Internal Assessment Test 2 – Jun 2022

Solution

Sub:	Web Technology & its Applications					Sub Code:	18CS63		Branch:	CSE		
Date:	09-06-2022	Duration:	90 mins	Max Marks:	50	Sem / Sec:	A, B & C	Time	8.30 - 10.00 am	OBE		

Answer any FIVE FULL Questions

		MARKS	CO	RBT
1 (a)	Explain different ways of-positioning elements in CSS layout techniques. It is possible to move an item from its regular position in the normal flow, and even move an item outside of the browser viewport so it is not visible or to position it so it is always visible in a fixed position while the rest of the content scrolls. The position property is used to specify the type of positioning, and the possible values are shown in Table.	[10]	CO2	L2

Value	Description
`absolute`	The element is removed from normal flow and positioned in relation to its nearest positioned ancestor.
`fixed`	The element is fixed in a specific position in the window even when the document is scrolled.
`relative`	The element is moved relative to where it would be in the normal flow.
`static`	The element is positioned according to the normal flow. **This is the default.**

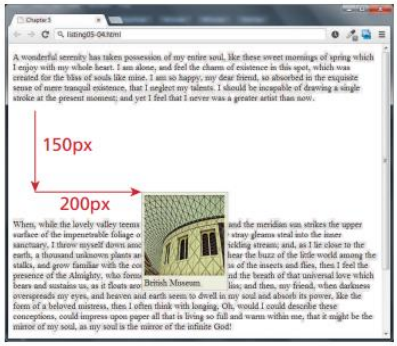
TABLE 5.1 Position Values

The left, right, top, and bottom properties are used to indicate the distance the element will move; the effect of these properties varies depending upon the position property. The next several sections will provide examples of how to use absolute, fixed, and relative positioning. While fixed position is used relatively infrequently, absolute and relative positioning are absolutely essential to many of the most common layout techniques in CSS. 5.2.1 Relative Positioning In relative positioning an element is displaced out of its normal flow position and moved relative to where it would have been placed. When an element is positioned relatively, it is displaced out of its normal flow position and moved relative to where it would have been placed. The other content around the relatively positioned element “remembers” the element’s old position in the flow; thus the space the element would have occupied is preserved as shown in Figure 5.

```
<p>A wonderful serenity has taken possession of my …

<figure>
   <img src="images/828.jpg" alt="" />
   <figcaption>British Museum</figcaption>
</figure>

<p>When, while the lovely valley …
```




```
figure {
   border: 1pt solid #A8A8A8;
   background-color: #EDEDDD;
   padding: 5px;
   width: 150px;
   position: relative;
   top: 150px;
   left: 200px;
}
```

FIGURE 5.4 Relative positioning

Absolute Positioning

When an element is positioned absolutely, it is removed completely from normal flow. Thus, unlike with relative positioning, space is not left for the moved element, as it is no longer in the normal flow. Its position is moved in relation to its container block. In the example shown in Figure 5.5,

```
<p>A wonderful serenity has taken possession of my …

<figure>
   <img src="images/828.jpg" alt="" />
   <figcaption>British Museum</figcaption>
</figure>

<p>When, while the lovely valley …
```

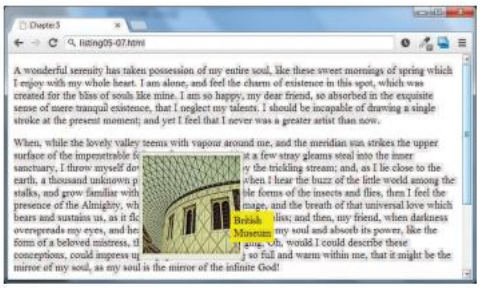

```
figure {
   margin: 0;
   border: 1pt solid #A8A8A8;
   background-color: #EDEDDD;
   padding: 5px;
   width: 150px;
   position: absolute;
   top: 150px;
   left: 200px;
}
```

FIGURE 5.5 Absolute positioning

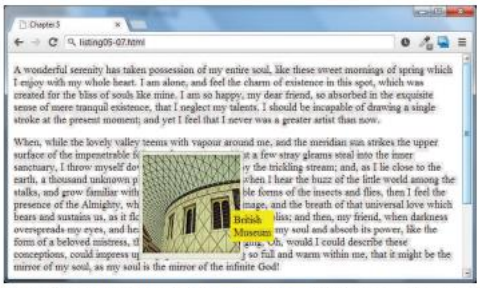
the container block is the element. Like with the relative positioning example, the moved block can now overlap content in the underlying normal flow.

