01 - Tags

Lists

Types of lists

1. Definition lists
2. Ordered lists
3. Unordered lists
4. Nested lists

List tags

- `<dl>` - defines a description/definition list
- `<dt>` - defines a term in a description
- `<dd>` - describes the term in a description data
- `<ul>` - defines an unordered list
- `<ol>` - defines an ordered list
- `<li>` - defines a list item

Definition list

```
<html>
    <body>
        <dl>
            <dt>HTML</dt>
                <dd>Hyper Text Markup Language</dd>
            <dt>CSS</dt>
                <dd>Cascade Style Sheet</dd>
        </dl>
    </body>
</html>
```

HTML
    Hyper Text Markup Language
CSS
    Cascade Style Sheet

Ordered list

```
<html>
    <body>
        <h4>Numbered list:</h4>
        <ol>
            <li>Apples</li>
             <li>Bananas</li>
             <li>Oranges</li>
        </ol>
        <h4>Letters list:</h4>
        <ol type="A">
            <li>Lemons</li>
```

```
            <li>Oranges</li>
        </ol>
    </body>
</html>
```

Numbered list:

1. Apples
2. Bananas
3. Oranges

Letters list:

1. Lemons
2. Oranges

Type	Description
type="1"	The list items will be numbered with numbers (default)
type="A"	The list items will be numbered with uppercase letters
type="a"	The list items will be numbered with lowercase letters
type="I"	The list items will be numbered with uppercase roman numbers
type="i"	The list items will be numbered with lowercase roman numbers

> type attribute works with or without quotes

Unordered list

```
<html>
   <body>
      <h4>Unorderd lists:</h4>
      <ul>
         <li>Apples</li>
         <li>Bananas</li>
         <li>Lemons</li>
         <li>Oranges</li>
      </ul>
      <h4>Disk list:</h4>
      <ul type="circle">
         <li>Apples</li>
         <li>Bananas</li>
         <li>Lemons</li>
         <li>Oranges</li>
      </ul>
   </body>
</html>
```

Unorderd lists:

- Apples
- Bananas
- Lemons
- Oranges

Disk list:

- Apples
- Bananas
- Lemons
- Oranges

Value	Description
disc	Sets the list item marker to a bullet (default)
circle	Sets the list item marker to a circle
square	Sets the list item marker to a square
none	The list items will not be marked

Nested lists

```
<html>
   <body>
      <h4>A nested List:</h4>
      <ul>
          <li>Coffee</li>
          <li>
             Tea
             <ul>
                <li>Black tea</li>
                <li>Green tea</li>
                <ol>
                   <li>aaaaa</li>
                   <li>bbbbb</li>
                </ol>
             </ul>
          </li>
          <li>
             Ice Cream
             <ol type="i">
                <li>Vanila</li>
                <li>Chocolate</li>
                <li>Strowberry</li>
             </ol>
          </li>
      </ul>
    </body>
</html>
```

A nested List:

- Coffee
- Tea
  - Black tea
  - Green tea
  1. aaaaa
  2. bbbbb
- Ice Cream
  1. Vanila
  2. Chocolate
  3. Strowberry

--- # Tables

Table tags

`<table>` - defines a table
`<th>` - defines a header cell in the table
`<tr>` - defines a row in a table
`<td>` - defines a cell in a table
`<caption>` - defines a table caption

