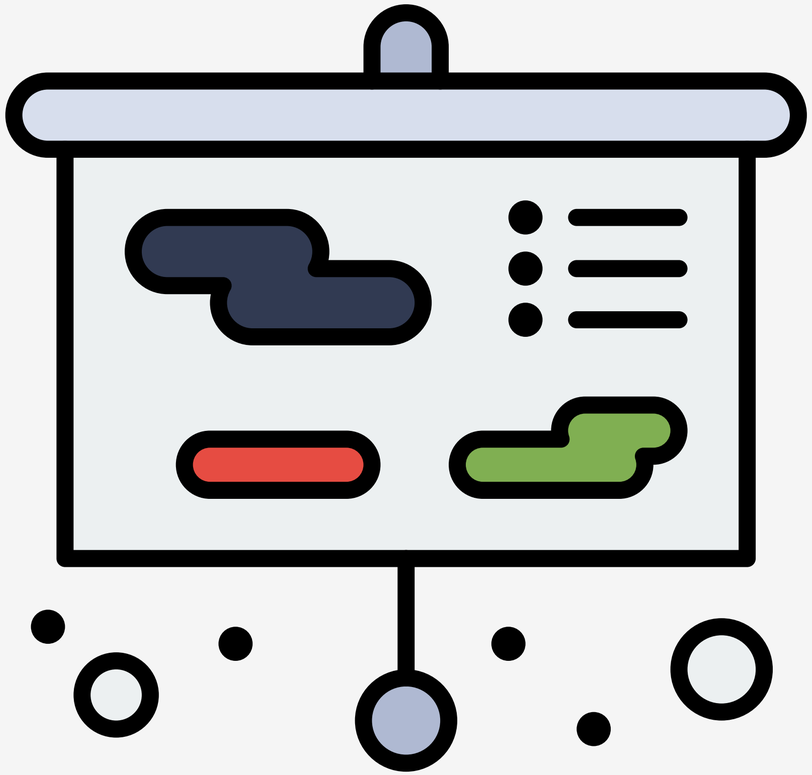
SERVER-SIDE
WEB DEVELOPMENT

Lecture 2

TODAY'S TOPICS



- Conditional Statements
- Loops
- Functions
- Debugging
- **Participation:** Deck of Cards
- **Exercise:** Dominoes

