**পিএইচপি date() ফাংশন :** পিএইচপি date() ফাংশন একটি টাইমস্ট্যাম্পকে পড়ার উপযোগী করে তারিখ এবং সময়ে সাজায়।

**date(format,timestamp)**

**প্যারামিটার এর বিবরণ**

Ø format : আবশ্যক। টাইমস্ট্যাম্প এর ফরম্যাটকে নির্দেশ করে।

Ø timestamp : ঐচ্ছিক। একটি টাইমস্ট্যাম্পকে নির্দেশ করে। ডিফল্ট হলো বর্তমান তারিখ ও সময়।

**সাধারণ তারিখ পাওয়ার উপায়**

আপনি কিভাবে তারিখের ফরম্যাট করতে চাচ্ছেন তাdate() ফাংশনের আবশ্যক প্যারামিটার( format) এর মাধ্যমে নির্ধারণ করে দিতে হয়। এখানে কিছু ক্যারেক্টার দেওয়া হলো যেগুলো সাধারণত তারিখের জন্য ব্যবহার করা হয়ঃ

Ø d - মাসের মধ্যে একটি দিনকে রিপ্রেজেন্ট করে (১ থেকে ৩১)।

Ø m - মাসকে রিপ্রেজেন্ট করে(১ থেকে ১২)।

Ø Y - একটি বৎসরকে রিপ্রেজেন্ট করে (চার ডিজিটে)।

Ø l(ছোট অক্ষরের'L') - সপ্তাহের একটি দিনকে রিপ্রেজেন্ট করে।

**<?php echo "আজকের তারিখ হলো " . date("d/m/Y"). "<br>"; ?>**

নিজের ইচ্ছামতো তারিখ এবং সময়কে ফরম্যাট করার জন্য স্পেশাল ক্যারেক্টার যেমনঃ "/", ".", অথবা "-" ব্যবহার করা যায়।

**সাধারণ সময় পাওয়ার উপায়**

এখানে কিছু ক্যারেক্টার দেওয়া হলো যেগুলো সাধারণত সময়ের জন্য ব্যবহার করা হয়ঃ

Ø h - 12-hour ফরম্যাটের ঘন্টাকে নির্দেশ করে যা শুন্য(০) থেকে শুরু হয়(00 থেকে12)

Ø i - মিনিটকে নির্দেশ করে যা শুন্য(০) থেকে শুরু হয়(00 থেকে59)

Ø s - সেকেন্ডকে নির্দেশ করে যা শুন্য(০) থেকে শুরু হয়(00 থেকে59)

Ø a - ছোট অক্ষরেরAnte meridiem(am) এবং Post meridiem(pm)

**<?php echo "বর্তমান সময় হলো " . date("h:i:sa"); ?>**

Ø স্বয়ংক্রিয়ভাবে সাইটের কপিরাইট সময় সেট করা: &copy;2023-<?php echo **date("Y");**?>

