

➤ **Program 5.5.3 :** Design a webpage to handle asynchronous requests using AJAX on mouseover event.

Solution :

- You will have to just change button event to on mouseover instead of on click

```
<button type="button" onmouseover="showFile()">Show Text File Content</button>
```

Review Questions

- Q. 1 Design a webpage to display student information in tubular format using AJAX.
- Q. 2 Write an AJAX application to display result of a student's based on marks entered out of 1000.
- Q. 3 Write an AJAX application to perform simple arithmetic operation.

CHAPTER

6

PHP

Unit III

University Prescribed Syllabus

Variables and Operators, Program Flow, Arrays, Working with Files and Directories, Working with Databases, Working with Cookies, Sessions and Headers.

Introduction

- The PHP Hypertext Preprocessor (PHP) is a programming language that allows web developers to create dynamic content that interacts with databases.
- PHP is basically used for developing web based software applications. Open Source scripting language, most commonly used on web servers to produce dynamic pages.

Syllabus Topic : Variables and Operators

6.1 Variables and Operators

6.1.1 Variable

Define Variable. Explain with example.

- A variable is an identifier which is used to store a value. It is always necessary to declare a variable before using it in a program.
- Variable start with a \$ sign.
- = operator is used to assign value to a variable.
- It is not necessary to declare the variable.
- A variable need not know the type of value that will be assigned to it prior.

- PHP does a good job of automatically converting types from one to another when necessary.

Rules -Naming Variables

- Should begin with a dollar sign (\$).
- After the \$ sign it should be either letter or underscore followed by any character, no limit in the length of the variable.
- It is case - sensitive (\$A or \$a are two distinct variables), so it's worth sticking to one variable naming method. For example, always using lowercase to avoid mistakes.

Examples

```
$m
$Variable
$x
$_123
```

It is not necessary to declare the variable before it is used. Also we can initialise values while we declare the variable.

```
$i=10;
```

We can understand from the above declaration the value stored in the variable decides the type of the variable.

PHP supports the following datatypes :

1. **Integers** : are whole numbers, without a decimal point, like 100.
2. **Doubles** : are floating-point numbers, like 3.14159 or 149.11.
3. **Booleans** : have only two possible values either true or false.
4. **NULL** : is a special type that only has one value: NULL.
5. **Strings** : are sequences of characters, like 'PHP supports string operations'.
6. **Arrays** : are named and indexed collections of other values.
7. **Objects** : are instances of programmer-defined classes, which can package up both other kinds of values and functions that are specific to the class.
8. **Resources** : are special variables that hold references to resources external to PHP (such as database connections).

6.1.2 Operators

What is an operator ? Explain its types.

These are special characters which perform some action on values or operands associated. Based on the operation performed they are classified into following types :

- | | |
|------------------------------------|--------------------------------------|
| 1. Arithmetic operator | 2. Comparison or Relational operator |
| 3. Logical operator | 4. Assignment operator |
| 5. Conditional or ternary operator | |

Note : Operators are also classified based on the number of operands like unary, binary or ternary.

1. Arithmetic Operators

This operator basically used to calculate and perform action on numeric values. It is classified under the binary category as it always takes two operands.

The Table 6.1.1 shows the list of Arithmetic operators :

Table 6.1.1

Operator	Description	Example
+	Addition	a + b 10 + 20 = 30
-	Subtraction	x - y 20 - 5 = 15
*	Multiplication	s * p 12 * 3 = 36
/	Division	n/m 12/3=4 (Gives the quotient)
%	Modulus	a % b 12%4=0 (Gives the remainder)

The following are also called as arithmetic operator but categorized under unary operator since it takes only one operand.

Operator	Description	Example
++	Increment the value by 1	x++ Increase the value of x by 1 and stores it back in x
--	Decrements the value by 1	a-- decreases the value by 1 and stores it back in a

2. Comparison or Relational Operators

- It is used to compare values. It returns a Boolean value either **True** or **false**, this operator also comes under the binary operator's category.
- The Table 6.1.2 shows the list of comparison operators :

Table 6.1.2

Operators	Description	Example
==	Equal to	A==b Returns true if both the values are same, false otherwise.
!=	Not equal to	A!=b Returns true if both the values are different, false otherwise
<	Less than	S<b Returns true if the value of S is less compared to b
>	Greater than	s>m Returns true if s>m else false.
<=	Less than equal to	g<=y Returns true if g is less than or equal to y.
>=	Greater than equal to	k>=p returns true if k is bigger or equal to p otherwise false.