Z-Index

Looking at Figure 5.6, you may wonder what would have happened if the <figcaption> had been moved so that it overlapped the <figure>. Each positioned element has a stacking order defined by the z-index property (named for the z-axis). Items closest to the viewer (and thus on the top).
Fixed Position The fixed position value is used relatively infrequently. It is a type of absolute positioning, except that the positioning values are in relation to the viewport (i.e., to the browser window). Elements with fixed positioning do not move when the user scrolls up or down the page, as can be seen in Figure 5.8



```
figure {
   position: absolute;
   top: 150px;
   left: 200px;
}
figcaption {
   position: absolute;
   top: 90px;
   left: 140px;
}
```



```
figure {
   ...
   z-index: 5;
}
figcaption {
   ...
   z-index: 1;
}
```

Note that this did **not** move the `<figure>` on top of the `<figcaption>` as one might expect. This is due to the nesting of the caption within the figure.

Fixed Position

The fixed position value is used relatively infrequently. It is a type of absolute positioning, except that the positioning values are in relation to the viewport (i.e., to the browser window). Elements with fixed positioning do not move when the user scrolls up or down the page.

```
figure {
   ...
   position: fixed;
   top: 0;
   left: 0;
}
```

Notice that figure is fixed in its position regardless of what part of the page is being viewed.

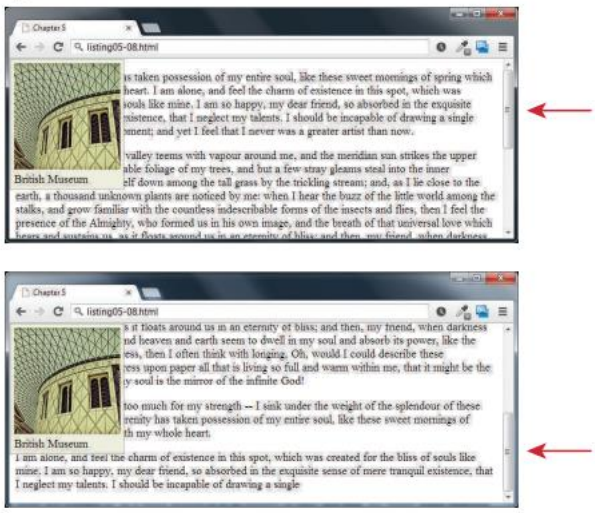
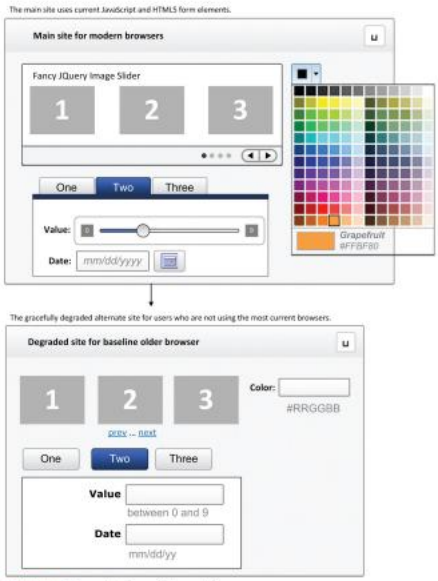
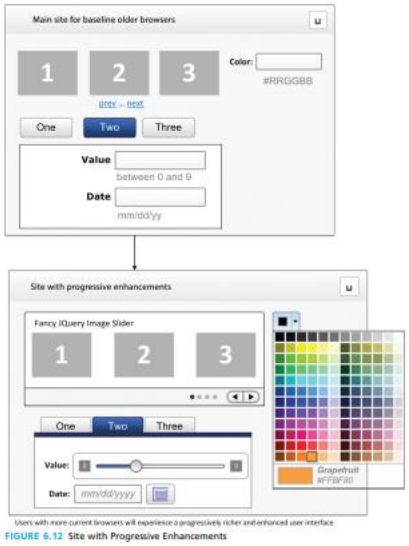
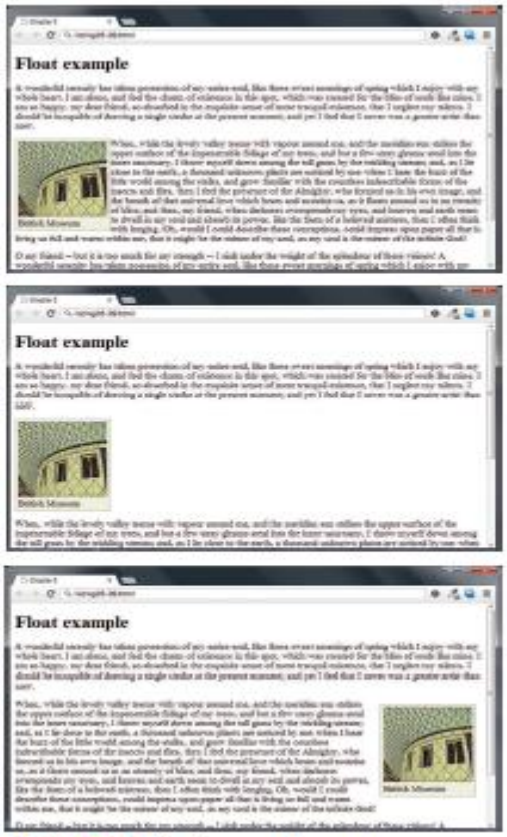