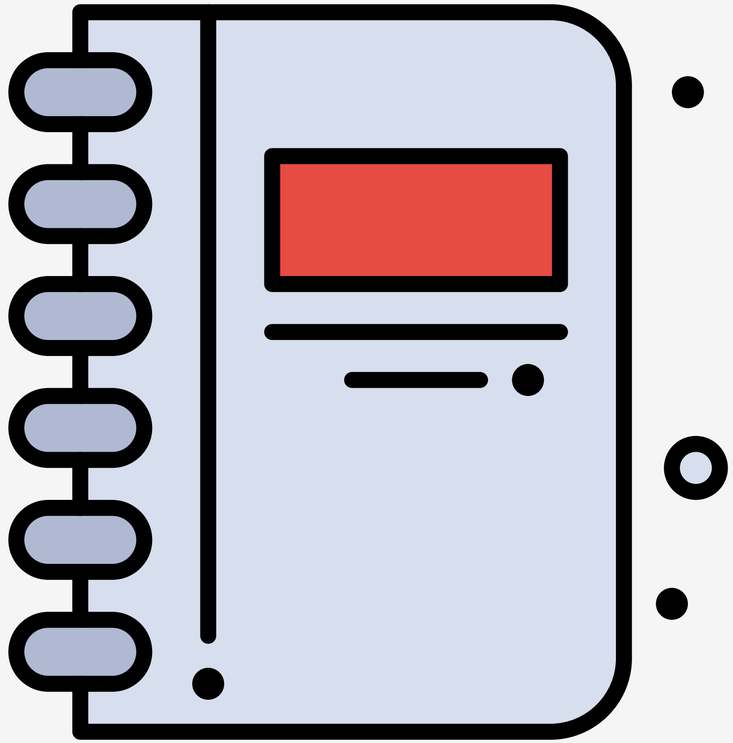
ANNOUNCEMENTS

- LinkedIn Learning
- Hybrid Assignments

QUESTIONS

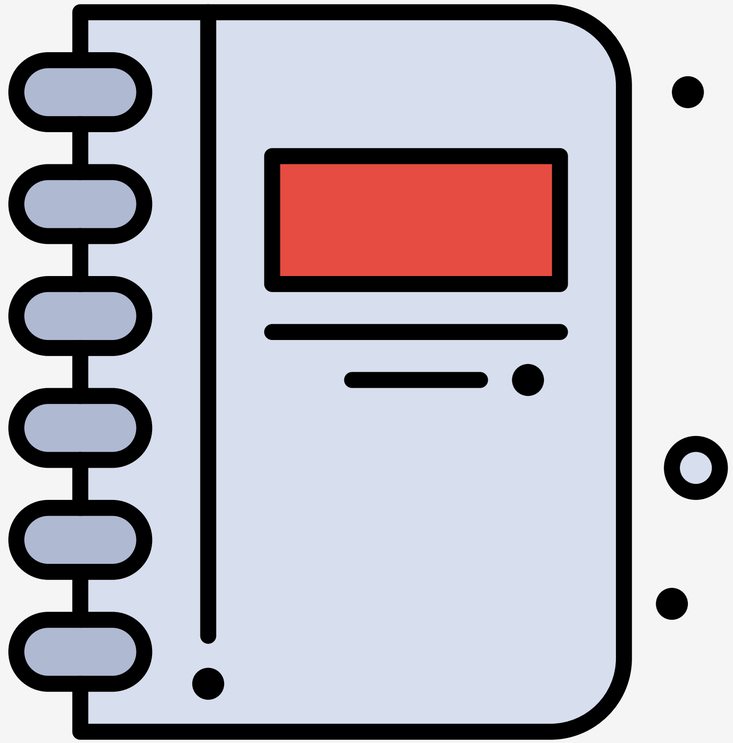
CONDITIONAL STATEMENTS

CONDITIONAL STATEMENTS



- Conditional statements control the flow of a program
- PHP has two conditional statements
  - `if`
  - `switch`

IF...ELSE STATEMENTS



- The `if` statement will execute a block of code, if it's expression is `TRUE`
- The `else` statement can be used to execute an alternate block of code, if the previous if expression is `FALSE`
- The `elseif` statement can be used to check for multiple conditions.
- *NOTE:* PHP also has `else if`, but it will only work with curly braces

IF STATEMENT

```php
<?php
  $a = 3;
  $b = 4;

  // comparing a to b
  if ($a > $b) {
    echo "a is larger than b";
  } elseif ($a < $b) {
    echo "a is smaller than b";
  } else {
    echo "a is equal to b";
  }
```

IF STATEMENT

```php
<?php
  $a = 3;
  $b = 4;

  // comparing a to b
  if ($a > $b) {
    echo "a is larger than b";
  } else if ($a < $b) {
    echo "a is smaller than b";
  } else {
    echo "a is equal to b";
  }
```

IF STATEMENT

```php
<?php
  $a = 3;
  $b = 4;

  // comparing a to b
  if ($a > $b) :
    echo "a is larger than b";
  elseif ($a < $b) :
    echo "a is smaller than b";
  else :
    echo "a is equal to b";
  endif;
```

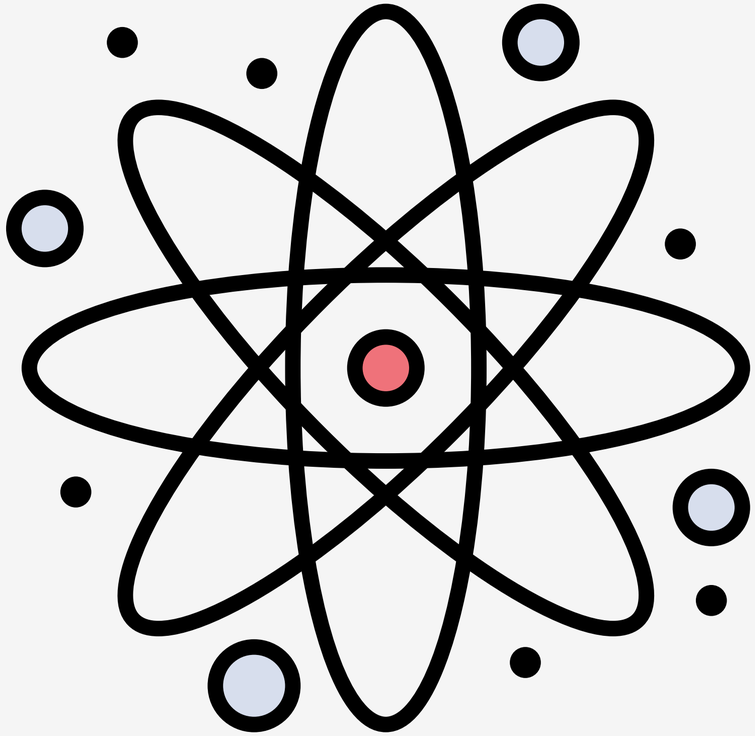
IF STATEMENT

```php
<?php
  $a = 3;
  $b = 4;

  // comparing a to b
  if ($a > $b) :
    echo "a is larger than b";
  else if ($a < $b) :
    echo "a is smaller than b";
  else :
    echo "a is equal to b";
  endif;
```

