**Experiment 10**

**Title :** Form validation using PHP regular expressions

**Aim :** To make student understand the concept of regular expressions and how to use them in form validation

**Form Validation Using Regular Expressions in PHP**

PHP regular expressions (regex) are a powerful tool for pattern matching and string manipulation. PHP supports two types of regular expressions: POSIX and Perl-compatible regular expressions (PCRE). Most modern PHP applications use PCRE due to its flexibility and powerful features.

Key Concepts of PHP Regular Expressions

Basic Syntax:

* Regular expressions are enclosed in delimiters, usually slashes /.
* Example: /pattern/.

Metacharacters:

* .: Matches any character except a newline.
* ^: Asserts the start of a line.
* $: Asserts the end of a line.
* \*: Matches 0 or more occurrences of the preceding element.
* +: Matches 1 or more occurrences of the preceding element.
* ?: Matches 0 or 1 occurrence of the preceding element.
* {n}: Matches exactly n occurrences of the preceding element.
* {n,}: Matches n or more occurrences of the preceding element.
* {n,m}: Matches between n and m occurrences of the preceding element.

Character Classes:

* [abc]: Matches any one of the characters a, b, or c.
* [^abc]: Matches any character except a, b, or c.
* [a-z]: Matches any lowercase letter from a to z.
* [A-Z]: Matches any uppercase letter from A to Z.
* [0-9]: Matches any digit.

Escaping Characters:

* Use a backslash \ to escape metacharacters if you want to match them literally.
* Example: \. matches a literal dot.

Quantifiers:

* \*, +, ?, {n}, {n,}, and {n,m} specify how many times the preceding element should be matched.

Anchors:

* ^ (caret) asserts the position at the start of a string.
* $ (dollar) asserts the position at the end of a string.

Grouping and Alternation:

* Parentheses () are used for grouping.
* The pipe | acts as a logical OR.
* Example: (cat|dog) matches either "cat" or "dog".

**PHP Functions for Regular Expressions**

1. preg\_match(): Checks if a pattern matches a string. Returns 1 if there is a match, 0 if not, and FALSE if an error occurred.

if (preg\_match("/pattern/", $subject)) {

// Match found

}

1. preg\_match\_all(): Finds all matches of a pattern in a string. Returns the number of matches found.

preg\_match\_all("/pattern/", $subject, $matches);

1. preg\_replace(): Replaces matched patterns with a specified replacement string.

$result = preg\_replace("/pattern/", "replacement", $subject);

1. preg\_split(): Splits a string by a regular expression.

$result = preg\_split("/pattern/", $subject);

1. preg\_grep(): Filters an array by a regex pattern.

$result = preg\_grep("/pattern/", $array);

**Conclusion:-**

Thus, we have studied, understood and practically checked regular expression concepts in PHP