FIGURE 5.8 Fixed position

2 (a)	Write short notes on graceful degradation and progressive enhancement	[05]	CO2	L2

	The principle of graceful degradation is one possible strategy. With this strategy you develop your site for the abilities of current browsers. For those users who are not using current browsers, you might provide an alternate site or pages for those using older browsers that lack the JavaScript (or CSS or HTML5) used on the main site. The idea here is that the site is "degraded" (i.e., loses capability) "gracefully" (i.e., without pop-up JavaScript error codes or without condescending messages telling users to upgrade their browsers). Figure 6.11 illustrates the idea of graceful degradation.  FIGURE 6.11 Example of graceful degradation The alternate strategy is progressive enhancement, which takes the opposite approach to the problem. In this case, the developer creates the site using CSS, JavaScript, and HTML features that are supported by all browsers of a certain age or newer. (Eventually, one does have to stop supporting ancient browsers; many developers have, for instance, stopped supporting IE 6.) To that baseline site, the developers can now "progressively" (i.e., for each browser) "enhance" (i.e., add functionality) to their site based on the capabilities of the users' browsers.  FIGURE 6.12 Site with Progressive Enhancements			
(b)	Discuss the advantages and disadvantages of client-side scripting. **The advantages of client-side scripting are:**	[05]	CO3	L2