WILL NOT COMPILE

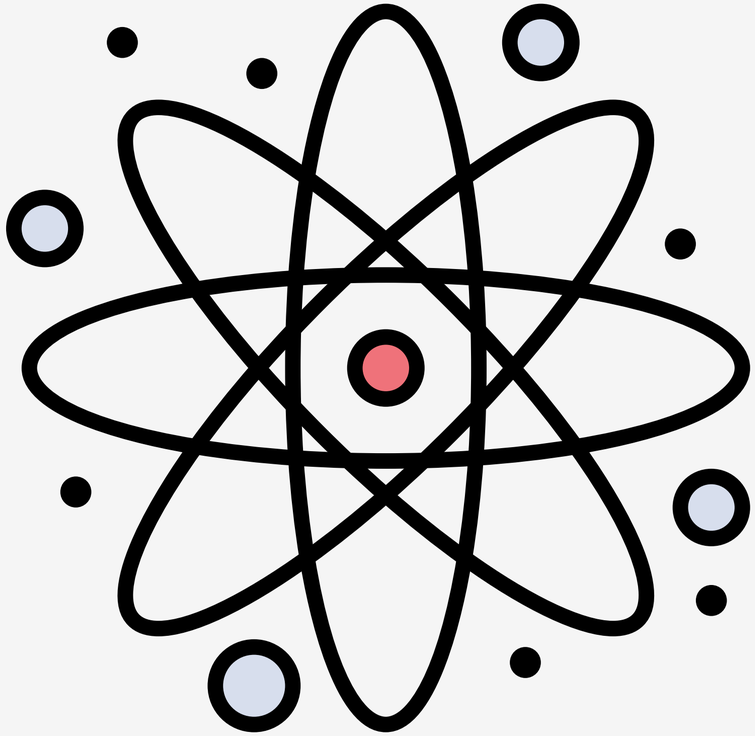
LOOPS

LOOPS



- Loops are used to carry out repetitive tasks or repeat a block of code
- The three most common loops in PHP are:
  - `while`
  - `for`
  - `foreach`

WHILE



- The `while` loop will continue to loop *while* the condition is `TRUE`
- The `while` loop can be used in any situation, but best when the number of iterations is unknowable

WHILE

```php
<?php
  $count = 1;

  // output 1 to 10
  while ($count <= 10) {
    echo "{$count}, ";
    $count++; // increment by 1
  }
```

WHILE

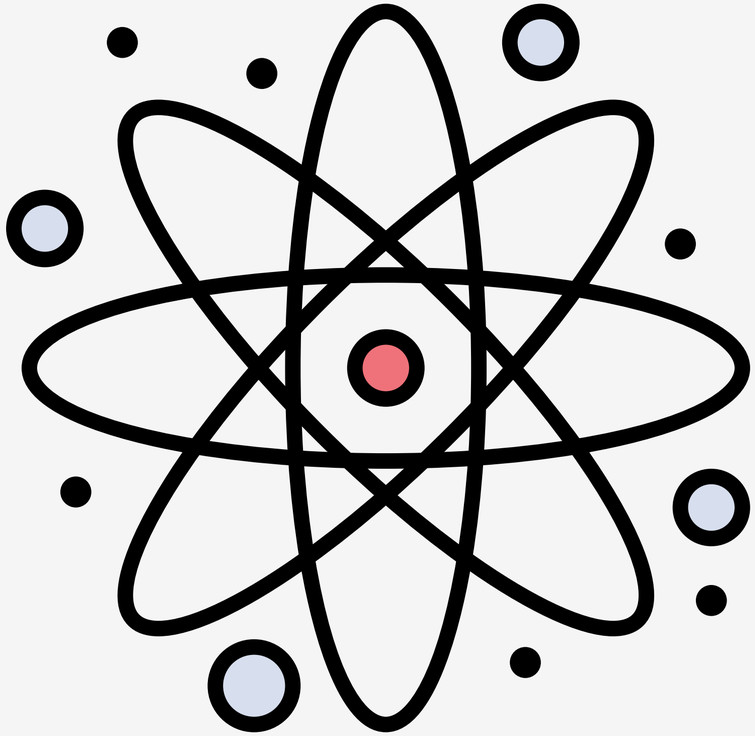
```php
<?php
  $count = 1;

  // output 1 to 10
  while ($count <= 10) {
    echo "{$count}, ";
  }
```

