break is when I am having trouble with my code and I cannot figure it out. It makes it easier to see what the HTML code looks like that PHP created.

So we have variables in PHP which can be numbers but they can also be strings (text, words, letters etc.). For example I have my heading text in a variable.

```
<!DOCTYPE html>
<html>
    <head>
        <title>Try Me</title>
        <meta charset="utf-8">
        <meta name="description" content="just a testing file">
    </head>
    <body>
<?php

$heading = 'Welcome to the Jungle';
print '<h1>' . $heading . '</h1>';

?>
    </body>
</html>
```

This will print the text inside the heading one. As a reminder this is a simple example and not very practical but I am using it to demonstrate a point. However having said that if you use a framework to create your web site this is pretty much what they do.



Welcome to the Jungle

All right we have some basics down:

1. PHP creates HTML.
2. Variables hold values in the computer's memory.
3. Variable names all begin with a $ sign.

4. Variable names cannot contain blank spaces or special characters other than the underscore (ok this one is new) and should be self documenting meaning I should not have to ask what the variable is for.
5. Variables can contain numbers, text and a bunch of other things.
6. Text is called a string.
7. Quotes are funky, generally in PHP we try to use single quotes around all string variables.
8. You can have single quotes on the outside with double quotes on the inside.
9. You can have double quotes on the outside with single quotes on the inside.
10. If you need to break the above quote rules you will have to escape the quotes on the inside with a \ character.
11. A period is used to concatenate text together.
12. A plus sign will add two numbers (we can guess about -,*, /).
13. To add line breaks in our source code we need to print PHP_EOL.

Let's jump ahead. There are three main programming structures in all programming languages:

Sequence – each line of code is executed one after the other in order.

Selection – allows us to choose which lines of code get executed meaning we can skip lines.

Repetition – allows us to keep doing the same lines of code over and over again as many times as we need to.

Let's start Sequence, pretty much all code is done in the order that it appears. Computer don't jump around in their list of tasks they need to do. So these three lines of PHP code just prints out the three lines of HTML in the same order. Since we have not printed a EOL (end of line) mark they will be all in one line but the browser will display them separately (give it a try).

```
print '<h1>This is a heading</h1>';

print '<p>This is a paragraph.</p>';

echo '<p>You can print or echo the text you want to display</p>';
```

This will show up on one line (ok in this document it word wraps but when you view the page source it will not wordwrap) like this in the browsers source code:

<h1>This is a heading</h1> This is a paragraph. You can print or echo the text you want to display

I want to talk about the Repetition programming structure which will repeat lines of code till you tell it to stop. Let me show you the PHP syntax for a for loop:

```
for(variable; condition; increment){

// Do these instructions

}
```

Notice the double slashes are PHP version to have comments. You can have as many instructions as you need. Let me add sample code that will print that line 100 times:

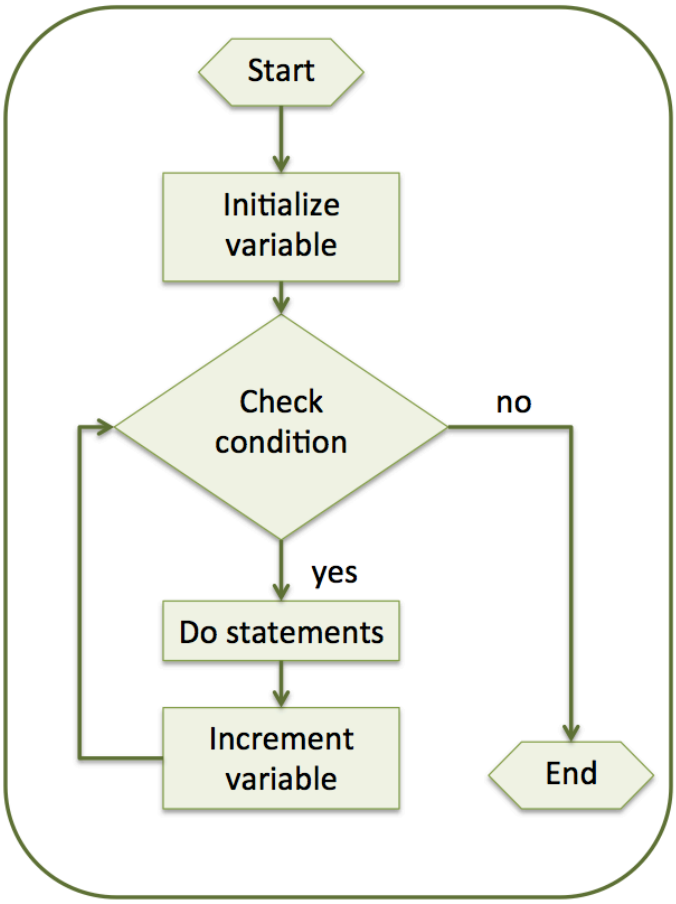
```
for($i=0; $i<100; $i++){

    print '<p>I will not talk in class.</p>';

}
```