	• Processing can be offloaded from the server to client machines, thereby reducing the load on the server. • The browser can respond more rapidly to user events than a request to a remote server ever could, which improves the user experience. • JavaScript can interact with the downloaded HTML in a way that the server cannot, creating a user experience more like desktop software than simple HTML ever could. **The disadvantages of client-side scripting are mostly related to how programmers use JavaScript in their applications. Some of these include:** ■ There is no guarantee that the client has JavaScript enabled, meaning any required functionality must be housed on the server, despite the possibility that it could be offloaded. ■ The idiosyncrasies between various browsers and operating systems make it difficult to test for all potential client configurations. What works in one browser, may generate an error in another. ■ JavaScript-heavy web applications can be complicated to debug and maintain. JavaScript has often been used through inline HTML hooks that are embedded into the HTML of a web page. Although this technique has been used for years, it has the distinct disadvantage of blending HTML and JavaScript together, which decreases code readability, and increases the difficulty of web development.			
3 (a)	What does floating an element do in CSS? How do you float an element? **Floating Elements:** • It is possible to displace an element out of its position in the normal flow via the CSS float **property**. • An element can be floated to the left or floated to the right. • When an item is floated, it is moved all the way to the far left or far right of its containing block and the rest of the content is "re-flowed" around the floated element, as can be seen in Figure 5.9. • Notice that a floated block-level element must have a width specified; if you do not, then the width will be set to auto, which will mean it implicitly fills the entire width of the containing block, and there thus will be no room available to flow content around the floated item. Also note in the final example in Figure 5.9 that the margins on the floated element are respected by the content that surrounds the floated element.	[05]	CO2	L2




```
<h1>Float example</h1>
<p>A wonderful serenity has taken ...</p>
<figure>
   <img src="images/B28.jpg" alt="" />
   <figcaption>British Museum</figcaption>
</figure>
<p>When, while the lovely valley …</p>

figure {
   border: 1pt solid #A8A8A8;
   background-color: #EDEDDD;
   margin: 0;
   padding: 5px;
   width: 150px;
}
```

Notice that a floated block-level element must have a width specified.

```
figure {
   …
   width: 150px;
   float: left;
}
```

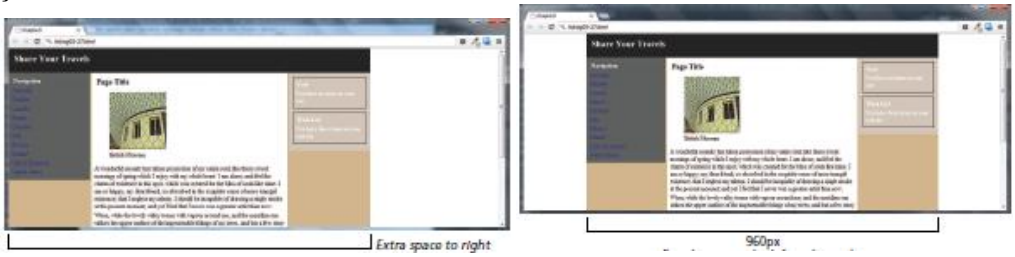
```
figure {
   …
   width: 150px;
   float: right;
   margin: 10px;
}
```

FIGURE 5.9 Floating an element

(b)	Write short notes on CSS Layout. • One of the main problems faced by web designers is that the size of the screen used to view the page can vary quite a bit. • Some users will visit a site on a 21-inch wide screen monitor that can display 1920 × 1080 pixels (px); others will visit it on an older iPhone with a 3.5 screen and a resolution of 320 × 480 px. • Users with the large monitor might expect a site to take advantage of the extra size; users with the small monitor will expect the site to scale to the smaller size and still be usable. • Satisfying both users can be difficult; the approach to take for one type of site content might not work as well with another site with different content. • Basic models: A. Fixed Layout: • In a **fixed layout**, the basic width of the design is set by the designer. • A common width used is something in the 960 to 1000 pixel range, which fits nicely in the common desktop monitor resolution (1024 × 768). • This content can then be positioned on the left or the center of the monitor. • Fixed layouts are created using pixel units, typically with the entire content within a container whose width property set to some width. <body> <header> ... </header> ...	[05]	CO3	L2

```
</div>
<footer>
...
</footer>
</div>
</body>


div#wrapper {
width: 960px;
background_color: tan;
}
```



The advantage of a fixed layout is that

a) It is easier to produce and generally has a predictable visual result.
b) It is also optimized for typical desktop monitors.

Fixed layouts have drawbacks:

c) For larger screens, there may be an excessive amount of blank space to the left and/or right of the content.
d) Much worse is when the browser window shrinks below the fixed width; the user will have to horizontally scroll to see all the content.