**পিএইচপি mktime() ফাংশনের মাধ্যমে একটি তারিখ তৈরি**

**mktime(hour,minute,second,month,day,year)**

<?php

//বিভিন্ন প্যারামিটার ব্যবহার করে তারিখ এবং সময় তৈরি

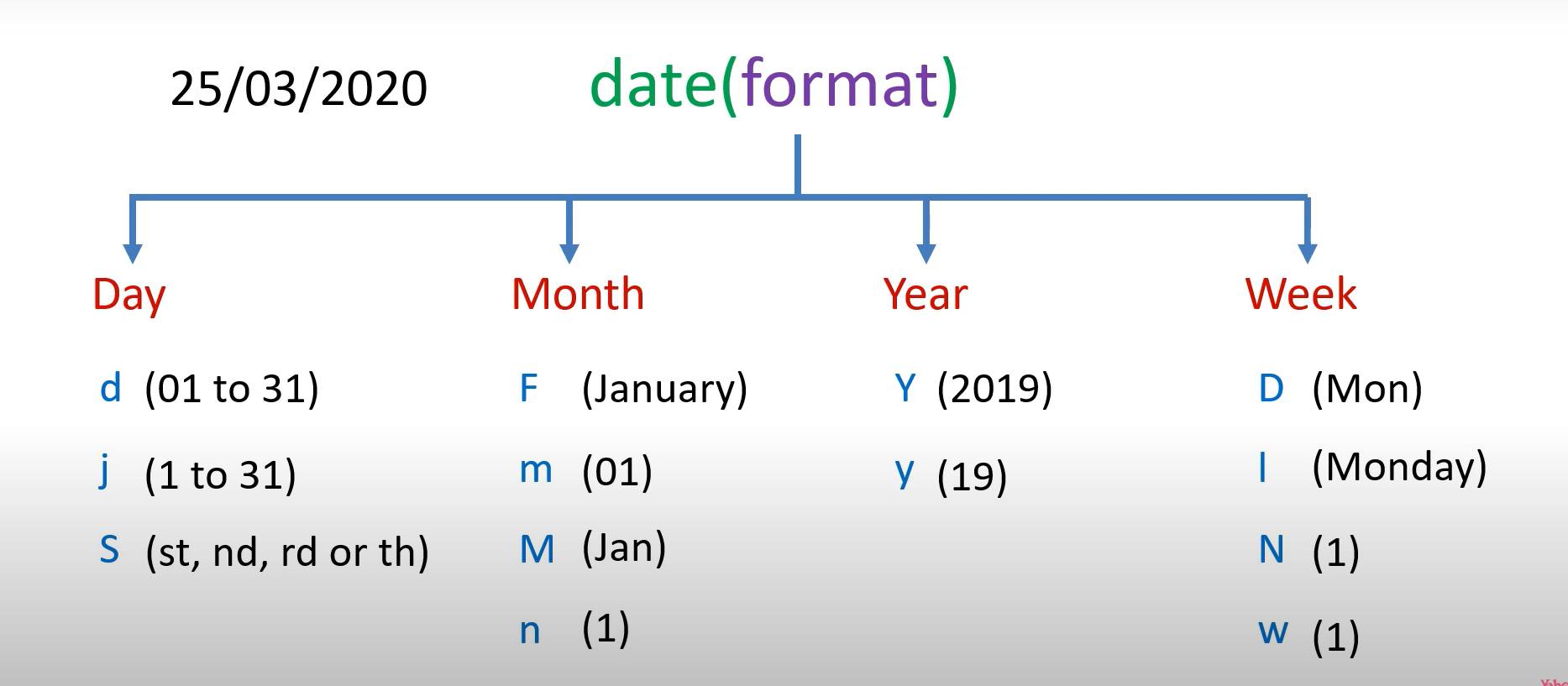
$dayay = **mktime(11, 14, 54, 8, 12, 2014);**

echo "তৈরিকৃত তারিখটি হলো". date("Y-m-d h:i:sa", $dayay);

?>

**ফলাফল**

তৈরিকৃত তারিখটি হলো2014-08-12 11:14:54am



[#List](https://www.facebook.com/hashtag/list?__eep__=6&__cft__%5b0%5d=AZUEkA6jINflAenVqNRxHeHEAQD1WZLLLaOwLcZii5TJv_OC4xEJjv0ONixNA7WZOi0UZ6Ra8uvs5JlZ0A3Ce0Sop91ceRnvvaNNifguremnhAGwq7DiWj7i3kcWUNUJSiaOPn0cLwvev_M0p1rSY-e0&__tn__=*NK-R) of common CSS text-related attributes along with their descriptions:

---------------------------------------------------------------------

1. **color**: Sets the color of the text.
2. **font-family**: Specifies the font family for the text.
3. **font-size**: Defines the size of the font.
4. **font-weight**: Sets the thickness or boldness of the text.
5. **font-style**: Specifies the style of the font (italic, oblique, normal).
6. **text-align**: Determines the horizontal alignment of the text (left, right, center, justify).
7. **text-decoration**: Adds decoration to the text (underline, overline, line-through, none).
8. **text-transform**: Controls the capitalization of the text (uppercase, lowercase, capitalize).
9. **line-height**: Sets the height of a line of text.
10. **letter-spacing**: Adjusts the spacing between characters.
11. **word-spacing**: Adjusts the spacing between words.
12. **text-indent**: Specifies the indentation of the first line of text within an element.
13. **white-space**: Defines how whitespace characters are handled (normal, nowrap, pre, pre-line, pre-wrap).
14. **word-wrap** / **overflow-wrap**: Controls whether long words can break and wrap onto the next line.
15. **hyphens**: Controls how hyphenation is handled in text.
16. **text-overflow**: Determines how overflowing text is displayed (ellipsis, clip).
17. **direction**: Specifies the direction of the text (ltr, rtl).
18. **unicode-bidi**: Controls the direction of the text's inline base direction.
19. **vertical-align**: Adjusts the vertical alignment of inline elements.
20. **tab-size**: Sets the size of tab characters.
21. **text-shadow**: Adds a shadow to text characters.
22. **writing-mode**: Defines the writing mode of the text (horizontal-tb, vertical-rl, vertical-lr).
23. **text-combine-upright**: Combines vertical text within a horizontal line.
24. **text-align-last**: Specifies the alignment of the last line of a block or a line before a forced line break.