Let's look at a flow chart, which I think you can follow along with. The Square boxes represent a line of code that is doing something. The diamond shape represents a Boolean (Boolean is fancy for a yes or no question) question you have to answer. The lines show you which way to go next.

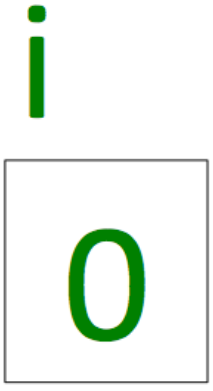
For Loop Flow Chart

```
for(variable, condition, increment){
    // Do these instructions
}
```

```
for($i=0; $i<100; $i++){
    print "<p>I will not talk in class</p>";
}
```



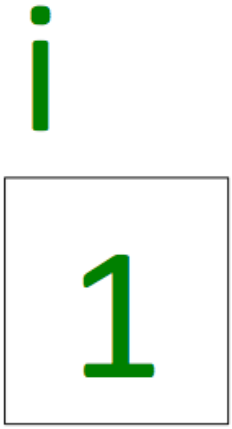
We start by initializing our variable $i to zero. This means we will start with the number zero and go from there. Normally a one letter variable name is bad but $i is a common one often used in loops for index. Other common loop variables are called j, k, and c (j because it comes after i, c for count and k for kount). So when we initialize a variable the computer creates a box in its memory and puts the value in it.



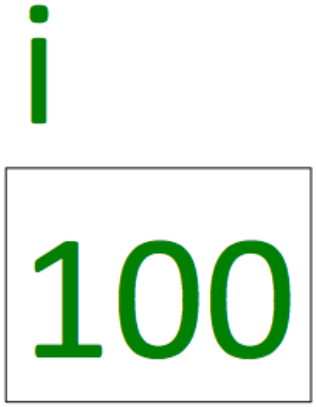
Now we answer the question. Is i less than 100? Since the answer is yes we go into the loop and do the statements found there. In this case we have just one:

I will not talk in class.

We now go to the next box which says to increment the value. We have $i++ which is a shortcut for $i = $i + 1.



Now we answer the question again. Is i less than 100? Since the answer is yes we go into the loop and do the statements again. We keep doing this until i becomes 100. So when i is 99 we do the statements then we increment i by 1 to 100



Now we answer the question again. Is i less than 100? Since the answer is no we don't go into the loop and we move on to the next statement after the loop.

Can you see what this loop would do?