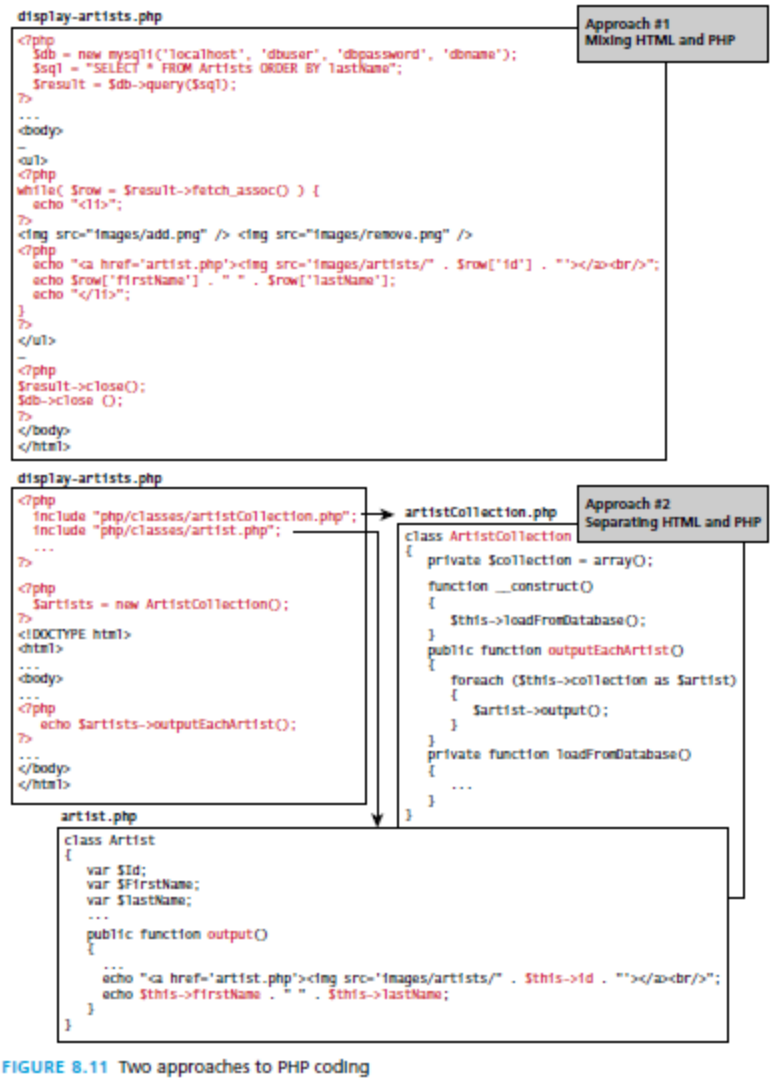
B. Liquid layout:

- In this approach, widths are not specified using pixels, but percentage values.
- Percentage values in CSS are a percentage of the current browser width, so a layout in which all widths are expressed as percentages should adapt to any browser size.

Advantage of a liquid layout

- It adapts to different browser sizes, so there is neither wasted white space nor any need for horizontal scrolling.

Disadvantage of a liquid layout

	• Liquid layouts can be more difficult to create because some elements, such as images, have fixed pixel sizes. • Another problem will be noticeable as the screen grows or shrinks dramatically, in that the line length (which is an important contributing factor to readability) may become too long or too short.			
4 (a)	With relevant code segments, explain two approaches to embed PHP script in HTML.  FIGURE 8.11 Two approaches to PHP coding	[04]	CO4	L2
(b)	Explain the following CSS properties with suitable examples: i) display ii) visibility iii) overflow **i) float** It is possible to displace an element out of its position in the normal flow via the CSS float **property**. An element can be floated to the left or floated to the right. When an item is floated, it is moved all the way to the far left or far right of its containing block and the rest of the content is "re-flowed" around the floated element. Notice that a floated block-level element must have a width specified; if you do not, then the width will be set to auto, which will mean it implicitly fills the entire width of the containing block, and there thus will be no room available to flow content around the floated item. **ii) position** The position property is used to specify the type of positioning, and the possible values are shown in Table 5.1. The left, right, top, and bottom properties are used to	[06]	CO2	L2