Table attributes

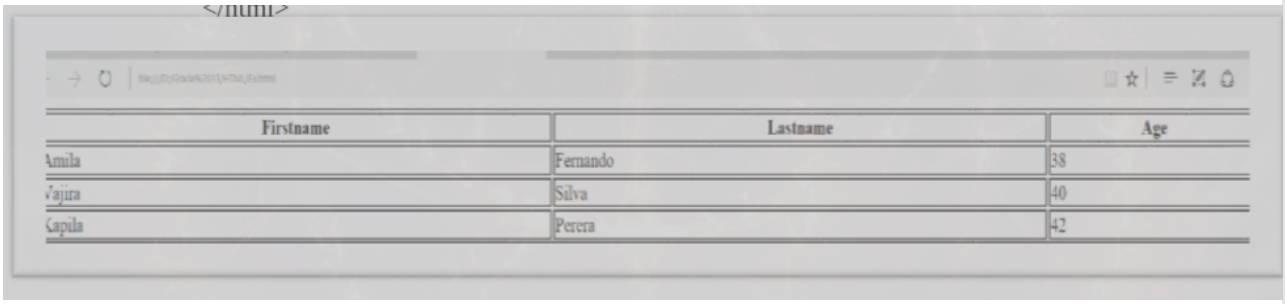
`colspan` - makes a cell span many columns
`rowspan` - makes a cell span many rows
`border` - specifies the size of the table border
`style` - use CSS to style the table

```
<html>
   <body>
      <table style="width:100%" border="1">
         <caption>Personal Information</caption>
         <tr>
            <th>Firstname</th>
            <th>Lastname</th>
            <th>Age</th>
         </tr>
         <tr>
            <td>Amila</td>
            <td>Fernando</td>
            <td>38</td>
         </tr>
         <tr>
            <td>Vajira</td>
            <td>Silva</td>
            <td>40</td>
         </tr>
         <tr>
            <td>Kapila</td>
            <td>Perera</td>
            <td>42</td>
         </tr>
      </table>
```

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

Personal Information

Firstname	Lastname	Age
Amila	Fernando	38
Vajira	Silva	40
Kapila	Perera	42



> Note that the caption for the table should be under the `table` tag definition

```
<table style="width:60%" border="1">
    <caption>Personal Information</caption>
```

Embed tags

```
<embed type="image/jpg" src="pic_trulli.jpg" width="300" height="200">
<embed type="video/webm" src="video.mp4" width="400" height="300">
<embed type="text/html" src="snippet.html" width="500" height="200">
```

Input tags

Attributes,

- `size` - the number of characters that will be displayed (width size)
- `maxlength` - the number of characters that is allowed in the input box

Different values that `type` accepts,

- `<input type="button">`
- `<input type="checkbox">`
- `<input type="color">`
- `<input type="date">`
- `<input type="datetime-local">`
- `<input type="email">`
- `<input type="file">`
- `<input type="hidden">`
- `<input type="image">`
- `<input type="month">`
- `<input type="number">`
- `<input type="password">`

- `<input type="radio">`
- `<input type="range">`
- `<input type="reset">`
- `<input type="search">`
- `<input type="submit">` (for a form)
- `<input type="tel">`
- `<input type="text">` (default value)
- `<input type="time">`
- `<input type="url">`
- `<input type="week">`

Radio buttons and Check boxes

Radio button	Check boxes
When a user selects one radio button, it automatically deselects any other radio buttons in the same group. This ensures that the user can choose only one option from the set.	Users can select multiple checkboxes simultaneously, and each checkbox operates independently of others.

```
<input type="radio" name="gender" value="male" checked> Male
<input type="radio" name="gender" value="female"> Female
```

<input type="radio" name="gender" value="male" checked> Male
<input type="radio" name="gender" value="female"> Female

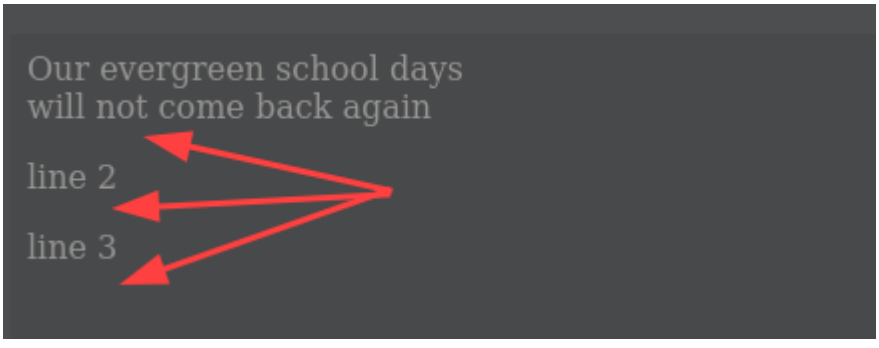
```
<input type="checkbox" name="fruit" value="apple" checked> Apple
<input type="checkbox" name="fruit" value="banana"> Banana
```