Remember that these attributes can be combined to style text in various ways. CSS is a powerful tool for controlling the appearance of text on web pages, and understanding these attributes will help you create visually appealing and readable content.

Thank You.

[#List](https://www.facebook.com/hashtag/list?__eep__=6&__cft__%5b0%5d=AZVdaG78w9XjzxjJ-3euanyThTQrBJoD-C37cG6IKbGC1lC2TnSxHAokbI6YgqARP889I2qvE64vLBf2XgoGxmw2mmXdp3olOy-xi0ZzoD9sMGZ8QqpkOn46jsIQSzsgqTT_2I800FffSPOcSp3qPiKl&__tn__=*NK-R) of common HTML tags that are used for working with text content on web pages:

----------------------------------------------------------------------

1. **<h1>** to **<h6>**: Headings, ranging from largest (h1) to smallest (h6).
2. **<p>**: Paragraph tag for containing text paragraphs.
3. **<span>**: Inline container for a small amount of text or a single inline element.
4. **<strong>** / **<b>**: Represents strong importance, typically displayed as bold text.
5. **<em>** / **<i>**: Represents emphasized text, typically displayed as italicized text.
6. **<u>**: Underlines the enclosed text.
7. **<s>** / **<strike>** / **<del>**: Represents deleted or strikethrough text.
8. **<sub>**: Renders text as subscript.
9. **<sup>**: Renders text as superscript.
10. **<blockquote>**: Indicates a block of text quoted from another source.
11. **<q>**: Indicates a short inline quotation.
12. **<abbr>**: Represents an abbreviation or acronym, with an optional full explanation.
13. **<cite>**: Specifies a citation or reference to a creative work.
14. **<code>**: Displays code or computer code inline.
15. **<pre>**: Preformatted text block, maintaining spacing and line breaks.
16. **<kbd>**: Represents user input, typically displayed as monospace text.
17. **<var>**: Represents a variable or placeholder in text.
18. **<mark>**: Highlights text to indicate relevance or search results.
19. **<ins>**: Represents inserted or underlined text.
20. **<ruby>**: Represents small annotations (like furigana) for East Asian typography.
21. **<rt>**: Specifies the pronunciation of characters in a ruby annotation.
22. **<bdo>**: Overrides the text directionality for its content.
23. **<br>**: Inserts a line break (useful for single-line breaks in HTML).
24. **<wbr>**: Suggests an optional line break opportunity.
25. **<a>**: Anchor tag for creating hyperlinks.
26. **<ul>** / **<ol>** / **<li>**: Unordered list, ordered list, and list item tags.
27. **<dl>** / **<dt>** / **<dd>**: Definition list, term in a definition list, and definition in a definition list.
28. **<address>**: Contains contact information for the author or owner of a document.
29. **<time>**: Represents a specific period in time or a range of time.
30. **<noscript>**: Provides alternative content for users with disabled or unsupported scripts.
31. **<small>**: Represents smaller text, often used for disclaimers or legal text.
32. **<big>**: Represents larger text (deprecated in HTML5).
33. **<strong>** / **<b>**: Represents strong importance, typically displayed as bold text.

These tags allow you to structure and style your text content in various ways, making your web pages more readable and informative.

Use of echo , print\_r and print in PHP

-----------------------------------------

In PHP, both **echo** and **print\_r** are used to output information, but they serve slightly different purposes and have different behaviors:

**A). echo**:**echo** is a language construct in PHP used to output text and variables.  
👍 It can output multiple values separated by commas.  
👍 It does not have a return value; it directly outputs the content to the browser or standard output.  
👍 It is commonly used for simple output purposes, like printing strings and variables.  
  