3. Logical Operators

This operator too is used with conditional statements. It returns either true or false. The Table 6.1.3 shows the list of Logical operators :

Table 6.1.3

Operators	Description	Example
&&	AND	A && B Returns true if both the operands are true otherwise false.
	OR	A B Returns true if any one or both the operands is true otherwise false.
!*	NOT	!A Returns false if operand is true or true otherwise.

Note : *Unary operator

4. Assignment Operator

It is classified into two types :

- Simple assignment operator
- Compound assignment operator

The main function of this operators is used to assign values.

(a) Simple Assignment Operator

"=" is the operator which is used to assign value to a variable.

Example

```
c=a + b;
```

(b) Compound Assignment Operator

This operator is basically a combination of simple assignment operator and the other operators. We can use this operator only when the variable used in left hand side is also there in the right hand side.

Operator	Description	Example
+=, -, =, *=, /=, %=	Performs addition and assignment respectively. Same with other operators	a+=5 ie a=a+5 Increase the value of a by 5 and stores back in a.

5. Conditional or Ternary Operator

This operator comes under the category of ternary operator, since it takes three operands. It is represented as `?:`. The expression is evaluated to true or false, depending on the value any of the given two statements is executed.

Example

```
c = (a > b ? a : b);
```

In the above statement c is assigned a if `a > b` otherwise b.

Syllabus Topic : Program Flow

6.2 Control Flow Statement

Q What is a control flow statement. Give its types.

Generally a program is executed sequentially one after another but there might be situations where the statements need to be executed only based on some condition.

Java script supports two types of statements :

1. Selection statement
2. Looping statement

6.2.1 Selection Statement

Q Write on If-else, If-else-if with example.

The statements are executed only if the condition is satisfied. We have following types of selection statements :

1. if statement
2. if-else statement
3. if-else-if statement
4. switch case

1. If statement

This is also called as conditional statement. The Fig. 6.2.1 depicts the working of if statement.

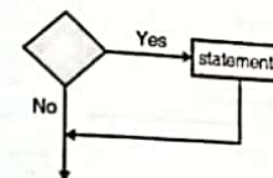


Fig. 6.2.1

The statements are executed only when the condition is true.

Syntax

```
if(condition)
{
  Statements;
}
```

Example

```
<html>
<body>

<?php
$d=10;
if($d > 0 )
echo "Positive";
?>

</body>
</html>
```

Output

Positive

2. If-else statement

- This statement is similar to if statement but the only difference is different set of statements are executed based on the condition whether it is true or false.

- The Fig. 6.2.2 depicts the working of if - else statement.

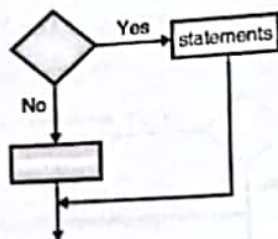


Fig. 6.2.2

Syntax

```

if(expression)
{
  Statements;
}
else
{
  Statements;
}
  
```

Example

```

<?php
$s="FY";
if($s=="FY")
    echo "Selected";
else
    echo "Rejected";
?>
  
```

Output

Selected

3. If-else-If Statement

- When we have to check multiple conditions if-else if statement can be used. The Fig. 6.2.3 depicts the working of if - else- if statement.

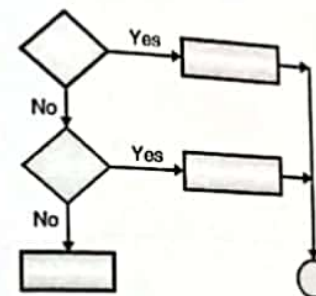


Fig. 6.2.3

Syntax

```

if(expression1)
{
  Statement-1;
}
else if (expression2)
{
  Statement-2;
}
else if(expression3)
{
  Statement-3;
}
else
{
  Statement-4;
}
  
```

Example

```

<?php
$d = "FY";
if($d == "FY")
    echo "Welcome FY!";
  
```

```
elseif($d == "SY")
    echo "Congrats !";

else
    echo " Sorry Detained ";
?>
```

Output

Welcome FY!