<input type="checkbox" name="fruit" value="apple" checked> Apple
<input type="checkbox" name="fruit" value="banana"> Banana

Paragraph tag

Note that **after** each and every paragraph tag, a blank line is being inserted at the end.

```
<p>Our evergreen school days<br/> will not come back again</p>
<p>line 2</p>
<p> line  3</p>
```



Button tag

Used to create a button. (for forms)

```
<button type="submit">Submit</button>
```

Submit

iframe tag

`<iframe>` tag specifies an inline frame.

```
<iframe src="demo_iframe.htm" style="height:200px;width:300px;" title="Iframe Examp
```

select tag

This tag is used to create a drop down menu

```
<label for="cars">Choose a car:</label>

<select name="cars" id="cars">
  <option value="volvo">Volvo</option>
  <option value="saab" selected>Saab</option>
  <option value="mercedes">Mercedes</option>
  <option value="audi">Audi</option>
</select>
```

Choose a car:

Saab

section tag

This is used to add a hyperlink to a specific section of your code.

```
<section id="target-section">
    <h2>Target Section</h2>
</section>
<a href="#target-section">Go to Target Section</a>
```

font tag

- To change the color of the letters

```
<font color="red">font Color is changed to red</font>
```

font Color is changed to red

- To change the size of the letters

```
<p><font size="1" > Size of the font is changed to 1 </font></p>
<p><font size="2" > Size of the font is changed to 2 </font></p>
```