Example:  
phpCopy code$name = "Abdur Rahim";  
echo "Hello, " . $name;

// Output: Hello, Abdur Rahim

**B). print\_r**:**print\_r** is a function in PHP used for printing human-readable information about a variable, especially arrays and objects.  
👍 It's often used for debugging and inspecting complex data structures.  
👍 It can handle arrays and objects, showing their structure and content.  
👍 It returns **true** if it successfully prints the variable, or **false** if an error occurs.  
  
Example:

$city\_name = array("Dhaka", "Rajshahi", "Khulna");  
print\_r($city\_name);  
/\* Output:  
Array  
(  
[0] => Dhaka  
[1] => Rajshahi  
[2] => Khulna  
)  
\*/

In PHP, the **print** statement is used to output text and variables to the browser or standard output. It is quite similar to the **echo** statement but with a few differences:

1. **Outputting Text and Variables**: Like **echo**, **print** can be used to display both plain text and variables. You can concatenate strings and variables together to create the desired output.  
   phpCopy code$name = "Abdur Rahim";  
   print "Hello, " . $name; // Output: Hello, Abdur Rahim
2. **Return Value**: Unlike **echo**, which doesn't have a return value and directly outputs content, **print** returns a value of **1**. This means that it can be used within expressions or assignments.  
   phpCopy code$result = print "Hello, World"; // Output: Hello, World  
   echo $result; // Output: 1
3. **Only One Argument**: The **print** statement can only take a single argument, while **echo** can take multiple arguments separated by commas.  
   phpCopy codeprint "Hello", "World"; // Invalid  
   echo "Hello", "World"; // Valid

In practice, **echo** is more commonly used than **print** due to its slightly better performance and greater flexibility with multiple arguments. However, both **echo** and **print** are used for the same purpose – to display content on the screen – and you can choose the one that fits your coding style and requirements.

**সিএসএস ব্যাকগ্রাউন্ড প্রোপার্টিসমূহ**

Ø **background :** একটি ডিক্লেয়ারেশনের মধ্যে সকল ব্যাকগ্রাউন্ড প্রোপার্টি সেট করার জন্য এই প্রোপার্টিটি ব্যবহার করা হয়।

Ø **background-color** : একটি এলিমেন্টের জন্য ব্যাকগ্রাউন্ড কালার সেট করার জন্য এই প্রোপার্টিটি ব্যবহার করা হয়।

Ø **background-image** : একটি এলিমেন্টের জন্য ব্যাকগ্রাউন্ড ইমেজ সেট করার জন্য এই প্রোপার্টিটি ব্যবহার করা হয়।

Ø **background-position** : ব্যাকগ্রাউন্ড ইমেজের পজিশন সেট করার জন্য এই প্রোপার্টিটি ব্যবহার করা হয়।

Ø **background-repeat** : ব্যাকগ্রাউন্ড ইমেজ কিভাবে রিপিট হবে, তা সেট করার জন্য এই প্রোপার্টিটি ব্যবহার করা হয়।

Ø **background-attachment** : পেজের ব্যাকগ্রাউন্ড ইমেজ ফিক্সড থাকবে নাকি স্ক্রল হবে তা সেট করার জন্য এই প্রোপার্টিটি ব্যবহার করা হয়।

**{**

Background: green;

background-color: green;

background-image: url("paper.jpg");

background-position: right top;

background-repeat: repeat-x; & repeat-y; / no-repeat;

background-attachment: fixed;

**}**

**OR**

{ background: [#ffffff](https://www.facebook.com/hashtag/ffffff?__eep__=6&__cft__%5b0%5d=AZUs_1tSrLahA31g2yhZB1zwUN29lKoQ7yt2rh-5XKRH4qmkmmrF1f3IpqjgR9DaUWiqPgTMWOxoLql3asS2qq92VDX2lPCnSMKTTEFibLT8uyPWxkjVBnrwFRJW9OHhBYMnKuzxnglGSPM9M5JgjCUA&__tn__=*NK-R) url("paper2.jpg") no-repeat right top fixed;}