INFINITE LOOP

WHILE

```php
<?php
  $coin = 1;

  // keep looping until $coin === 0
  while ($coin) {
    // randomly choose a number between 0 and 1
    $coin = rand(0, 1);
    echo "{$coin} <br>";
  }
```

FOR



- The `for` loop consists of a three part expression: initialization, condition, and iteration
- The condition is checked before each loop and, if TRUE, will execute the block of code
- The iteration will execute at the end of each loop

FOR

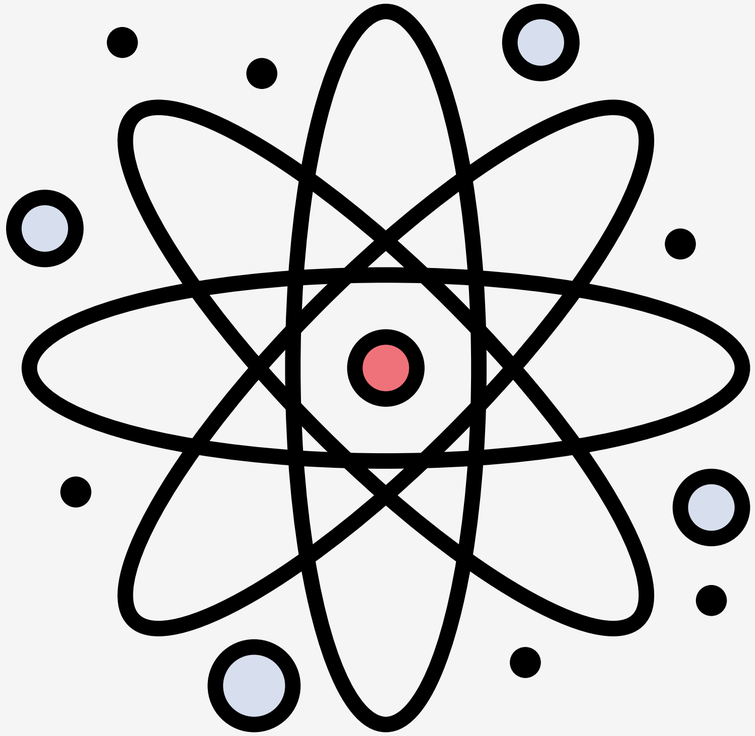
```php
<?php
  // output 1 to 10
  for ($count = 1; $count <= 10; $count++) {
     echo "{$count}, ";
  }
```

FOR

```php
<?php
  $ages = [4, 8, 15, 16, 23, 42];
  $cnt = count($ages);

  for ($i = 0; $i < $cnt; $i++) {
    echo "{$ages[$i]} <br>";
  }
```

INDEXED ARRAYS ONLY

FOREACH



- The `foreach` loop is designed to iterate over arrays
- The `foreach` expression consists of:
  - the array
  - `as` keyword
  - a variable that will hold the items value

FOREACH

```php
<?php
  $ages = [4, 8, 15, 16, 23, 42];

  foreach ($ages as $age) {
    echo "Age: {$age} <br>";
  }
```

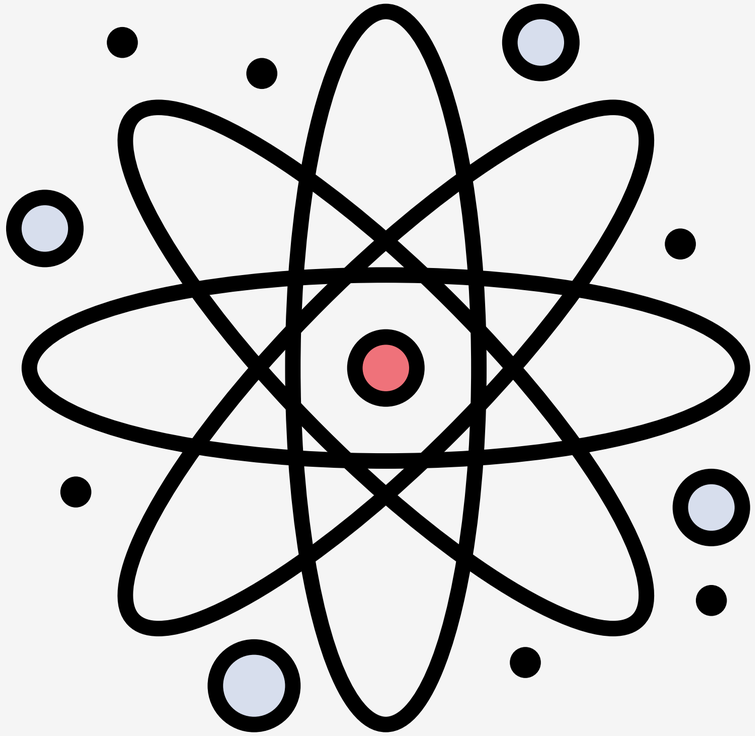
FOREACH

```php
<?php
  $person = [
    "name"   => "Michael",
    "title"  => "Professor",
  ];

  foreach ($person as $key => $value) {
    echo "{$key}: {$value}<br>";
  }
```