Size of the font is changed to 1

Size of the font is changed to 2

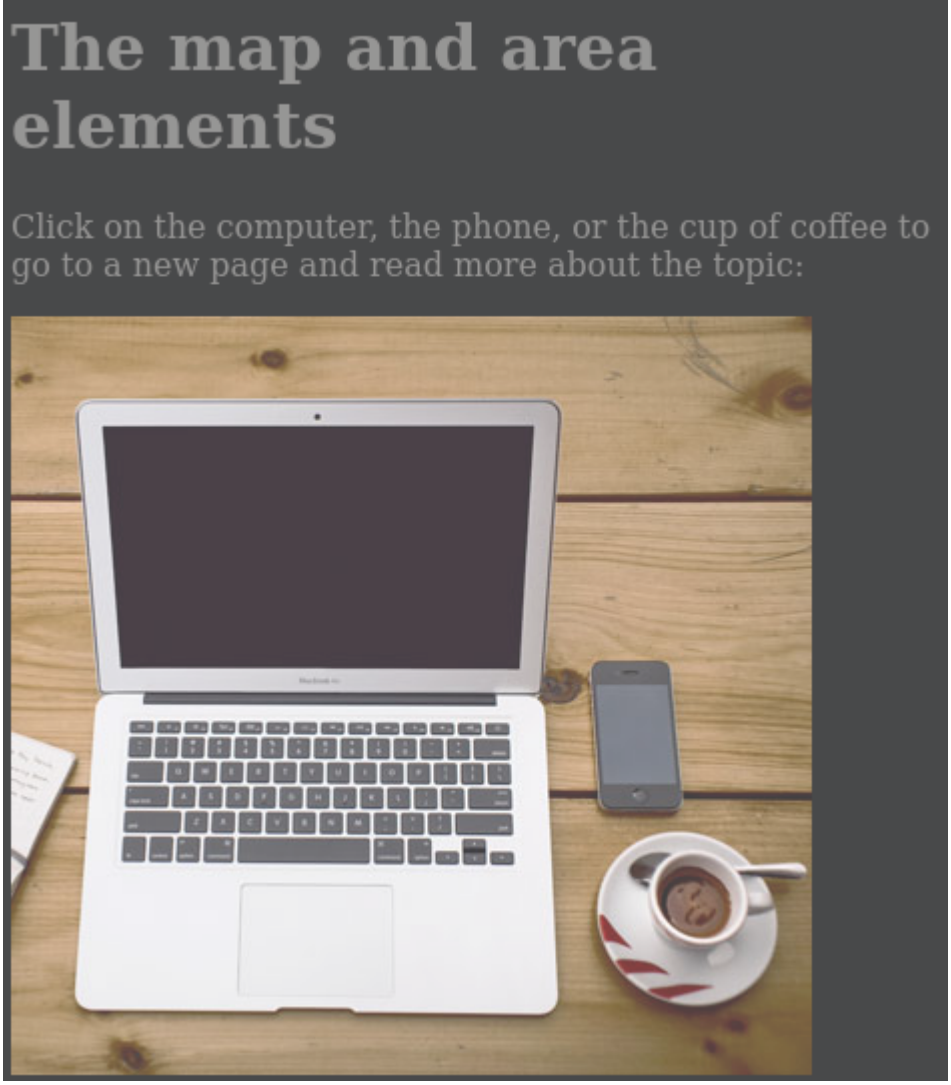
- To change font type ```html

font type is changed to "verdana"

font type is changed to "Algerian"

```

font type is changed to "verdana"

font type is changed to "Algerian"

# Map tag

This creates an image with a clickable link.

```
<img src="workplace.jpg" alt="Workplace" usemap="#workmap" width="400" height="379"

<map name="workmap">
  <area shape="rect" coords="34,44,270,350" alt="Computer" href="computer.htm">
  <area shape="rect" coords="290,172,333,250" alt="Phone" href="phone.htm">
  <area shape="circle" coords="337,300,44" alt="Cup of coffee" href="coffee.htm">
</map>
```
```



We create a map with a name and then we use the `img` tag with the attribute `usemap` to specify the map

When we click on a certain area of the image, we go to different pages. Those co-ordinates are specified in the `map`

Marquee tag

- marquee with different widths

```
<marquee width = "50%"> 50% width of page</marquee>
```

CSS

- CSS describes how HTML elements are to be displayed on screen, paper, or in other media
- For easy maintenance and update web pages, style sheets guarantees the consistency throughout the website, can restyle the HTML without modifying it, a single document can be presented in multiple styles.

Selectors

```
CSS selectors are used to "find" (or select) HTML elements based on their element name, id, class, attribute, and more
```

There are a couple of different selectors.

1. element selector - here we find the element using the element name i.e `div`, `p`

```
p { text-align: center; color: red; }

 <p>Me too!</p>
```

2. id selector - here we find the element by giving it a `id` attribute

```
#para1 { text-align: center; color: red; }

<p id="para1">Hello World!</p>
```

3. class selector - here we find the element by the class its in

```
 .center { text-align: center; color: red; }

<h1 class="center">Red and center-aligned heading</h1>
<p class="center">Red and center-aligned paragraph.</p>
```

Here, this style will be applied to both `h1` and `p` tags as the class is set as `center`

If we want to be specific of what element we want to center using the class, we can use it like this

```
p.center { text-align: center; color: red; }

<h1 class="center">This is not affected</h1>
<p class="center">Red and center-aligned paragraph.</p>
```

In this case, the `h1` won't be affected as in the CSS we are specifying it to **only affect `p` elements in the `center` class**.

4. group selectors - here we can specify multiple tags to have the same style

```
 h1, h2, p { text-align: center; color: red; }

<h1>Hello World!</h1>
<h2>Smaller heading!</h2>
<p>This is a paragraph.</p>
```

In here, we can specify all the tags that have the same style once. So here all `h1`, `h2` and `p` tags will be affected.