**PHP Function**

ফাংশন হচ্ছে একগুচ্ছ স্টেটমেন্ট যাকে আমরা প্রোগ্রামের মধ্যে বার বার ব্যবহার করতে পারি। ফাংশন নিজে নিজে সম্পাদিত(execute) হয় না। ফাংশনকে সম্পাদন(execute) করতে হলে অবশ্যই কল(call) করতে হবে।

একটা ফাংশনকে যখন কল করা হয় তখনই এটা execute হয় আর পেজের যেকোনো জায়গা থেকে একটা ফাংশনকে কল করা যায়। এটা পিএইচপি এর মূল শক্তি বলতে পারেন। পিএইচপি তে প্রায় ১০০০ এরও বেশি বিল্ট ইন ফাংশন আছে । পিএইচপির নিজস্ব ফাংশন থাকা সত্ত্বেও আমরা আমাদের প্রয়োজন অনুযায়ী ফাংশন তৈরি করতে পারি।

পিএইচপি ফাংশন তৈরি করতে প্রথমে "function" কী-ওয়ার্ড লিখতে হয় এর পরে ফাংশন এর নাম এর পরে () এবং {}

**function**functionName() {

code to be executed;

}

**টিপস:** ফাংশনের নাম দেয়ার সময় এমন নাম দিন যেটা দেখেই যেন বোঝা যায় ফাংশনটি কি করবে। ফাংশনের নাম অক্ষর বা\_ দিয়ে শুরু হতে পারে, নাম্বার দিয়ে শুরু হবেনা।

<?php

function writeName(){

echo "Abdur Rahim";

}

echo "My name is ";

writeName();

?>

**পিএইচপি ফাংশন প্যারামিটার**

প্যারামিটারের মাধ্যমে ফাংশনে তথ্য সরবরাহ করা হয়। পিএইচপিতে প্যারামিটার ভ্যারিয়েবল এর মতই। ফাংশনের নামের পরে প্রথম বন্ধনী()-র মধ্যে প্যারামিটার রাখা হয়। আপনার দরকার অনুযায়ী যত খুশি প্যারামিটার ব্যবহার করতে পারবেন, শুধু কমা(,) চিহ্নের মাধ্যমে প্যারামিটার গুলোকে আলাদা করতে হবে।

নিচের উদাহরণে ফাংশনের মধ্যে 2টি প্যারামিটার **$country\_name** , **$capital\_city**ব্যবহার করা হয়েছে। countryName ফাংশনকে কল করার সময় এর মধ্যে প্যারামিটারের বিভিন্ন ভ্যালু ব্যবহার করা হয়েছে|

<?php

function countryName**($country\_name, $capital\_city**) {

echo "Contry Name- **$country\_name**and Capital\_city name is **$capital\_city**<br>";

}

countryName("Bangladesh", "Dhaka");

countryName("India","[New Delh](https://l.facebook.com/l.php?u=https%3A%2F%2Fwww.google.com%2Fsearch%3Fsca_esv%3D565227327%26sxsrf%3DAM9HkKnaowzRa-bQDqF_1-pArfV-RNqvTw%253A1694661889629%26q%3DNew%2520Delhi%26stick%3DH4sIAAAAAAAAAONgVuLQz9U3SMkpM3jEaMwt8PLHPWEprUlrTl5jVOHiCs7IL3fNK8ksqRQS42KDsnikuLjgmngWsXL6pZYruKTmZGQCAFcKT4ROAAAA%26sa%3DX%26ved%3D2ahUKEwi0_ce_k6mBAxXHUGwGHVRuAbUQzIcDKAB6BAgbEAE%26fbclid%3DIwAR0Na2itSkRmHdAcqkvxto7eHyqkkbg7kT15TJnunW122c6kOoAtYFAIH8o&h=AT0Eo0ZonkIPq5vR-IoFQyFabsi6UQ8vk0vbIxUAivw4IQByJJo0Sgeu_2IM6uckahQZhoR_1jHRQTaqMP3zQFQSGoyECtyX2J7lKP4QSdbODLemzJHsC2shmONY5TsBdTYr&__tn__=-UK-R&c%5b0%5d=AT1N2QLv73Z6HKDZupQmI9rEXkP5lDFuJGgeBJsH8raAQ_zB6L9Nm7sQhHScEiwD8y7syYicHxkJt9sYwHX3M6D9Z_Ht6GBMA6X31HVSyrW7lnN9YLLG62jzpmAvvNVVC64F1tXKCJgHvw6heAU_UCt7XQY1BH5ZnaIlRBY)i");

