UNIT III: String Manipulation and Regular Expression

3.1 Creating and accessing String.

3.2 Searching & Replacing String.

3.3 Formatting, joining and splitting String.

3.4 String Related Library functions.

3.5 Use and advantage of regular expression over inbuilt function.

3.6 Friend Function.

3.1 Creating and accessing String.

String is a sequence of characters. Or String is and array of Character.

In PHP, a character is the same as a byte; therefore there are exactly 256 different characters possible. Long string is supported in PHP; there is no limit to the size of strings.

In PHP string can be specified in three different ways.

1. Single quoted
2. Double quoted
3. Heredocs.

Single quoted String:
A string specified in single quote (‘’) is called single quoted string.

Example:

```
<?php
$str='One line simple string.<br>';
print $str;
?>
```

Double quoted String

A string specified in Double quote (“”) is called double quoted string.
When we want to print some special characters or the values of variables within a string then we enclose the string with double-quotes(") character.

Example:

```
<?php
$str=”One line simple string.<br>”;
print $str;
?>
```

Heredocs Strings

Heredocs string is a long length string. If you want to use very long string in your PHP script then you can use this heredoc string.

Using heredoc string we need not use single or double quotation marks and inside string you can use single or double quotation marks without using escape

sequence character.

Syntax
Variable_name=<<<marker text …..
text ….. marker;

Where marker is any valid identifier used as an end marker.

Example : Heredocs

<?php
$s1=<<<cocsit

COCSIT is managed by “Royal” Education Society‟s

cocsit;

echo $s1; ?>

Output

COCSIT is managed by “Royal” Education Society‟s

cocsit is considered as marker, from the first cocsit the string begins and the last cocsit indicates the string is end there.

When we want to display special characters of HTML such as <, >, “,& on a web page.

We have to convert these character or represents these character as <, >, ", &

Because browser processes HTML special character as HTML tag & try to process those tag & display output on browser.

Example:
<?php
print "amit<ram";
?>
Output amit

When < is occurs, browser considers one tag is started and will process the tag In the above example <ram is consider as tag so it is not displayed

To get the required output we need to add the print statement as below

Example

```
<?php
print "amit&lt;ram";
?>
```

Output amit<ram.

We have to manually represent or converts those character for displaying on browser.
But performing this manually for every occurrence of character is difficult. For performing this PHP provides **htmlspecialchars** function.

PHP provides **htmlspecialchars** function to convert special characters of html into their equivalent characters.

The following characters are converted:

Less-than signs (<) are converted to <.
Greater-than signs (>) are converted to >.
Ampersands (&) are converted to &.
Double quotes (") are converted to "

```
<h2>Basic Structure of HTML Program</h2><hr>
<?php

$str1="<html>";

$str2="<head><title>PageTitle</title></head>";

$str3="<body>Body Text <body>";

$str4="<html>"; end;

$s1 = htmlspecialchars($str1);

$s2 = htmlspecialchars($str2);

$s3 = htmlspecialchars($str3);

$s4 = htmlspecialchars($str4);

print "<h3>$s1";print "<h3>$s2"; print "<h3>$s3"; print "<h3>$s4";
?>
```

Removing HTML tags

The **strip_tags( )** function removes HTML tags from a string:

$input = ' Courses are running at COCSIT are <UL> BCA <li> B.ScCS <UL>'; Print $input;
$output = strip_tags($input); Print $output;
Output
Courses are running at COCSIT are

- BCA
- B.Sc.CS

Courses are running at COCSIT are BCA B.ScCS.

3.2 Searching & Replacing String.

PHP has many functions to work with strings. The most commonly used functions for searching and modifying strings are those that use regular expressions to describe the string in question.

The functions described in this section do not use regular expressions they are faster than regular expressions, but they work only when you're looking for a fixed string.

Substrings

substr()

The function substr is used to extract or copy some character from strings
$p = substr(string, start [, length]);

The start argument is the position in string at which to begin copying, with 0 meaning the start of the string. The length argument is the number of characters to copy (the default is to copy until the end of the string).

Example

```
$name = "COCSIT Latur is greate ";
$part1 = substr($name, 8, 5);
$part2 = substr($name, 8);
Print “<br> $part2”;
```

Output

Latur
Latur is greate

substr_count()

To know or find how many times a smaller string occurs in a larger one, use substr_count( ) is used.

Syntax $number = substr_count(big_string, small_string);

Example:

```
$sketch = <<< End_of_Sketch
egg bacon and spam; egg bacon sausage and spam; spam bacon sausage spam;
End_of_Sketch;
$count = substr_count($sketch, "spam"); print("spam occurs $count times.");
```

Output:

spam occurs 4 times.

substr_replace( )

The substr_replace( ) function permits many kinds of string modifications:

$string = substr_replace(original, new, start [, length]);

The function replaces the part of original indicated by the start (0 means the start of the string) and length values with the string new. If no fourth argument is given, substr_replace( ) removes the text from start to the end of the string.