	indicate the distance the element will move; the effect of these properties varies depending upon the position property. absolute The element is removed from normal flow and positioned in relation to its nearest positioned ancestor. fixed The element is fixed in a specific position in the window even when the document is scrolled. relative The element is moved relative to where it would be in the normal flow. static The element is positioned according to the normal flow. **This is the default**. **iii)overflow**  FIGURE 3.23 Overflow property			
5 (a)	Explain 2 methods in Java Script to access DOM nodes with examples. In addition to these moderately useful properties, there are some essential methods (see Table 6.4) you will use all the time. They include `getElementByTagName()` and the indispensable `getElementById()`. While the former method returns an array of DOM nodes (called a `NodeList`) matching the tag, the latter returns a single DOM element (covered below), that matches the id passed as a parameter as illustrated in Figure 6.19.  FIGURE 6.19 Relationship between HTML tags and getElementByID() and getElementsByTagName()	[05]	CO3	L2

Method	Description
`createAttribute()`	Creates an attribute node
`createElement()`	Creates an element node
`createTextNode()`	Creates a text node
`getElementById(id)`	Returns the element node whose `id` attribute matches the passed `id` parameter
`getElementsByTagName(name)`	Returns a NodeList of elements whose tag name matches the passed `name` parameter

(b) Write a Javascript code that displays text "CORONA VIRUS" with increasing font size in the interval of 100 ms in blue color, when font size reaches 50 pt in teal color and should stop. [05] CO3 L3

```
<!DOCTYPE HTML>
<html>
<head>
<title>Program</title>
</head>
<body>
        <p id="demo"></p>
        <script>
                var var1 = setInterval(inTimer, 100);
                var fs = 5;
                var ids = document.getElementById("demo");
                function inTimer() {
                        ids.innerHTML = ' CORONA VIRUS;
                        ids.setAttribute('style', "font-size: " + fs + "px; color:
                        blue");
                        fs += 5;
                        if(fs >= 50 ){
                                clearInterval(var1);
                                var2 = setInterval(deTimer, 100);
                        }
                }
                function deTimer() {
                        fs -= 5;
                        ids.innerHTML = “CORONA VIRUS”;
                        ids.setAttribute('style', "font-size: " + fs + "px; color: teal");
                        if(fs === 5 ){
                                clearInterval(var2);
                        }
                }
        </script>
</body>
</html>
```

6 (a)	Briefly explain function in PHP with suitable examples	[10]	CO4	L2

Functions

In PHP there are two types of function:

1. User-defined functions

2. Built-in functions.

A **user-defined function** is one that you the programmer define. A **built-infunction** is one of the functions that come with the PHP environment

Function Syntax

To create a new function you must think of a name for it, and consider what it willdo. Functions can return values to the caller, or not return a value. They can be set upto take or not take parameters.

```
function getNiceTime() {
return date("H:i:s");
}
```

The definition of a function to return the current time as a string

```
function outputFooterMenu() {
echo '<div id="footer">';
echo '<a href=#>Home</a> | <a href=#>Products</a> | ';
echo '<a href=#>About us</a> | <a href=#>Contact us</a>';
echo '</div>';
}
```

The definition of a function without a return value

Calling a Function

To call a function you must use its name with the () brackets. Since getNiceTime()returns a string, you can assign that return value to a variable, or echo that returnvalue directly, as shown below.

```
$output = getNiceTime();
echo getNiceTime();
```

If the function doesn't return a value, you can just call the function: outputFooterMenu();

Parameters

It is more common to define functions with parameters, since functions are more powerful and reusable when their output depends on the input they get. **Parameters** are the mechanism by which values are passed into functions, and there are some complexities that allow us to have multiple parameters, default values, and to pass objects by reference instead of value.

To define a function with parameters, you must decide how many parameters you want to pass in, and in what order they will be passed. Each parameter must be named.

```
function getNiceTime($showSeconds) {
if ($showSeconds==true)
return date("H:i:s");
else
return date("H:i");
}
```

A function to return the current time as a string with an integer parameterThus to call our function, you can now do it in two ways:

```
echo getNiceTime(1); // this will print seconds
echo getNiceTime(0); // will not print seconds
```

In fact any nonzero number passed in to the function will be interpreted as true since the parameter is not type specific.