4. switch case

- switch statement is to give an expression to evaluate and several different statements to execute based on the value of the expression.
- The interpreter checks each case against the value of the expression until a match is found. If nothing matches, a default condition will be used.

Syntax

```
switch (expression)
{
    case condition 1: statement(s);
    break;
    case condition 2: statement(s);
    break;
    ...
    case condition n: statement(s);
    break;
    default: statement(s);
}
```

- break statement will terminate from the switch case statement, if omitted all the case statements will be executed.

Example

```
<?php
$color="r";
switch ($color)
{
```

```
case 'r': echo "Red";
    break;
case 'b': echo "Blue";
    break;
case 'g': echo "green";
    break;
default: echo "Wrong Entry";
}
?>
```

Output

Red

6.2.2 Looping Statement

Write on entry checking loop with example.

Explain for loop with example.

When we want to execute statement's more than once looping statements are used. The looping statements are basically divided into three parts :

- Initialization or starting point
- Condition or terminating point
- Increment or decrement value

The following are the types of looping systems;

- | | |
|----------|--------------|
| 1. while | 2. do -while |
| 3. for | 4. foreach |

1. while

It is also called entry checking loop. The statements are defined inside the body of the loop and is executed till the condition defined is true. The loop will be executed zero or more number of times.

Flow chart

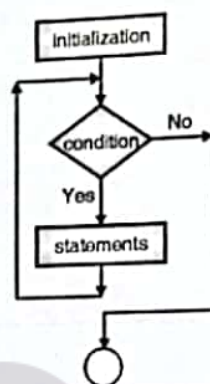


Fig. 6.2.4

Syntax

```

initialization;
while(expression)
{
    statements;
    increment or decrement;
}
  
```

Example

```

$i=1;
while($i<=10)
{
    echo " the value is $i";
    $i++;
}
  
```

The above loop will be executed 10 times and then terminates.

2. do -while

This is called exit checking loop. The statements are defined inside the body of the loop and are executed till the condition defined is true. The loop will be executed at least once.

Flow chart

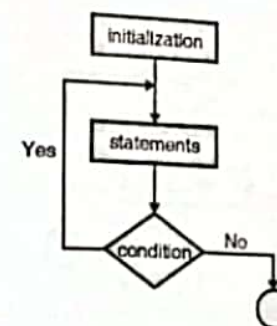


Fig. 6.2.5

Syntax

```

initialization;
do
{
    statements;
}while(condition);
  
```

Example

```

$i=1;
do
{
    echo "i=$i";
    $i++;
}while($i<=10);
  
```

Since the condition is checked at the end the above loop is executed eleven times.

3. for

One of the simplest loop as it is easy to implement. Initialization, condition and increment or decrement is defined on the same statement.

Flow chart

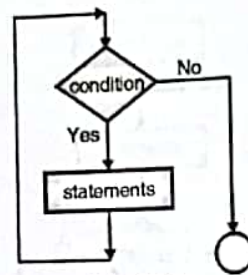


Fig. 6.2.6

Syntax

```

for(initialization; condition; increment/decrement)
{
    statements;
}
  
```

Example

```

for($i=1;$i<=10;$i++)
{
    echo "i=$i";
}
  
```

The above loop is executed ten times.

4. foreach

The foreach statement is used to loop through arrays. Each value from the array element is assigned to \$value and the array pointer is moved by one and the next element will be processed.

Syntax

```

foreach (array as value) {
    code to be executed;
}
  
```

Example

```

<html>
<body>
  
```

```

<?php
    $ar=array(1,2,3,4,5);
    foreach( $aras $val)
    {
        echo"Value is $val<br />";
    }
?>
</body>
</html>
  
```

Syllabus Topic : Arrays

6.3 Arrays

Define array. Give its types.

It is a set of consecutive memory location which is used to store values of a same type. It is a data structure which is used to store homogeneous values.

Arrays are classified into three types based on index. The index is used to access the individual elements in the array.

1. **Numeric array** : Index is a number. Generally it starts from 0.
2. **Associative array** : Index is a string.
3. **Multi dimension array** : We have multiple index values.

