Date: 2021-01-22

* Php info. Function:
  + - phpinfo() are used to see the information about a php version, supporting library etc.
    - Ex: phpinfo();
* Use of echo for a print string and use of html tag:
  + - Ex: echo ‘hello worled..’;
      * echo echo ‘<b>hello</b>’;
* put a php in normal html:
  + - Ex:<?php

$msg = ‘helllo’;

?>

<input type=’text’ valur=”<?php echo $msg; ?>”>

* Comments in php
  + - Single line comment:
      * + //
    - Multiline comment:
      * + /\*….\*/
* Error reporting:
  + - If we don’t want to display any error on browser then we can do it by using a php.ini fire.
    - Set error\_reporting = 0 –means not set any yet…
    - Error Level Constants:
    - ; E\_ALL - All errors and warnings (includes E\_STRICT as of PHP 5.4.0)
    - ; E\_ERROR - fatal run-time errors
    - ; E\_RECOVERABLE\_ERROR - almost fatal run-time errors
    - ; E\_WARNING - run-time warnings (non-fatal errors)
    - ; E\_PARSE - compile-time parse errors
    - ; E\_NOTICE - run-time notices (these are warnings which often result
    - ; from a bug in your code, but it's possible that it was
    - ; intentional (e.g., using an uninitialized variable and
    - ; relying on the fact it is automatically initialized to an
    - ; empty string)
    - ; E\_STRICT - run-time notices, enable to have PHP suggest changes
    - ; to your code which will ensure the best interoperability
    - ; and forward compatibility of your code
    - ; E\_CORE\_ERROR - fatal errors that occur during PHP's initial startup
    - ; E\_CORE\_WARNING - warnings (non-fatal errors) that occur during PHP's
    - ; initial startup
    - ; E\_COMPILE\_ERROR - fatal compile-time errors
    - ; E\_COMPILE\_WARNING - compile-time warnings (non-fatal errors)
    - ; E\_USER\_ERROR - user-generated error message
    - ; E\_USER\_WARNING - user-generated warning message
    - ; E\_USER\_NOTICE - user-generated notice message
    - ; E\_DEPRECATED - warn about code that will not work in future versions
    - ; of PHP
    - ; E\_USER\_DEPRECATED - user-generated deprecation warnings
    - ;
    - ; Common Values:
    - ; E\_ALL (Show all errors, warnings and notices including coding standards.)
    - ; E\_ALL & ~E\_NOTICE (Show all errors, except for notices)
    - ; E\_ALL & ~E\_NOTICE & ~E\_STRICT (Show all errors, except for notices and coding standards warnings.)
    - ; E\_COMPILE\_ERROR|E\_RECOVERABLE\_ERROR|E\_ERROR|E\_CORE\_ERROR (Show only errors)
    - ; Default Value: E\_ALL & ~E\_NOTICE & ~E\_STRICT & ~E\_DEPRECATED
    - ; Development Value: E\_ALL
    - ; Production Value: E\_ALL & ~E\_DEPRECATED & ~E\_STRICT
    - ; http://php.net/error-reporting
    - error\_reporting = E\_ALL (default)
* How to change php.ini file setting from our php filr:
  + - Ini\_set(‘setting\_name’,’value’);
      * Ex: ini\_set(‘error\_reporting’,’0’);
    - For specific setting:
      * Ex: error\_reporting(value);
* Variables in php
  + - There are no need any keyword for declare a variable.
    - But variable name must be start with a $ sign.
    - A variable type is depend on the value we assign.
    - If we assign a string it become a string or if we assign a number it become a number and so on.
* Concatenation in php:
  + - There are a 2 way to combat a string
    - ‘+’ sign and ‘.’ Sign.
* If ….else statement:
  + - Syntax:

If(condition)

{

//code

}

else

{

//code

}

* If …else …if statement:
  + - Syntax:

If(condition)

{

//code

}

else if(condition)

{

//code

}

else

{

//code

}

* Assignment operator:
  + - ‘=’
    - ‘+=’
    - ‘-+’
    - ‘\*=’
    - ‘/=’
    - ‘.=’ – for concatenate a string..
* Comparison operatror:
  + - ‘==’
    - ‘!=’
    - ‘<’
    - “>’
    - ‘<=’
    - ‘>=’
* Arithmetic operator:
  + - +
    - -
    - \*
    - /
    - %
    - ++
    - --
* Logical operator:
  + - &&
    - ||
    - !
* Triple Equals:
  + - ‘===’ not compare a equality but also check the datatype of that values.
    - It is used when we want to check a strictly comparison.
* While loop:
  + - Syntax:

While(condition)

{

//code

}

* Do while loop:
  + - Syntax:

Do

{

//code

}

While(condition);

* + - This loop execute once even the condition is false.
* For loop:
  + - Syntax:

For(initialization ; condition ; increment/decrement)

{

//code

}

* Switch statement:
  + - Syntax:

switch(var\_name)

{

Case 1:

Case2:

Casen:

}