> When it comes to the order or interpreting them, **whatever is defined first will be applied to elements.**

```
<html>
<head>
 <style> .hehe { color: red; }; p { color: green; } </style>

</head>
<body>
 <p class="hehe"> hello </p>
</body>
</html>
```

hello

Here since the `hehe`, id selector is defined first for color red, that will be applied regardless of having a p element selector after

Inserting CSS

There are three ways to insert CSS

- External style sheet
- Internal style sheet
- Inline style

External style sheet

With external style sheets, we can change the look of the entire web page. We use the `link` tags to include the style sheet. **This should be included under the `head` tags**

```
<head>
    <link rel="stylesheet" type="text/css" href="mystyle.css">
</head>
```

Advantages of external styling

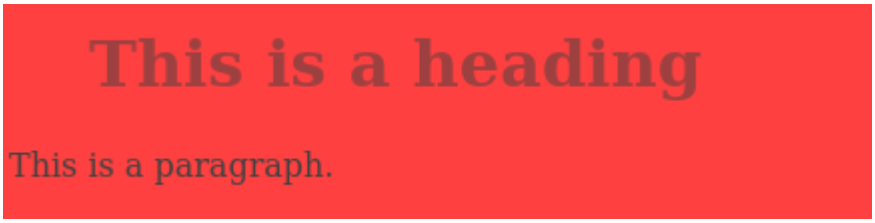
- Less code lines to manage (modification in one place)
- Same style can be applied to multiple web pages
- Reduce code complexity

Internal style sheet

An internal style sheet may be used if one single page has a unique style. These should be specified inside `style` tags. **These should also be included under the `head` tags**

```
<head>
    <style>
        body { background-color: red; }
```

```
        h1 { color: maroon; margin-left: 40px; }
    </style>
</head>


<body>
    <h1>This is a heading</h1>
    <p>This is a paragraph.</p>
</body>
```



Here the style will be applied to only the `h1` tag, not the `p`

Inline style

This is used to specify style to a single tag in the body. **This is included under the `style` attribute of elements**

```
<body>
    <h1 style="color:blue;margin-left:30px;">This is a heading</h1>
    <p>This is a paragraph.</p>
</body>
```

Here the style will be applied to the `h1` tag

Cascading Order

If we were to use all these inserting types in a single document, there is a specific order that the styles will be shown. From the highest priority to lowest, it looks like this.

1. Inline style (inside an HTML element)
2. External and internal style sheets (in the head section)
3. Browser default

I.e say you have `h1` colour set to `red` in a external style sheet for the entire web page. But in one `h1` tag, its color to `blue` is changed by inline style. That means the inline style will take priority and change it to `blue`

```
<h1> this will be red </h1>
<h1 style="color:blue"> this will be blue </h1>
```

Misc

- Embed an image in the background

```
body {
    background-image: url("school.png");
}
```

- Color representation

```
<h1 style="background-color:rgb(255, 99, 71);">RGB</ h1>
<h1 style="background-color:#ff6347;">Hexa decimal</h1>
```

PHP

Integers

The PHP *var_dump()* function returns the data type and value

```
<?php

    $a = 1234; // decimal number
    var_dump ($a);
    echo "<br>";

    $b = -123; // a negative number
    var_dump ($b);
    echo "<br>";

    $c = 0123; // octal number (equivalent to 83 decimal)
    var_dump ($c);
    echo "<br>";

    $d= 0x1A; // hexadecimal number (equivalent to 26 decimal)
    var_dump ($d);
    echo "<br>";

    $e = 0b11111111; // binary number (equivalent to 255 decimal)
    var_dump ($e);
    echo "<br>";

?>
```

Output

```
int(1234)
int(-123)
int(83)
int(26)
int(255)
```

```php
<?php

        $x = 10.365;
        var_dump($x);
        echo "<br>";
```