6.3.1 Numeric Array

It can store any kind of values and the values are stored in linear fashion. We use array() to create arrays.

Example

```

<?php
    $numb =array(1,2,3,4,5);

    foreach( $numb as $v ){
        echo"Value=$v";
    }
  
```



```
/*storing string in an array?
$numbers[0]="one";
$numbers[1]="two";
$numbers[2]="three";
$numbers[3]="four";
$numbers[4]="five";
```

```
foreach( $numbers as $val){
echo"Value is $val<br />";
}
?>
```

6.3.2 Associative Array

Associative array will have their index as string. This helps to have a strong association with key or the index and the value.

```
<?php
/* First method to associate create array. */
$salaries =array("X"=>2000,"Y"=>1000,"Z"=>500);

echo"Salary of X is ". $salaries['X']."<br />";
echo"Salary of Y is ". $salaries['Y']."<br />";
echo"Salary of Z is ". $salaries['Z']."<br />";

/* Second method to create array. */
$salaries['x']="high";
$salaries['r']="medium";
$salaries['a']="low";
```

```
echo"Salary of x is ". $salaries['x']."<br />";
echo"Salary of r is ". $salaries['r']."<br />";
echo"Salary of a is ". $salaries['a']."<br />";
?>
```

6.3.3 Multi-Dimensional Array

A multi-dimensional array each element in the main array can also be an array. And each element in the sub-array can be an array, and so on. Values in the multi-dimensional array are accessed using multiple index.

Example

```
<?php
$marks =array(
"jay"=> array (
"c"=>35,
"c++"=>30,
"php"=>39
),
"ajay"=> array (
"c"=>30,
"c++"=>32,
"php"=>29
),
"vijay"=> array (
"c"=>31,
"c++"=>22,
"php"=>39
)
);

/* Accessing multi-dimensional array values */
echo"Marks for jay in c : ";
echo $marks['jay']['c']."<br />";

echo"Marks for ajay in c++ : ";
echo $marks['ajay']['c++']."<br />";
```

```
echo "Marks for vijay in php : ";
echo $marks['vijay']['php']."<br />";
?>
```


Syllabus Topic : Working with Files

6.4 Working with Files

- What we store in a hard drive is stored in the form of files inside directories. PHP provides ways to handle any kind of files. A file is basically a sequence of bytes.
- We can perform the following operation on file :

- | | |
|---------|----------|
| 1. Open | 2. Close |
| 3. Read | 4. Write |

6.4.1 Opening and Closing Files

 How to create a file, what are the modes in which a file can be opened ?

- `fopen()` is the function which is used to open the file. It takes two arguments file name and then mode in which to operate.
- Files modes can be specified as one of the six options in Table 6.4.1.


Table 6.4.1

Mode	Purpose
r	Opens the file for reading only. Places the file pointer at the beginning of the file.
r+	Opens the file for reading and writing. Places the file pointer at the beginning of the file.
w	Opens the file for writing only. Places the file pointer at the beginning of the file and truncates the file to zero length. If files does not exist then it attempts to create a file.

Mode	Purpose
w+	Opens the file for reading and writing only. Places the file pointer at the beginning of the file and truncates the file to zero length. If files does not exist then it attempts to create a file.
a	Opens the file for writing only. Places the file pointer at the end of the file. If files does not exist then it attempts to create a file.
a+	Opens the file for reading and writing only. Places the file pointer at the end of the file. If files does not exist then it attempts to create a file.

- If an attempt to open a file fails then `fopen` returns a value of **false** otherwise it returns a **file pointer** which is used for further reading or writing to that file.
- After making a changes to the opened file it is important to close it with the `fclose()` function. The `fclose()` function requires a file pointer as its argument and then returns **true** when the closure succeeds or **false** if it fails.

6.4.2 Reading a File

 Give the use of `fgets()`, `fread()` functions with examples.