DEMO

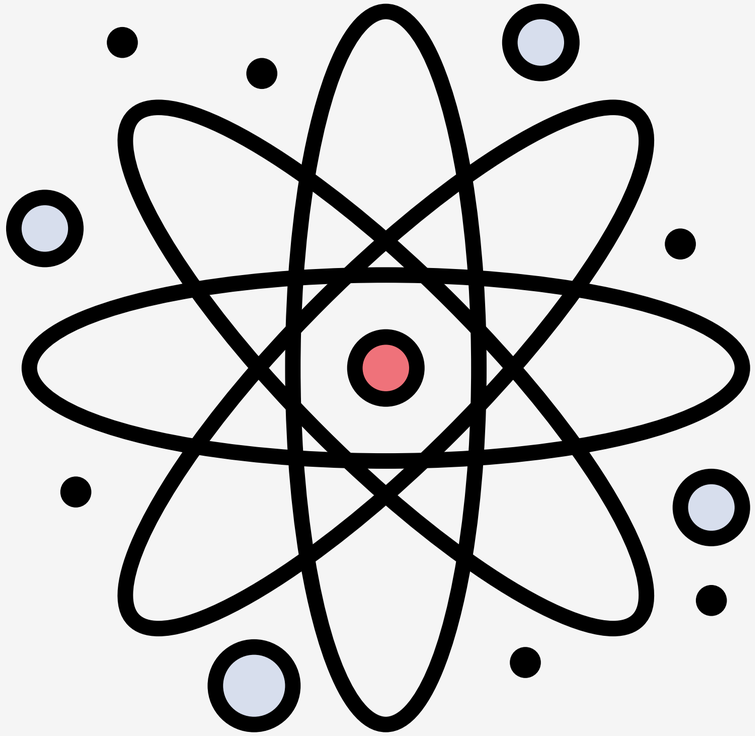
FUNCTIONS

FUNCTIONS



- A function is a block of code that can to be executed at some point in the future
- A function is defined using the following:
  - The `function` keyword
  - The name
  - Set of parentheses `( )`
  - Set of curly braces `{ }`

FUNCTIONS



- A function will not execute until it is called
- To call a function, use the function's name followed by a set of parentheses
- A function can take multiple arguments
- A function can return value with the `return` statement

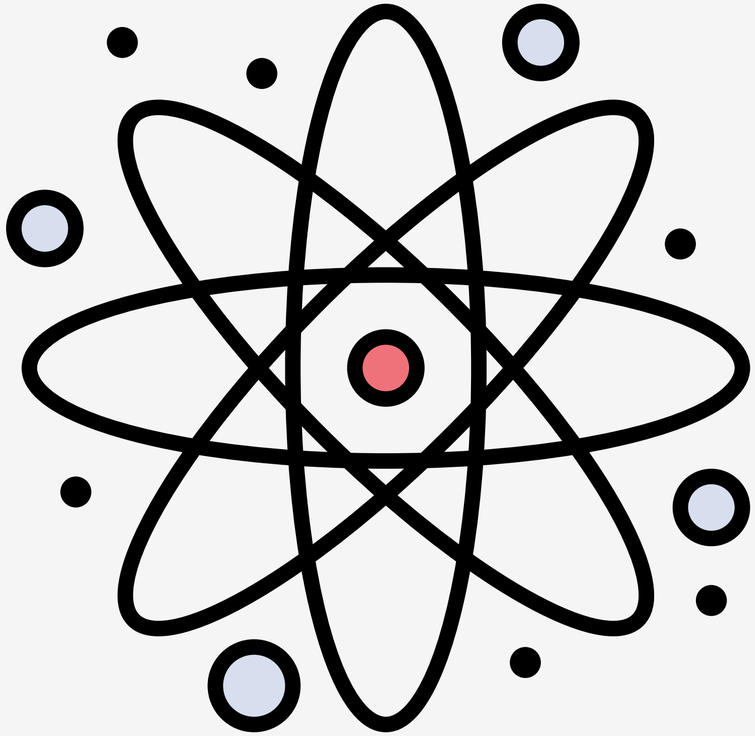
FUNCTION

```php
<?php
  // defining the function
  function say_hello() {
    echo "Hello World!";
  }

  // calling the function
  say_hello(); // Hello World!
```