Example:

```
$greeting = "good morning citizen";
$farewell = substr_replace($greeting, "bye", 5, 7);
// $farewell is "good bye citizen"
```

Use a length of 0 to insert without deleting:

```
$farewell = substr_replace($farewell, "kind ", 9, 0);
// $farewell is "good bye kind citizen"
```

Use a replacement of "" to delete without inserting:

```
$farewell = substr_replace($farewell, "", 8);
// $farewell is "good bye"
```

can insert at the beginning of the string:

```
$farewell = substr_replace($farewell, "now it's time to say ", 0, 0);
```

str_replace()

This function replaces the characters with alternate character.
The str_replace() function replaces some characters with some other characters in a string.

Syntax:

str_replace(“old word”,”new word”,”String”);

Example:

```
<?php
echo str_replace("world","Peter","Hello world!");
?>
```

3.3 Formatting, joining and splitting String.

For formatting string PHP provides numbers of function as below.

strtolower() -function converts a string to lowercase.
strtoupper() - converts a string to uppercase.
lcfirst() - converts the first character of a string to lowercase.
ucfirst() - converts the first character of a string to uppercase.
ucwords() - converts the first character of each word in a string to uppercase.

Example:

```
<?php
echo strtolower("Hello WORLD.");
?>
```

Output Hello world.

Joining and Splitting a String.

For joining and splitting a string PHP provides join, implode and explode function

join() Function

Join function is used to join array elements with a string:

```
<?php
$arr = array('Hello','World!','Beautiful','Day!'); echo join(" ",$arr);
?>
```

Output:
Hello World! Beautiful Day!

The join () function returns a string from the elements of an array.
The join () function is an alias of the implode() function.
The separator parameter of join () is optional.

Syntax

join(separator,array)
Parameter
separator - Optional. Specifies what to put between the array elements.
Default -is "" (an empty string)
array -Required. The array to join to a string

Exploding and imploding

Exploding means separating a string word by word by using some character in the string in the array, also called splitting a string

Imploding means combing the array of string into a string also called joining a string

Data often arrives as strings, which must be broken down into an array of values.

Ex.

you might want to separate out the comma-separated fields from a string such as "COCSIT,Ambajogai Road, Latur,413512."

In these situations, use the explode( ) function:

$array = explode(separator, string [, limit]);

The first argument, separator, - is a string containing the field separator.

The second argument, -string, is the string to split.

The optional third argument, -limit, is the maximum number of values to return in the array.

```
$input = ' COCSIT,Ambajogai Road, Latur,413512.';
$fields = explode(',', $input);
// $fields is array('Fred', '25', 'Wilma')
```

The implode( ) function

The implode( ) function does the exact opposite of explode( )it creates a large string from an array of smaller strings:

$string = implode(separator, array);

The first argument, separator, is the string to put between the elements of the second argument, array.

To reconstruct the simple comma-separated value string,

```
$fields = array(„COCSIT“,“Ambajogai“, „Road“, „Latur“,“413512');
$string = implode(',', $fields); // $string is 'COCSIT,Ambajogai Road,
Latur,413512'
```

3.4 String Related Library functions.

Following are the some of the string related function in PHP

strcmp( )

To explicitly compare two strings as strings, casting numbers to strings if necessary, use the strcmp( ) function:

Syantax: $r = strcmp(string_1, string_2);

```
<?php
$s1 = “amol”;
$s2 = "sunil";
$r=strcmp($s1,$s2) if ($r==0)
{
echo(" $s1 & $s2 are equal <br>");
}
if ($r1>0) {
echo(" $s1 is greater than $s2 <br>");
}
if ($r1<0) {
echo(" $s1 is less than $s2 <br>");
}
?>
```

Output

amol is less than sunil

strcasecmp( )

This converts strings to lowercase before comparing them. Its arguments and return values are the same as those for strcmp( ):

$n = strcasecmp(“Darsh", "darSH");

```
<?php
$s1 = “amol”;
$s2 = "AMOL";
```

```
$r=strcasecmp($s1,$s2) if ($r==0)
{
echo(" $s1 & $s2 are equal <br>");
}
```

Output

amol & AMOL are equal.

strncmp( ) and strncasecmp( )

To compare only the first few characters of the string these function are used. The strncmp( ) and strncasecmp( ) functions take an additional argument, the initial number of characters to use for the comparisons:

```
$r = strncmp(string_1, string_2, len);
$r = strncasecmp(string_1, string_2, len);
```

Program for comparing first four characters of two string

```
<?php
$s1="Amol Mane";
$s2="Amol Kale";
$r=strncmp($s1,$s2,4); //compare first 4 char of $s1 & $s2 if($r==0)
{
print "<h2>First four char of two string are equal ";
}
else
{
print "<h2>First four char of two string are not equal ";
}
?>
```