Parameter Default Values

```
function getNiceTime($showSeconds=1){
if ($showSeconds==true)
return date("H:i:s");
else
return date("H:i");
}
```

A function to return the current time with a parameter that includes a default. If you do include a value in your function call, the default will beoverridden by whatever that value was.

Passing Parameters by Reference

By default, arguments passed to functions are **passed by value** in PHP.

```
function changeParameter($arg) {
$arg += 300;
echo "<br/>arg=" . $arg;
}
$initial = 15;
echo "<br/>initial=" . $initial; // output: initial=15
changeParameter($initial); // output: arg=315
echo "<br/>initial=" . $initial; // output: initial=15
```

Passing a parameter by value

The mechanism in PHP to specify that a parameter is passed byreference is to add an ampersand (&) symbol next to the parameter name in thefunction declaration.

```
function changeParameter(&$arg) {
$arg += 300;
echo "<br/>arg=". $arg;
}
$initial = 15;
echo "<br/>initial=" . $initial; // output: initial=15
changeParameter($initial); // output: arg=315
echo "<br/>initial=" . $initial; // output: initial=315
```

Passing a parameter by reference

Variable Scope within Functions

It will come as no surprise that all variables defined within a function (such asparameter variables) have **function scope**, meaning that they are only accessiblewithin the function.

```
$count= 56;
function testScope() {
echo $count; // outputs 0 or generates run-time warning/error
}
testScope();
echo $count; // outputs 56
```

While variables defined in the main script are said to have **global scope**,

```
$count= 56;
function testScope() {
global $count;
echo $count; // outputs 56
}
testScope();
echo $count; // outputs 56
```

Using the global keyword

CO PO Mapping

Course Outcomes		Modules covered	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3	PSO4
CO1	Apply appropriate technologies to create an interactive dynamic webpage	1,2,3,4, 5	2	2	2	-	3	2	-	-	2	-	2	2	3	-	-	2
CO2	Summarize advanced dynamic web projects using client-Server technologies.	1,2,3,4, 5	2	2	2	-	3	2	-	-	2	-	2	2	3	-	-	2
CO3	Demonstrate ability to adapt to changing web development and design skills.	1,2,3,4, 5	2	2	2	-	3	2	-	-	2	-	2	2	3	-	-	2
CO4	Analyze and develop a client server application using appropriate technologies considering performance.	1,2,3,4, 5	2	2	2	-	3	2	-	-	2	-	2	2	3	-	-	2
CO5	Inspect JavaScript frameworks like jQuery and Backbone which facilitates developer to focus on core features	5	2	2	2	-	3	2	-	-	2	-	2	2	3	-	-	2

COGNITIVE LEVEL	REVISED BLOOMS TAXONOMY KEYWORDS
L1	List, define, tell, describe, identify, show, label, collect, examine, tabulate, quote, name, who, when, where, etc.
L2	summarize, describe, interpret, contrast, predict, associate, distinguish, estimate, differentiate, discuss, extend
L3	Apply, demonstrate, calculate, complete, illustrate, show, solve, examine, modify, relate, change, classify, experiment, discover.
L4	Analyze, separate, order, explain, connect, classify, arrange, divide, compare, select, explain, infer.
L5	Assess, decide, rank, grade, test, measure, recommend, convince, select, judge, explain, discriminate, support, conclude, compare, summarize.

PROGRAM OUTCOMES (PO), PROGRAM SPECIFIC OUTCOMES (PSO)				CORRELATION LEVELS	
PO1	Engineering knowledge	PO7	Environment and sustainability	0	No Correlation
PO2	Problem analysis	PO8	Ethics	1	Slight/Low
PO3	Design/development of solutions	PO9	Individual and team work	2	Moderate/ Medium
PO4	Conduct investigations of complex problems	PO10	Communication	3	Substantial/ High
PO5	Modern tool usage	PO11	Project management and finance		
PO6	The Engineer and society	PO12	Life-long learning		
PSO1	Develop applications using different stacks of web and programming technologies				
PSO2	Design and develop secure, parallel, distributed, networked, and digital systems				
PSO3	Apply software engineering methods to design, develop, test and manage software systems.				
PSO4	Develop intelligent applications for business and industry				