FUNCTION

```php
<?php
  function add ($a, $b) {
    return $a + $b;
  }

  $result1 = add(3, 4); // 7
  $result2 = add(5, $result1);
  echo $result2; // 12
```

VARIABLE SCOPE



- Two types of scope: global scope and local scope
- Variables defined inside a function are in local scope and only available inside the function
- Variables defined outside of a function are in global scope and are available everywhere *EXCEPT* inside a function
- Use the `global` keyword to use global variables inside a function

FUNCTION

```php
<?php
  $a = 1; // global scope

  function test() {
    return $a; // local scope
  }

  var_dump(test()); // NULL
```

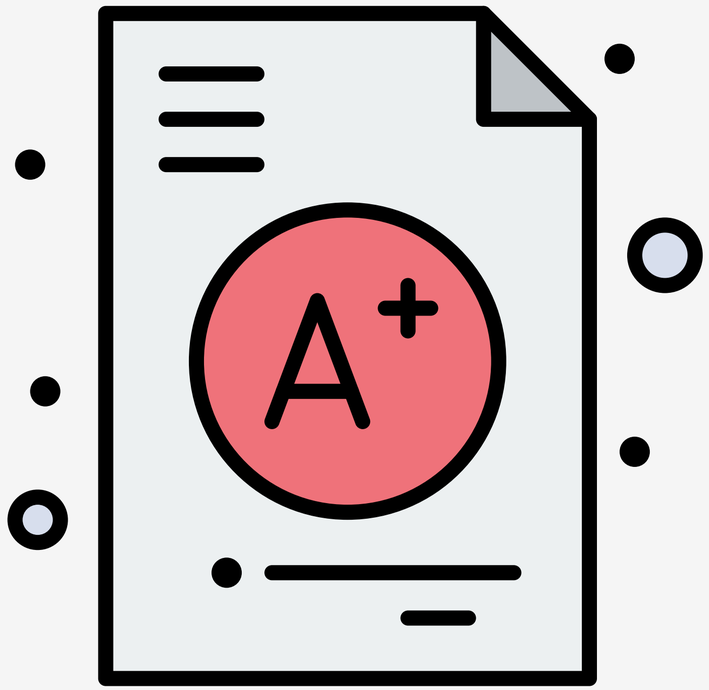
FUNCTION

```php
<?php
  $a = 1; // global scope

  function test() {
    global $a; // global scope

    return $a; // global scope
  }

  var_dump(test()); // int(1)
```

DEMO

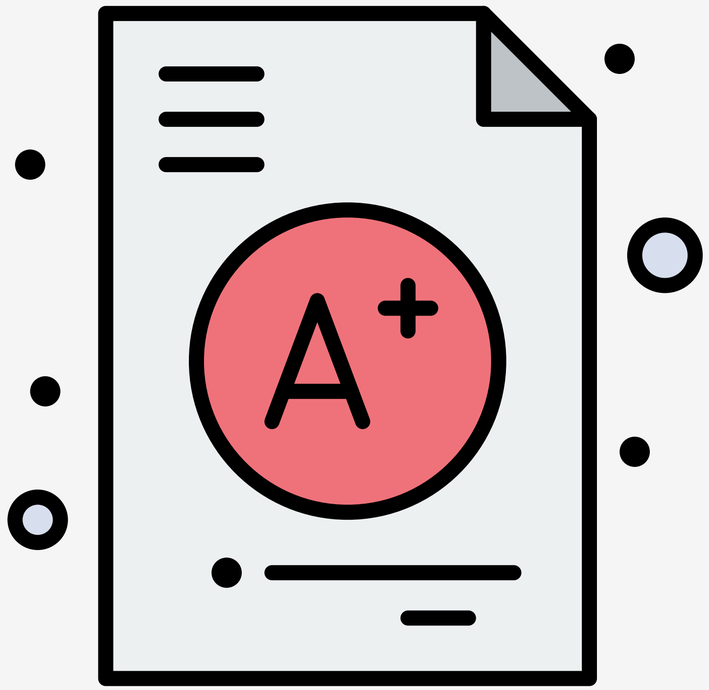
DEBUGGING

COMMON PROBLEMS



- Server Not Running
- PHP Not Working
- Syntax Error
  - Typo
  - Missing Semicolon
  - Missing bracket or quote
  - Misspelled variable name