- Once a file is opened using `fopen()` we can read by using the `fread()` function. This function requires two arguments. These must be the file pointer and the length of the file in bytes.
- Size of the file can be obtained by the function `filesize()` which takes the file name as its argument and returns the size of the file expressed in bytes.
- So here are the steps required to read a file with PHP.
 - Open a file using `fopen()` function.
 - Get the file's length using `filesize()` function.
 - Read the file's content using `fread()` function.
 - Close the file with `fclose()` function.
- We also have the following function which helps to read value from the file
 - `fgets()` : read a line from the file

Example

```
$fp = fopen("c:\\file.txt", "r");//open file in read mode
echo fgets($fp);
```

2. fgetc() : read a character from the file**Example**

```
$fp = fopen("c:\\file1.txt", "r");//open file in read mode
while(!feof($fp)) {
    echo fgetc($fp);
}
```

The following example assigns the content of a text file to a variable then displays those contents on the web page.


Example

```
<?php
$filename = "first.txt";
$file = fopen($filename, "r");

if( $file == false ){
    echo("Error in opening file");
    exit();
}

$filesize = filesize( $filename );
$filetext = fread( $file, $filesize);
fclose( $file );
echo("<pre>$filetext</pre>");
?>
```

6.4.3 Writing a File

 Explain with example fputs(), fwrite().

- A new file can be written or text can be added by using fwrite(). This function takes three arguments specifying a file pointer and the string of data that is to be written.

- Optionally a third integer argument can be included to specify the length of the data to write.

- If the third argument is included, writing will stop after the specified length has been reached.

- The following example creates a new text file then writes a short text heading inside it. After closing this file its existence is confirmed using file_exists() function which takes file name as an argument.

fputs() is also used to write a single line into the file.

```
<?php
$filename = "newfile.txt";
$file = fopen( $filename, "w" );

if( $file == false ) {
    echo ( "Error in opening new file" );
    exit();
}

fwrite( $file, "This is a simple test\n" );
fclose( $file );
?>
```

Reading the above file

```
<html>
<head>
<title>Writing a file using PHP</title>
</head>
<body>
<?php
```

```

$filename = "newfile.txt";
$file = fopen( $filename, "r" );
if( $file == false ) {
    echo ( "Error in opening file" );
    exit();
}

$filesize = filesize( $filename );
```

```
$filetext = fread( $file, $filesize );
fclose( $file );
echo ( "$filetext" );
?>
```

Syllabus Topic : Working with Directories

6.5 Working with Directories

Discuss about directory function.

The directory functions help to access the information about directories and their contents.

The following are the list of directory functions in PHP shown in Table 6.5.1.

Table 6.5.1

Function	Description
chdir()	Changes the current directory.
chroot()	Changes the root directory
closedir()	Closes a directory handle
dir()	Returns an instance of the Directory class
getcwd()	Returns the current working directory
opendir()	Opens a directory handle
readdir()	Returns an entry from a directory handle
rewinddir()	Resets a directory handle
scandir()	Returns an array of files and directories of a specified directory

Syllabus Topic : Working with Databases

6.6 Working with Databases

PHP can be used to connect to the database. MySQL is the most popular database system used with PHP.

6.6.1 What is MySQL?

- It is a database system used on the web.

- Database system that runs on a server.
- It is used by small and large applications.
- It is very fast, reliable, and easy to use.
- MySQL uses standard SQL.
- It compiles on a number of platforms.
- Free to download and use.
- The data in a MySQL database are stored in tables. A table is a collection of related data, and it consists of columns and rows.

6.6.2 Database Queries

Give the steps to connect with a database.

How to insert, update a record in a database ?

- A query is a question or a request. We can query a database for specific information and have a record set returned.

Look at the following query (using standard SQL) :

SELECT LastName FROM Employees

- The query above selects all the data in the "LastName" column from the "Employees" table.
- In order to work with a database we have to do the following steps :

- | | |
|----------------------------|------------------------|
| 1. Connect to the database | 2. Create the database |
| 3. Insert | 4. Update |
| 5. Delete and | 6. View |
| 7. Close the connection | |

1. Connecting

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
```



```
// Create connection
$co = new mysqli($servername, $username, $password);

// Check connection
if ($co->connect_error) {
    die("Connection failed: " . $co->connect_error);
}
echo "Connected successfully";
?>
```

2. Create Database

The following is the code to create database. Before the creation of database it is necessary to open the connection.