?>

Outpot: Contry Name- Bangladesh, and Capital\_city name is Dhaka Contry Name- India, and Capital\_city name is [New Delhi](https://l.facebook.com/l.php?u=https%3A%2F%2Fwww.google.com%2Fsearch%3Fsca_esv%3D565227327%26sxsrf%3DAM9HkKnaowzRa-bQDqF_1-pArfV-RNqvTw%253A1694661889629%26q%3DNew%2520Delhi%26stick%3DH4sIAAAAAAAAAONgVuLQz9U3SMkpM3jEaMwt8PLHPWEprUlrTl5jVOHiCs7IL3fNK8ksqRQS42KDsnikuLjgmngWsXL6pZYruKTmZGQCAFcKT4ROAAAA%26sa%3DX%26ved%3D2ahUKEwi0_ce_k6mBAxXHUGwGHVRuAbUQzIcDKAB6BAgbEAE%26fbclid%3DIwAR0HET9gvPZUvrPSKd00tdC8NmDjs8VsSdtW4efa4qJz9fb2FeOzEfyUi4c&h=AT0Eo0ZonkIPq5vR-IoFQyFabsi6UQ8vk0vbIxUAivw4IQByJJo0Sgeu_2IM6uckahQZhoR_1jHRQTaqMP3zQFQSGoyECtyX2J7lKP4QSdbODLemzJHsC2shmONY5TsBdTYr&__tn__=-UK-R&c%5b0%5d=AT1N2QLv73Z6HKDZupQmI9rEXkP5lDFuJGgeBJsH8raAQ_zB6L9Nm7sQhHScEiwD8y7syYicHxkJt9sYwHX3M6D9Z_Ht6GBMA6X31HVSyrW7lnN9YLLG62jzpmAvvNVVC64F1tXKCJgHvw6heAU_UCt7XQY1BH5ZnaIlRBY)

**পিএইচপি রিটার্ন ভ্যালু**

ফাংশন থেকে ভ্যালু রিটার্ন করার জন্য **return** স্টেটমেন্ট ব্যবহার করা হয়ঃ

<?php

function calculate**($numberOne**, **$numberTwo**) {

**$total** = $numberOne + $numberTwo;

return **$total**;

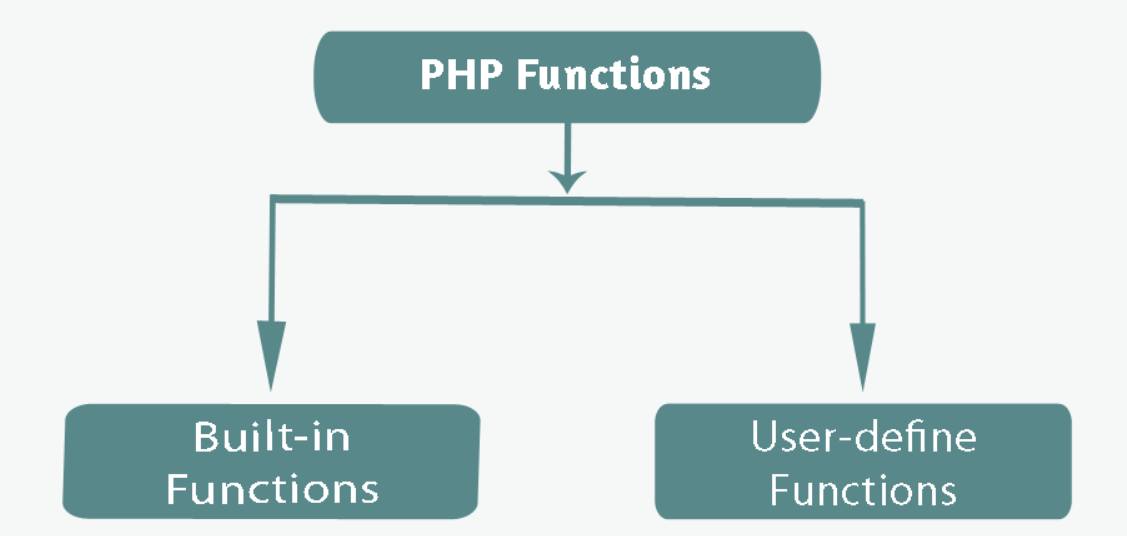
}

echo "11 + 9 = " **. calculate(11, 9) .** "<br>";

echo "8 + 7 = " **. calculate(8, 7) .** "<br>";

echo "3 + 2 = **" . calculate(3, 2)**;