ERROR TYPES



- Fatal Errors
- Parse Errors
- Warnings & Notices
- Deprecated

FATAL ERROR

```php
<?php
  // undefined function
  $array = [1, 2, 3, 4];
  var_dum($array);
```

PARSE ERROR

```php
<?php
  // missing semicolon
  $array = [1, 2, 3, 4]
  var_dump($array);
```

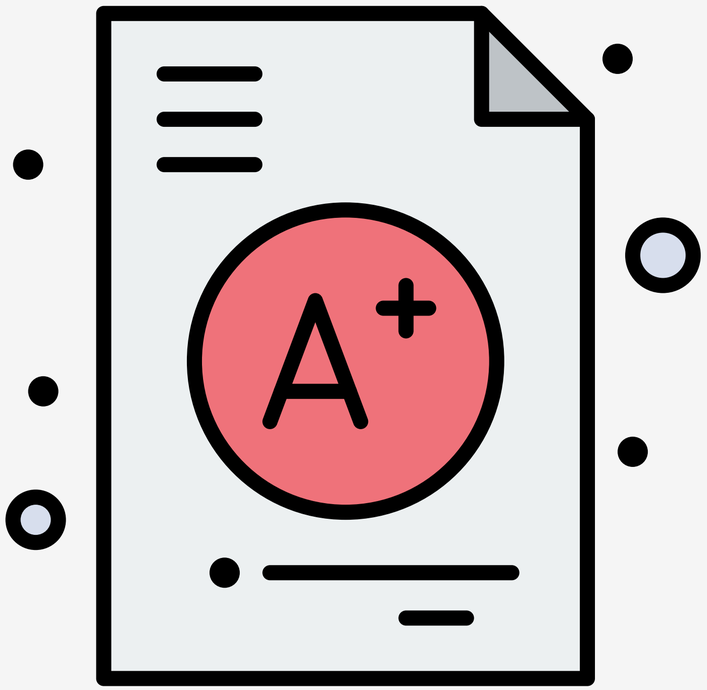
NOTICE

```php
<?php
  // array to string
  $array = [1, 2, 3, 4];
  echo $array;
```

DEPRECATED

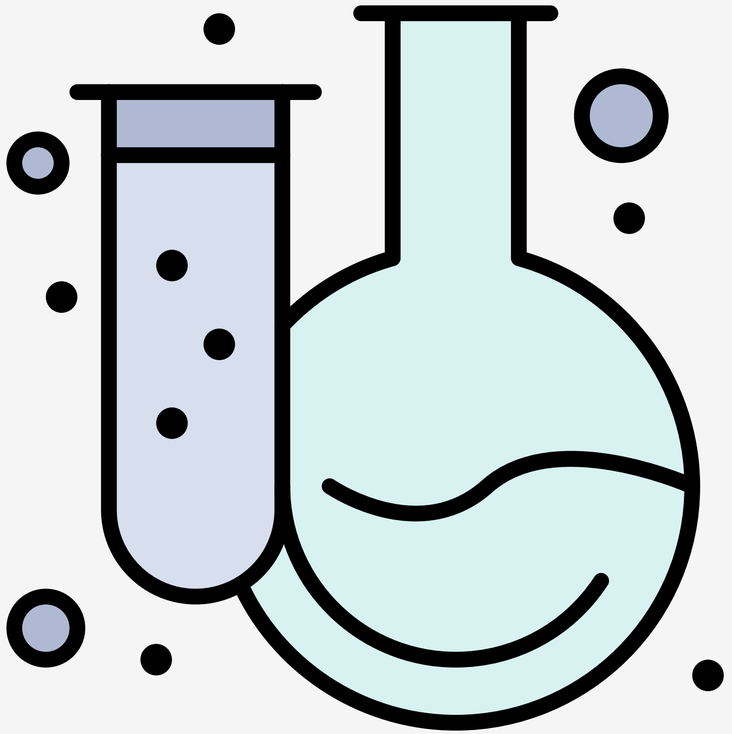
```php
<?php
  // deprecated function
  $array = [1, 2, 3, 4];
  $item = each($array);
```