```
$sql = "CREATE DATABASE myDB";
if ($conn->query($sql) === TRUE) {
    echo "Database created successfully";
} else {
    echo "Error creating database: " . $conn->error;
}
```

3. Insert

After a database and a table have been created, we can start adding data in them.

Here are some syntax rules to follow :

- The SQL query must be quoted in PHP.
- String values inside the SQL query must be quoted.
- Numeric values must not be quoted.
- The word NULL must not be quoted.
- The INSERT INTO statement is used to add new records to a MySQL table:
INSERT INTO table_name (column1, column2, column3,...)
VALUES (value1, value2, value3,...)

Example

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$sql = "INSERT INTO MyGuests (firstname, lastname, email)
VALUES ('Abc', 'Def', 'abc@yahoo.com')";

if ($conn->query($sql) === TRUE) {
    echo "New record created successfully";
} else {
    echo "Error: " . $sql . "<br>" . $conn->error;
}

$conn->close();
?>
```

4. View

The SELECT statement is used to select data from one or more tables :

SELECT column_name(s) FROM table_name

or we can use the * character to select ALL columns from a table :

SELECT * FROM table_name

The following example selects the id, firstname and lastname columns from the MyGuests table and displays it on the page.

Example

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$sql = "SELECT id, firstname, lastname FROM MyGuests";
$result = $conn->query($sql);

if ($result->num_rows > 0) {
    // output data of each row
    while($row = $result->fetch_assoc())
    {
        echo "id: " . $row["id"]. " - Name: " . $row["firstname"]. " " . $row["lastname"]. "<br>";
    }
}
else
{
    echo "0 results";
}
$conn->close();
?>
```

5. Delete

The DELETE statement is used to delete records from a table :

DELETE FROM table_name
WHERE some_column = some_value

Example

The following examples delete the record with id=3 in the "MyGuests" table .

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

// sql to delete a record
$sql = "DELETE FROM MyGuests WHERE id=3";

if ($conn->query($sql) === TRUE) {
    echo "Record deleted successfully";
} else {
    echo "Error deleting record: " . $conn->error;
}

$conn->close();
?>
```




6. Update

The UPDATE statement is used to update existing records in a table :

```
UPDATE table_name
SET column1=value, column2=value2,...
WHERE some_column=some_value
```

Example

```
<?php
$servername = "localhost";
$username = "username";
$password = "password";
$dbname = "myDB";

// Create connection
$conn = new mysqli($servername, $username, $password, $dbname);
// Check connection
if ($conn->connect_error) {
    die("Connection failed: " . $conn->connect_error);
}

$sql = "UPDATE MyGuests SET lastname='pat' WHERE id=2";

if ($conn->query($sql) === TRUE) {
    echo "Record updated successfully";
} else {
    echo "Error updating record: " . $conn->error;
}

$conn->close();
?>
```



7. Closing

```
$conn->close();
```

Syllabus Topic : Working with Cookies

6.7 Working with Cookies

What is a cookie ?

- Cookies are text files stored on the client computer and are used for tracking. It is a small file that the server embeds on the user's computer.
- Each time the same computer requests a page with a browser, it will send the cookie too. With PHP, you can both create and retrieve cookie values.

Cookie creation

A cookie is created with the `setcookie()` function.

Syntax

```
setcookie(name, value, expire, path, domain, secure, httponly);
```

Only the *name* parameter is required. All other parameters are optional.

6.7.1 PHP Create/Retrieve a Cookie

How to create a cookie ?

- The following example creates a cookie named "user" with the value "Jan.". The cookie will expire after 30 days (86400 * 30). The "/" means that the cookie is available in entire website (otherwise, select the directory you prefer).
- We then retrieve the value of the cookie "user" (using the global variable `$_COOKIE`). We also use the `isset()` function to find out if the cookie is set :

```
<?php
$cookie_name = "user";
$cookie_value = "Jan";
setcookie($cookie_name, $cookie_value, time() + (86400 * 30), "/"); // 86400 = 1 day
?>

<html>
<body>
```

```
<?php
if(!isset($_COOKIE[$cookie_name])) {
    echo "Cookie named '" . $cookie_name . "' is not set!";
} else {
    echo "Cookie '" . $cookie_name . "' is set!<br>";
    echo "Value is: " . $_COOKIE[$cookie_name];
}
?>

</body>
</html>
```

Note : The setcookie() function must appear BEFORE the <html> tag.