?>



**PHP Loop**

লুপ দিয়ে একটা কোডব্লক যতবার ইচ্ছা এক্সিকিউট করানো যায়। পূনরাবৃত্তির কাজগুলি লুপ দিয়ে করা হয়। লুপে শর্ত দেয়া যায় এবং যতক্ষন শর্ত না মিলবে ততক্ষন পূনরাবৃত্তি চলতেই থাকবে।

পিএইচপিতে ৪ (চার) ধরনের লুপ আছে যেমন

**হোয়াইল লুপ (while loop):** হোয়াইল লুপে প্রথমেই একটা শর্ত/এক্সপ্রেশন থাকে, যতক্ষন শর্তটি পুরন না হয় ততক্ষন হোয়াইল লুপ(while loop) টি একটা কোড ব্লক কেexecute করতে থাকে। যখন শর্ত মিলে যায় তখন লুপ থেমে যায়। যদিcondition false হয় তখন সেকেন্ড ব্রাকেটের ভিতর (কোডব্লকটি) কোডটি এড়িয়ে যায়(ব্রাকেট না বলেcurly braces বলা উচিৎ)। যখন শেষের ব্রাকেটটি আসবে তখনcondition আবার চেক করবে, true হলে কোড আবারexecute করবে।condition এ যতবার দেয়া আছে ততবার এভাবে চলবেই।

<?php

$i=1;

while($i<6){

echo 'A counter start from '. $i . '<br/>';

$i++;

}

?>

**আউটপুট**

A counter start from 1

A counter start from 2

A counter start from 3

A counter start from 4

A counter start from 5

**ব্যাখ্যা:**প্রথমে $i এর মান ১ দিয়ে শুরু হয়েছে। এরপরwhile লুপের ভিতর শর্তটি($i<6) চেক করবে $i এর মান কত?

১ম বার $i এর মান ১ ,তার মানে ১ এখানে ৬ এর চেয়ে ছোট তাই এখন কোডব্লকে ঢুকবে এবং সেখানকার echo স্টেটমেন্ট বা যে কোডই থাকুন একবার এক্সিকিউট করবে। এবং$i++ এর কারনে$i এর মান ১ বাড়াবে।

২য় বার, তাহলে এবার$i এর মান পাবে ২ এবং এটাও ৬ এর চেয়ে ছোট ফলে শর্তটি($i<6) true রিটার্ন করবে এবং লুপটি আবার একবার কোডব্লকটি এক্সিকিউট করবে। যেহেতু লুপের ভিতর$i++ আছে তাই আবার$i এর ১ বাড়াবে সুতরাং$i এর মান এই লুপ শেষে হল ৩। কেননা প্রথমবার ১ বাড়িয়ে ২ হয়েছিল।

এভাবে ৫ বার লুপটি পূনরাবৃত্তি হবে। যখনি$i এর মান বাড়তে বাড়তে ৬ হয়ে যাবে তখন শর্তটি($i<6) আর মিলবেনা কেননা ৬ তো ৬ এর ছোট নয় বরং সমান সুতরাং শর্তটিfalse রিটার্ন করবে।

এবার প্রথমে $i = 6 দিলে কি আউটপুট আসবে? কিছুই আসবেনা কারন তখন($i<6) এই শর্তের এর সাথে মিলবেনা তাই কোডexecute হবেনা। অর্থাৎ এখন$i (বা ৬) ৬ এর চেয়ে ছোট নয়।

Ø **ডু হোয়াইল লুপ (do...while loop) :**এটাwhile লুপ এর মতই শুধু পার্থক্য এটুকু যে শর্তটি টি শেষে যাচাই হয়, শুরুতে করার পরিবর্তে। অর্থাৎ কমপক্ষে একবার লুপ টি ঘুরবেই।