TROUBLESHOOTING



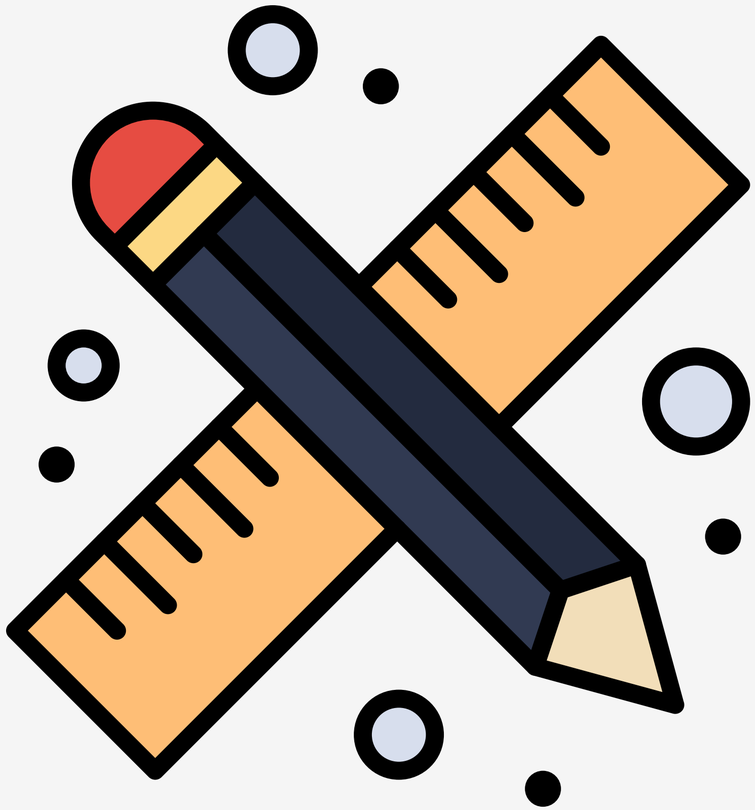
- `echo`
- `print_r()`
- `var_dump()`
- `gettype()`
- `get_defined_vars()`

DECK OF CARDS



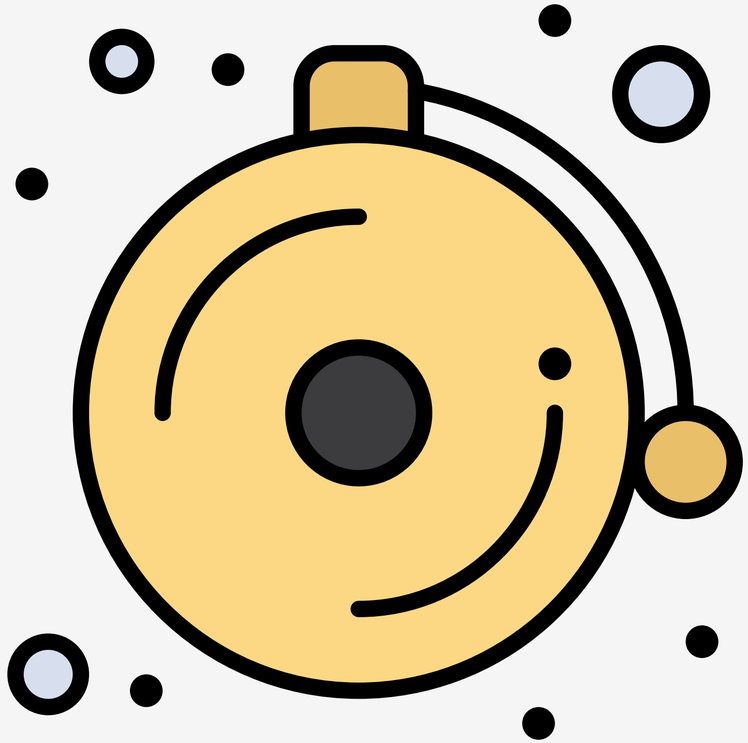
- *GITHUB CLASSROOM ASSIGNMENT*
- Use arrays and loops to generate and display all 52 cards of a deck
- Use the `rand()` function to generate 52 random cards
- Provide comments and pseudo code
- Submit your repository URL
- *DUE:* **Thu. Oct. 1 @ 11:59 PM**

DOMINOES



- *GITHUB CLASSROOM ASSIGNMENT*
- Use loops and an array to generate 100 random dominoes
- Use the `rand()` function to randomize the dominoes
- Use the CSS classes provided
- Submit your repository URL
- *DUE:* **Thu. Oct. 1 @ 11:59 PM**

NEXT TIME...



- Requests
- Includes
- **Participation:** aMazing Adventure
- **Exercise:** Functional Fishing