Output

First four char of two string are equal

strrev( )

The strrev( ) function takes a string and returns a reversed copy of it:

Syantax : $string = strrev(string);

Example:
$s=strrev("There is no cabal"); print $s;

Output
labac on si erehT

str_repeat( )
The str_repeat( ) function takes a string and a count and returns a new string consisting of the argument string repeated count times:

$repeated = str_repeat(string, count);

to build a crude horizontal rule:

echo str_repeat('-', 40);

str_pad( )

The str_pad( ) function pads one string with another.
Optionally, you can say what string to pad with, and whether to pad on the left, right, or both:

$padded = str_pad(to_pad, length [, with [, pad_type]]);

The default is to pad on the right with spaces:

$string = str_pad('COCSIT', 30); print "$string";

Output
COCSIT

The optional third argument is the string to pad with:

$string = str_pad('COCSIT', 15, '. '); echo "$string 35";

Output
COCSIT. . . . . .

The optional fourth argument can be either STR_PAD_RIGHT (the default), STR_PAD_LEFT, or STR_PAD_BOTH (to center).

echo '[' . str_pad('COCSIT', 15, '. ', STR_PAD_LEFT) . "]
";
echo '[' . str_pad('COCSIT', 14, ' .', STR_PAD_BOTH) . "]
";

3.5 Use and advantage of regular expression over inbuilt function.

Regular Expressions:

If we want to search the particular type of string or word in a given string for that you can use regular expression.

A regular expression is a string that represents a pattern.

The regular expression functions compare that pattern to another string and see if any of the string matches the pattern.

PHP provides support for two different types of regular expressions:

POSIX and Perl-compatible. POSIX regular expressions are less powerful, and sometimes slower, than the Perl-compatible functions, but can be easier to read.

There are three uses for regular expressions:
Matching, which can also be used to extract information from a string;
Substituting new text for matching text; and
Splitting a string into an array of smaller pieces.

PHP provides following function for this:
ereg( )
preg_match( )

If you search for the regular expression "modi" in the string "modi is the pm of india" you get a match because "modi" occurs in that string.

Some characters have special meanings in regular expressions. For instance, a caret (^) at the beginning of a regular expression indicates that it must match the beginning of the string:

```
ereg('^modi, ' modi is the pm of india '); // returns true or 1
ereg('^modi, ' the pm of india is modi'); // returns false or nothing
```

Similarly, a dollar sign ($) at the end of a regular expression means that it must match the end of the string:

```
ereg('modi$, ' modi is the pm of india '); // returns false or nothing
ereg('modi$, ' the pm of india is modi'); // returns true or 1
```

You can define a range of characters with a hyphen (-). This simplifies character classes like "all letters" and "all digits":

```
ereg('[0-9]%', 'we are 25% complete');
// returns true
```

You can use the vertical pipe (|) character to specify alternatives in a regular expression:

```
ereg('cat|dog', 'the cat rubbed my legs');  // returns true
ereg('cat|dog', 'the dog rubbed my legs'); // returns true

ereg('cat|dog', 'the rabbit rubbed my legs'); // returns false
```

Repeating Sequences

To specify a repeating pattern, you use something called a quantifier. The quantifier goes after the pattern that's repeated and says how many times to repeat that pattern.

Regular expression quantifiers

Quantifier	**Meaning**
?	0 or 1
*	0 or more
+	1 or more
{n}	Exactly n times
{n,m}	At least n, no more than m times
{n,}	At least n times

Subpatterns

You can use parentheses to group bits of a regular expression together to be treated as a single unit called a sub pattern:

Example:

```
<?php
$mobile=ereg('[987][0-9]{9}','9845654607'); // returns true
$email=ereg('[a-zA-Z0-9]{3,}@[a-z]{3,}.[a-z]{2,}' ,gopal@rediff.com');//returns
true.

if($r1==1)
{
print "<h2>Valid Mobile Number";
else
{
print "<h2> InValid Mobile Number";
}
?>
```

3.6 Friend Function.

A friend function is a special function which is not a member function of a class but can access private and protected data of a class.

A friend function is used for accessing the non-public members of a class.

A class can allow non-member functions and other classes to access its own private data, by making them friends.

Thus, a friend function is an ordinary function or a member of another class.

When a data is declared as private inside a class, then it is not accessible from outside the class.

A function that is not a member or an external class will not be able to access the private data.

A programmer may have a situation where he or she would need to access private data from non-member functions and external classes.

For handling such cases, the concept of Friend functions is a useful tool.

Such functions can use all attributes of the class which names them as a friend, as if they were themselves members of that class.

For implementation of friend function in PHP you will need a base class.

A class that should be extended by the classes, that needs to have friends.

This base class provides the friend- architecture, so that it will need to be coded only once.

The End