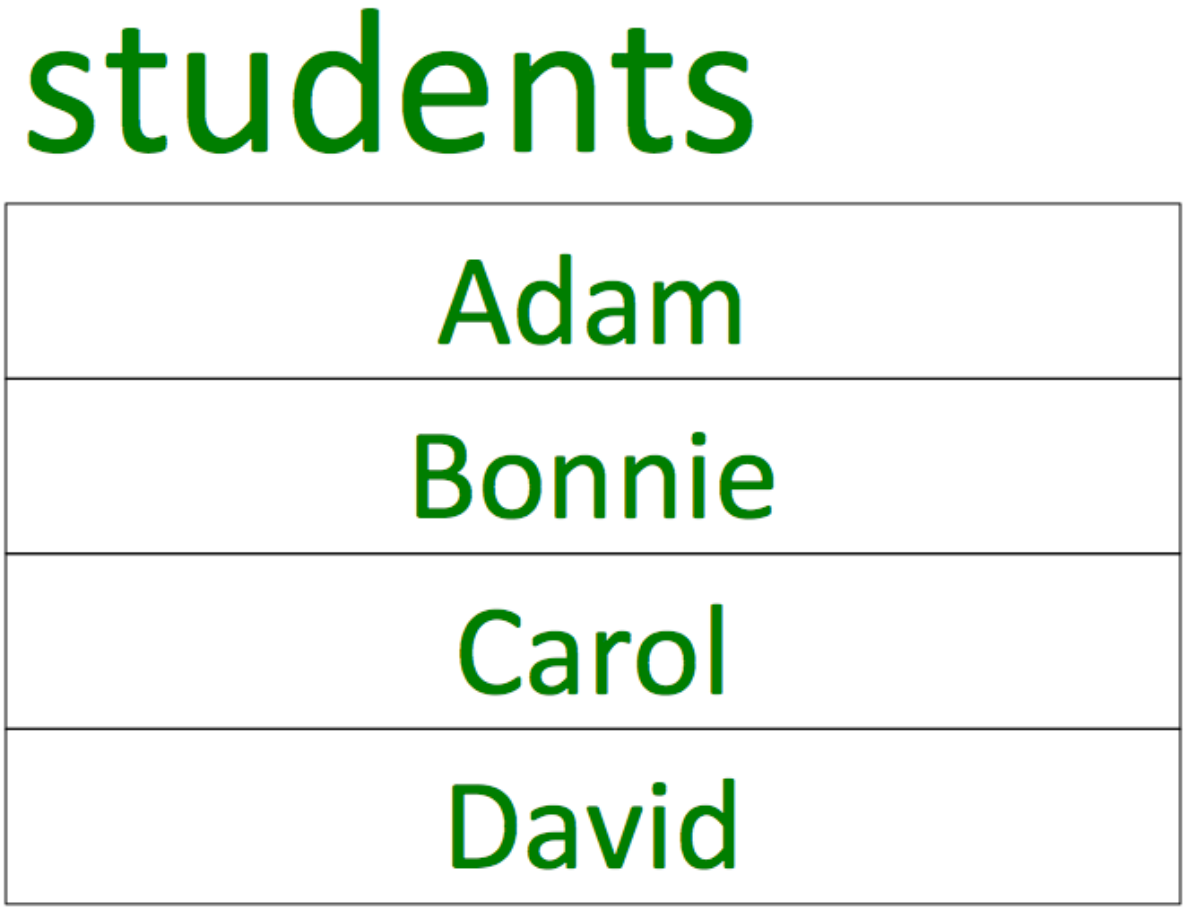
```
for($i=1; $i<=6; $i++;){
```

```
        print '<h' . $i . '>I will not talk in class.</h' . $i .
'>';

}
```

It will print out all 6 heading levels. Give it a try. I started with a different starting point, one instead of zero. My condition had less than or equal to instead of just less than. Of course I was also printing out the value of i each time to create the different HTML elements.

Ok so maybe the for loop examples are not something you would use but they are good examples of the potential uses. In this class we actually don't use the for loop all that much but we use the concept all the time. Let me lead you through another example.

We have talked about variables and we have learned that, we can assign a value to a variable, we can print a variable, we can use the variable in a formula and we can change the value of a variable. There is a variable that can have more than one value, it is called an array. You can think of an array as a table. For example here is a table or an array of my students:



To create this in PHP code we would initialize the variable to be an array with the values in it like this:

```
$students = array('Adam', Bonnie, 'Carol', 'David');
```

The real question is how do we access or print these variables? A temporary way to print your array just to see if it works is to use print_r method made specifically for arrays.

```
print '<p>Students</p><pre>';

print_r ($students);

print '</pre>';
```

This would print it out like this:

Students

```
Array
(
    [0] => Adam
    [1] => Bonnie
    [2] => Carol
    [3] => David
)
```

This is of course not the way to print your array on a web page but it is a handy way so that you can "see" what is in the computers memory. You may be able to guess that we can access each value by its index number. Of course they start with zero since computers like to start with 0. We can print out one value by putting the index number in the square brackets:

```
print '<p>' . $students[0] . '</p>';
```

You could put this into a for loop and print them all out by replacing the 0 with an $i variable

```
for ($i=0;$i<count($students);$i++){

    print '<p>' . $students[$i] . '</p>';

}
```

However since it is so common to use arrays in PHP we have a special loop structure just for them. It is called a foreach loop and this is what we should use for arrays.

The syntax for the foreach loop is:

```
foreach(array-variable -name as one-row-of-the-array){

// Do these instructions

}
```

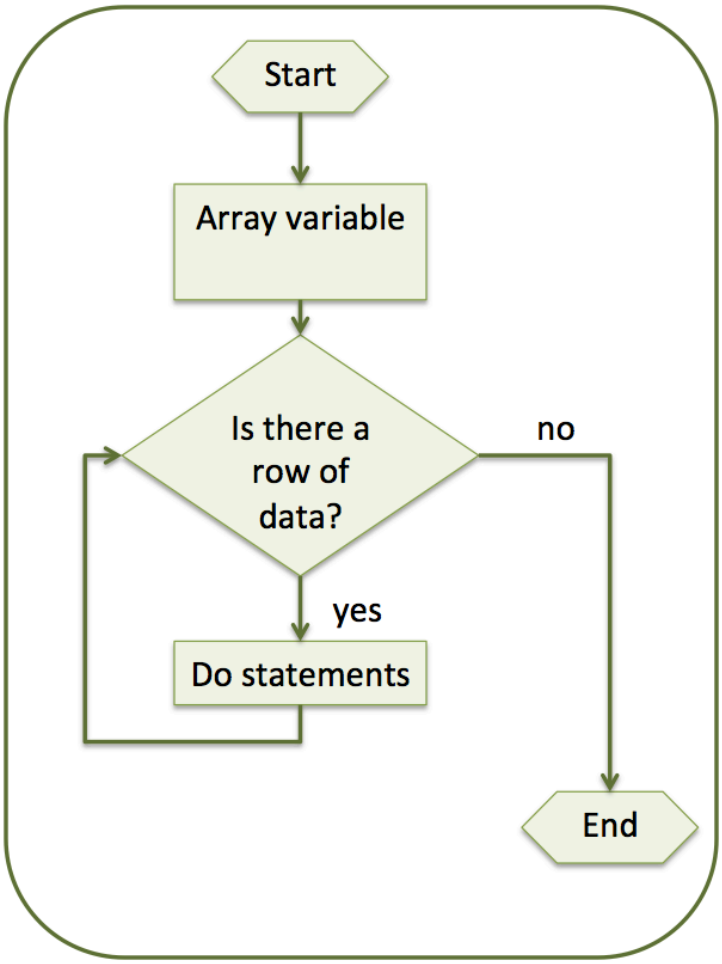
Try to name your arrays in the plural form, ie students, cars, people. Then use the singular name for the as variable, ie student, car, person as it makes for easier understanding of the code.

Let's look at the foreach flowchart to see if you can follow through:

For Each Loop Flow Chart

```
foreach(array as variable){
    // Do these instructions
}
```

```
foreach($students as $student){
    print "<p>Student:" . $student . "</p>";
}
```