<?php

$i=1;

do{

echo 'A counter start from '. $i . '<br/>';

$i++;

}

while ($i<6); ?>

**আউটপুট**

A counter start from 1

A counter start from 2

A counter start from 3

A counter start from 4

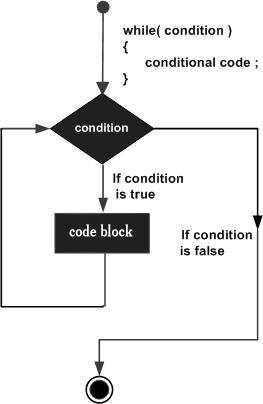
A counter start from 5

কিন্তু এখানে যদি$i=6 দেই তাহলে কি হবে?

নিচের মত আউটপুট আসবে কারন প্রথমবার শর্ত যাচাই ছাড়াই লুপটি ঘুরবে।

**A counter start from 6**

দেখুন লুপ টি প্রথমবার চেকিং ছাড়াই একবার ঘুরেছে। প্রথমবার ঘোরার পর চেক করেছে যে লুপটিrepeat হবে কিনা? কিন্তুwhile লুপে এটা প্রথমবারই এই চেক হয়।



**php তে fgets(STDIN) এবং readline() এর পরিচয়?**

**--------------------------------------------------------------**

PHP এ **fgets(STDIN)** এবং **readline()** দুটি ভিন্ন মেথড ব্যবহার করে কমান্ড লাইন থেকে ইনপুট পড়া যায়। এই দুটি মেথড একই উদ্দেশ্যে কাজ করে, তবে তাদের কিছু পার্থক্য রয়েছে:

**fgets(STDIN):fgets**এর মাধ্যমে ডেটা পড়া হয় একটি ফাইল থেকে বা, স্ট্যান্ডার্ড ইনপুট থেকে (সাধারণভাবে কমান্ড লাইন থেকে)। আর**STDIN** হলো PHP-র পূর্বনির্ধারিত ফাইল হ্যান্ডলার, যা সাধারণভাবে কমান্ড লাইন থেকে ইনপুট নেয়।  
**fgets(STDIN)** স্ট্যান্ডার্ড ইনপুট থেকে একটি লাইন টেক্সট পড়ে এবং সেটি স্ট্রিং হিসেবে প্রদান করে।  
আপনি **fgets** ব্যবহার করার সময় স্ট্যান্ডার্ড ইনপুট (**STDIN**) স্পষ্টভাবে সূচনা করতে হবে।

একটি উদাহরণ, **fgets(STDIN)** ব্যবহার করে কমান্ড লাইন থেকে ইনপুট পড়ার:

1. <?php
2. echo "আপনার নাম লিখুন: ";
3. $name = fgets(STDIN);
4. echo "হ্যালো, " . $name;
5. ?>

আর, **readline()**:**readline()** হলো PHP দ্বারা সরলীকরণ করা ফাংশন, যা কমান্ড লাইন থেকে ইনপুট নেওয়া সহজ করে।  
এটি স্ট্যান্ডার্ড ইনপুট থেকে একটি লাইন টেক্সট পড়ে এবং সেটি স্ট্রিং হিসেবে প্রদান করে।  
**readline()** সাধারণভাবে ইন্টারেকটিভ কমান্ড-লাইন স্ক্রিপ্টে ব্যবহৃত হয়, কারণ এটি ইনপুটের যাচাইকরণ করে এবং ইনপুটের পূর্বে দেওয়া কমান্ডের ইতিহাস সংরক্ষণ করে রাখে।

একটি উদাহরণ, **readline()** ব্যবহার করে কমান্ড লাইন থেকে ইনপুট পড়ার:

1. <?php
2. $name = readline("আপনার নাম লিখুন: ");
3. echo "হ্যালো, " . $name;
4. ?>

In summary, both **fgets(STDIN)** and **readline()** are inbuilt function in PHP that is used to read user input from the command line window, but **readline()** is often more convenient for interactive scripts because of its built-in features. **fgets(STDIN)** is more basic **(কোডমাম-তে ব্যবহার হবে )**and requires you to handle input validation and history management manually.

Thank You.