The value of the cookie is automatically URLencoded when sending the cookie, and automatically decoded when received (to prevent URLencoding, use setrawcookie() instead).

6.7.2 Modify a Cookie Value

How to modify a cookie ?

To modify a cookie, just set (again) the cookie using the setcookie() function:

Example

```
<?php
$cookie_name = "user";
$cookie_value = "Fan";
setcookie($cookie_name, $cookie_value, time() + (86400 * 30), "/");
?>
<html>
<body>

<?php
if(!isset($_COOKIE[$cookie_name])) {
    echo "Cookie named '" . $cookie_name . "' is not set!";
```

```
} else {
    echo "Cookie '" . $cookie_name . "' is set!<br>";
    echo "Value is: " . $_COOKIE[$cookie_name];
}
?>
</body>
</html>
```

6.7.3 Delete a Cookie

Give the function to destroy a cookie.

To delete a cookie, use the setcookie() function with an expiration date in the past :

Example

```
<?php
// set the expiration date to one hour ago
setcookie("user", "", time() - 3600);
?>
<html>
<body>

<?php
echo "Cookie 'user' is deleted.";
?>
</body>
</html>
```

Syllabus Topic : Working with Session

6.8 Working with Session

Define a session.

A session is a way to store information (in variables) to be used across multiple pages. Unlike a cookie, the information is not stored on the users computer.

- A session creates a file in a temporary directory on the server where registered session variables and their values are stored. This data will be available to all pages on the site during that visit.
- The location of the temporary file is determined by a setting in the `php.ini` file called `session.save_path`. Before using any session variable make sure you have setup this path.

6.8.1 Starting a PHP Session

How to start a session ?

- When we call the `session_start()` function a session is started. This function first checks if a session is already started and if not then a new session is started. It is recommended to put the call to `session_start()` at the beginning of the page.
- Session variables are stored in associative array called `$_SESSION[]`. These variables can be accessed during lifetime of a session.
- We can use `isset()` function to check if session variable is already set or not.

```
<?php
session_start();

if(isset($_SESSION['counter'])){
    $_SESSION['counter']+=1;
}else{
    $_SESSION['counter']=1;
}

$msg="You have visited this page ". $_SESSION['counter'];
$msg.="in this session.";
?>
```

<html>

<head>

<title>Setting up a PHP session</title>

</head>

<body>

<?php echo (\$msg);?>

</body>

</html>

6.8.2 Destroying a PHP Session

How to destroy a session ?

A PHP session can be destroyed by `session_destroy()` function. This function does not need any argument and a single call can destroy all the session variables. If you want to destroy a single session variable then you can use `unset()` function to unset a session variable.

```
<?php
unset($_SESSION['counter']);
?>
```

Example


```
<?php
unset($_SESSION['counter']);
?>
```

Here is the call which will destroy all the session variables :

```
<?php
session_destroy();
?>

<?php
unset($_SESSION['counter']);
?>
```

6.8.3 Turning on Auto Session

 How to turn on the session automatically ?

- It's not always necessary to start a session explicitly we can set the `session.auto_start` to `1` in `php.ini` file.
- This will automatically start the session.

Syllabus Topic : Working with Headers

6.9 Working with Headers

- It returns the raw http header.
- Generally, header is called before the actual output is sent.

Syntax

`header(string,replace,http_response_code)`

Parameter	Description
string	Required header string to send
replace	Optional. Whether to replace previous header or add a second header. Default is TRUE (will replace). FALSE (allows multiple headers of the same type)
http_response_code	Optional. Forces the HTTP response code to the specified value (available in PHP 4.3 and higher)

Example

```
<?php
header("Content-type:application/pdf");

//calling Download.pdf
header("Content-Disposition:attachment;filename='downloaded.pdf'");

//Pdf dource is original.pdf
readfile("original.pdf");
?>

<html>
<body>
```

Review Questions

- Q.1 Write a php program to find whether a number is odd or even.
- Q.2 Write a program to generate all even numbers between 1 to 100.
- Q.3 Write a program to print the records from a database.
- Q.4 How to hide a paragraph while clicking on the button show.

□□□