Output

```
float(10.365)
float(10200)
```

```php
        $y = 10.2e3;
        var_dump($y);

?>
```

Arrays

a corresponding value.
The PHP *print_r()* function prints human-readable information about a variable

Example :

```php
<?php
    $cars = array("Volvo","BMW","Toyota");
    print_r($cars);
    echo "<br>";

        $color_codes = array (
                "Red" => "#ff0000",
                "Green" => "#00ff00",
                "Blue" => "#0000ff" );
        print_r($color_codes);
?>
```

Output

```
Array ( [0] => Volvo [1] => BMW [2] => Toyota )
Array ( [Red] => #ff0000 [Green] => #00ff00 [Blue] => #0000ff )
```

Manipulating PHP Strings

- Get items from an array

Example 01:

```php
<?php
$foods = array("Bread", "Chocolate", "Ice-Cream");

/*Display array elements by using indexes*/
echo "I like " . $foods [0] . ", " . $foods [1] . " and " . $foods [2] . ".";
?>
```

Output
I like Bread, Chocolate and Ice-Cream.

- Get the length of an array

- Get The Length of an Array - The count() Function

The count() function is used to return the length (the number of elements) of an array:

Example:

```php
<?php
$foods = array("Bread", "Chocolate", "Ice-Cream");
echo count($foods);
?>
```

Output
3

- Loop through an indexed array

Example:

```php
<?php
$foods = array("Bread", "Chocolate", "Ice-Cream");
$arrlength = count($foods);

for($x = 0; $x < $arrlength; $x++) {
  echo $foods[$x]. "<br>";
}
?>
```

Output
Bread
Chocolate
Ice-Cream

Assertive arrays

```php
<?php
$device = array("input"=>"Mouse", "output"=>"Monitor", "storage"=>"CD");
echo $device['input'] . " is a input device.";
?>
```

Output
Mouse is a input device.

o

- Loop through assertive arrays

To loop through and print all the values of an associative array, you could use a foreach loop, like this:

Example:

```
<?php
$device = array("input"=>"Mouse", "output"=>"Monitor", "storage"=>"CD");

foreach($device as $x => $x_value) {
   echo "Key=" . $x . ", Value=" . $x_value;
   echo "<br>";
}
?>
```

Output
Key=input,
Value=Mouse
Key=output,
Value=Monitor

String manipulation

- Character count

```
<?php
        $my_str = 'Welcome to PHP';
        echo strlen($my_str);
?>
```

Output
14

- Word count

- **Counting Number of Words in a String**

  The *str_word_count()* function counts the number of words in a string.

  Example:

```
<?php
        echo str_word_count("Hello world!");
?>
```

Output
2

- String reverse

- **Reversing a String**

  The *strrev()* function reverses a string.

  Example:

```
<?php
        echo strrev("Hello world!");
?>
```

Output
!dlrow olleH

- String replace

```php
<?php
    $my_str = "Hello Students!";
    echo str_replace("Students", "Sri Lanka", $my_str);
?>
```

Output

Hello Sri Lanka!

Constants

define(*name, value, case-insensitive*)

Parameters:

- name: Specifies the name of the constant
- value: Specifies the value of the constant
- case-insensitive: Specifies whether the constant name should be case-insensitive. Default is false

- Case sensitive constant

Example 01 : Creates a constant with a **case-sensitive** name:

```php
<?php
    define("GREETING", "Welcome to PHP!");
    echo GREETING;
?>
```

Output

Welcome to PHP!

- Case insensitive constant

Example 02 : Creates a constant with a **case-insensitive** name:

```php
<?php
    define("GREETING", "Welcome to PHP!", true);
    echo greeting;
```

Output

Welcome to PHP!