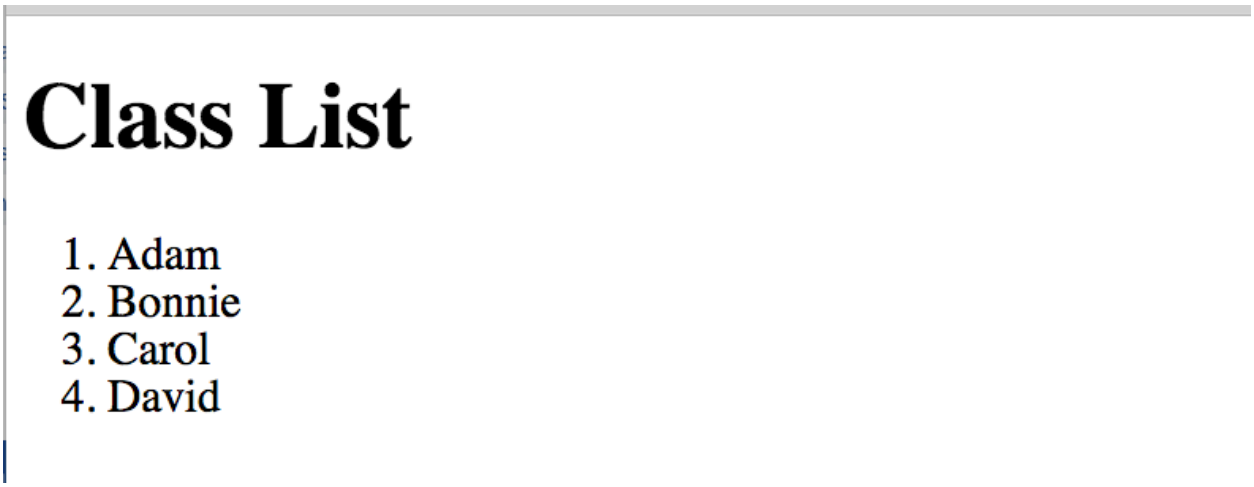


In this case we have an array variable and the question to ask is: Is there a row of data? If yes we do the statements. The increment is automatic as the foreach loop will automatically go to the next record till there are no more. Let's look at some sample code:

```
<?php

$students=array('Adam','Bonnie','Carol', 'David');

print '<h1>Class List</h1>';

print '<ol>';
```

```
foreach ($students as $student){

    print '<li>' . $student . '</li>';

}

print '</ol>';  ?>
```

A couple of things to notice. Since I am printing an ordered list I need to print the open and closing ol elements outside of the loop. The HTML code that will be repeated is the list item (li) inside the loop. See how the variable $student is for the one student at a time. The output from this code looks like this:



With the corresponding HTML source code created looking like this (we did not put in any EOF)

```
<!DOCTYPE html>
<html>
    <head>
        <title>Try Me</title>
        <meta charset="utf-8">
        <meta name="description" content="just a testing file">
    </head>
    <body>
        <h1>Class List</h1><ol><li>Adam</li><li>Bonnie</li><li>Carol</li><li>David</li></ol>    </body>
</html>
```

Clear as mud? I am trying to take quantum leaps with your understanding and I am hoping you are able to follow along with the basics that I am giving you. A one column array is fairly straight forward but generally we always have more than one column. I want to show you a two column array (more than two is really just the same syntax just more of it). To make a two column array you just have each array box (like were Adam was in the box) be an array itself. You can think of an array with more than one column as an array in an array. Here is the code to create a two column array of the freeze over dates for Lake

Champlain. I have tried to help you picture it as an array with many arrays inside it. It may take awhile to make sense.

```
$freezeDates = array(
     array(2000, 'not closed'),
     array(2001, '2001-03-02'),
     array(2002, 'not closed'),
     array(2003, '2003-02-15'),
     array(2004, '2004-01-27'),
     array(2005, '2003-03-08'),
     array(2006, 'not closed'),
     array(2007, '2007-03-02'),
     array(2008, 'not closed'),
     array(2009, 'not closed'),
     array(2010, 'not closed'),
     array(2011, 'not closed'),
     array(2012, 'not closed'),
     array(2013, 'not closed'),
     array(2014, '2014-02-12'),
     array(2015, '2015-02-16')
);
```

Notice how each row (except for the last one) has a comma at the end. As a reminder, to create a one-dimensional array (one column array) it would be:

```
$students = array('Adam', 'Bonnie', 'Carol', 'David');
```