Operators

- Arithmetic

Operator	*Name*	*Example*	*Description*	*Result*
+	Addition	5+6	Sum of 5 and 6	11
-	Subtraction	8-6	Difference of 8 and 6	2
*	Multiplication	5*2	Product of 5 and 2	10
/	Division	10/4	Quotient of 10 and 4	2.5
%	Modulus	15%4	Remainder of 15 divided by 4	3
**	Exponentiation	2**5	Result of raising 2 to the 5'th power	32

- `Assignment`

Operator	*Name*	*Example*	*Same as ...*
=	Assign	$x = $y	$x = $y
+=	Add and assign	$x += $y	$x = $x + $y
-=	Subtract and assign	$x -= $y	$x = $x - $y
*=	Multiply and assign	$x *= $y	$x = $x * $y
/=	Divide and assign	$x /= $y	$x = $x / $y
%=	Divide and assign modulus	$x %= $y	$x = $x % $y

- `Comparison`

Operator	*Name*	*Example*	*Result*
==	Equal	$x == $y	Returns true if $x is equal to $y
===	Identical	$x === $y	Returns true if $x is equal to $y, and they are of the same type
!=	Not equal	$x != $y	Returns true if $x is not equal to $y
<>	Not equal	$x != $y	Returns true if $x is not equal to $y
!==	Not identical	$x !== $y	Returns true if $x is not equal to $y, or they are not of the same type
>	Greater than	$x > $y	Returns true if $x is greater than $y
<	Less than	$x < $y	Returns true if $x is less than $y
>=	Greater than or equal to	$x >= $y	Returns true if $x is greater than or equal to $y
<=	Less than or equal to	$x <= $y	Returns true if $x is less than or equal to $y

- `Increment/Decrement`

Operator	*Name*	*Description*
++$x	Pre-increment	Increments $x by one, then returns $x
$x++	Post-increment	Returns $x, then increments $x by one
--$x	Pre-decrement	Decrements $x by one, then returns $x
$x--	Post-decrement	Returns $x, then decrements $x by one

- `logical`

Operator	*Name*	*Example*	*Result*
and	And	$x and $y	True if both $x and $y are true
or	Or	$x or $y	True if either $x or $y is true
xor	Xor	$x xor $y	True if either $x or $y is true, but not both
|&&	And	$x && $y	True if both $x and $y are true
||	Or	$x || $y	True if either $x or $y is true
!	Not	!$x	True if $x is not true

Conditional statements

- Case

```php
<?php
$today = "Tue";

switch ($today) {
   case "Mon":
      echo "Today is Monday. Wake up early in the morning.";
      break;
   case "Tue":
      echo "Today is Tuesday. Do your homework.";
      break;
   case "Wed":
      echo "Today is Wednesday. Play a game";
      break;

   case "Thu":
      echo "Today is Thursday. Its movie time";
      break;
```

> Note that there should be a `break`

- While loops

```php
<?php
$x = 2;

do {
   echo " $x <br>";
   $x +=2;
} while ($x <= 10);
?>
```

Output

2
4
6
8
10

Example 02:

```php
<?php
$x = 6;

do {
   echo " $x";
   $x++;
} while ($x <= 5);
?>
```

Output
6

- For loops

```php
<?php
for ($x = 1; $x <= 10; $x++) {
   echo $x. " , ";
}
?>
```

Output
1 , 2 , 3 , 4 , 5 , 6 , 7 , 8 , 9 , 10

```php
<?php
for ($x = 1; $x <= 10; $x += 2) {
   echo $x. " , ";
}
?>
```

Output
1 , 3 , 5 , 7 , 9

- foreach loop

```php
<?php
$foods = array("Bread", "Chocolate", "Biscuit", "Ice-Cream");

foreach ($foods as $value) {
   echo "$value <br>";
}
?>
```

Output
Bread
Chocolate
Biscuit
Ice-Cream

DB connection

```php
<?php
$servername = "localhost";
$username = "root";
$password = "1234";
$database= "school";

// Create connection
$conn =mysqli_connect($servername, $username, $password, $database);

// Check connection
if ($conn === false) {
    die("Error : Connection failed: " . mysqli_connect_error());
```