So 'Adam' was one element in the array just like array(2000, 'not closed') is one element in the array. We are just making the one element bigger. It is a little bit more complicated when looping through the multi column array as you need to specify which column to display. Remember to start with zero.

```
foreach ($freezeDates as $freezeDate) {

     print '<p>' . $freezeDate[0] . ' – ' . $freezeDate[1] . '</p>';

}
```

Make sense? Arrays are used heavily in the web. How we populate the array changes as you gain more experience but looping through the array is the same. It will be hard at first but once you wrap your head around it won't be as bad as it sounds the first time. This video may help (especially when you make a mistake)

Code Walk Through:
https://www.youtube.com/watch?v=9WEIGziEylo&list=PLxWIQk1hqg09WQMlrnyjK8sVsrdoqo8lJ&index=14&t=545s

Self Test Questions

1. The PHP command print causes whatever that comes after it to become _____?
2. This will print:
    a. print 1 + 2 * 3;
    b. print 10 / 2 - 3;
    c. print 8 + 20 * 2 - 6;
    d. print (8 + 2) * (6 - 2) + 2;

       For the next few questions use these values:
       $a = 1;
       $b = 2;
       $c = 3;
    e. print $a . $b
    f. print ($a + $b) * $c
    g. print $a + $b * $c
3. T (true) or F (false)
    a. print 'Welcome to the Jungle';
    b. print "<h2>Welcome to the Jungle</h2>";
    c. echo ' Welcome to the Jungle ';
    d. print 'Let's Go Fishing!';
    e. print 'Let\'s Go Fishing!';
    f. echo "Let's Go Fishing!";
4. Write a PHP statement to print your name.
5. Write a PHP statement to print a heading one element with the text Don't Make Me Think
6. Given this array $numbers =array(3,33,37,51,57,21);
   what will this statement print?
   print ' Number: ' . $numbers[3];
7. Create an array for your top three favorite places, print the array.

Challenge Questions

1. **Personal Information**
   Write some php code that stores the following information in respective variables and then prints it on the page.
    - Your name
    - Your address, with city, state and ZIP
    - Your telephone number
    - Your college major

2. **Sales Prediction**
   A company has determined that its annual profit is typically 23 percent of total sales. Write some php code that will display the profit made from an amount that is stored as a variable. Print the profit on screen.

3. **Distance Traveled**
   Assuming there are no accidents or delays, the distance that a car travels down the interstate can be calculated with the following formula: *Distance = Speed X Time*
   A car is traveling at 70 miles per hour. Write some php code that displays the following:
   - The distance the car will travel in 6 hours
   - The distance the car will travel in 10 hours
   - The distance the car will travel in 15 hours

4. **Day of the Week**
   Write some php code that uses an array to print the day of the week based on a number, where 1 = Monday, 2 = Tuesday, 3 = Wednesday, 4 = Thursday, 5 = Friday, 6 = Saturday, 7 = Sunday.

5. **Roman Numerals**
   Write some php code that prints a roman numeral from 1 to 10 depending on the value of another variable. (Hint: store the roman numerals in an array)

Answers to Challenge Questions

1. **Answer:**

```
<?php
$name = “Alec”;
$address = “388 College St, Burlington Vt, 05405”;
$number = “123-456-7890”;
$major = “Computer Science”;

print $name;
print $address;
print $number;
print $major;
?>
```

2. **Answer:**

```
<?php
$totalSales = 100;

print 0.23 * $totalSales;
?>
```

3. **Answer:**

```
<?php
$speed = 70;

print $speed * 6;
print $speed * 10;
print $speed * 15;
?>
```

4. **Answer:**

```
<?php
$array = [Monday,Tuesday,Wednesday,Thursday,Friday,Saturday,Sunday];
$day = 1;

$print $array[$day];
?>
```

5. **Answer:**

```
<?php

$number = 5;
$romanNumerals = [‘I’, ‘II’, ‘III’, ‘IV’, ‘V’, ‘VI’, ‘VII’, ‘VIII’, ‘IX’, ‘X’];

print $romanNumerals[$number - 1];

?>
```