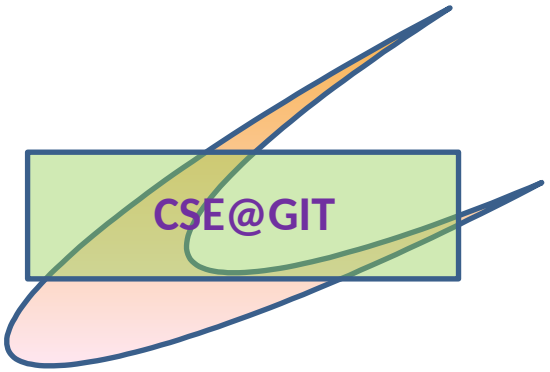
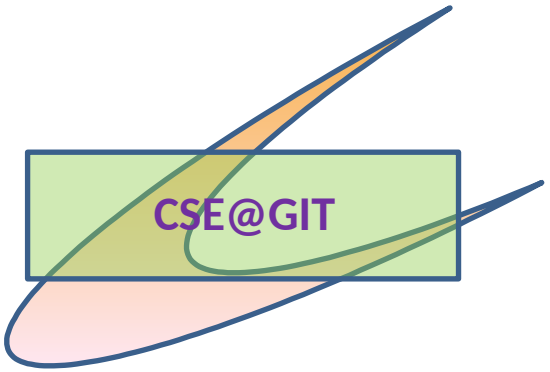
Experiment No. 9

Problem Definition: 9. Write a PHP program to store current date-time in a COOKIE and display the 'Last visited on' date-time on the web page upon reopening of the same page.



Experiment No. 10

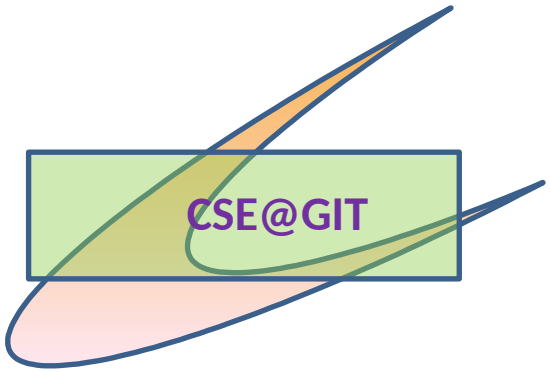
Problem Definition: 10. Write a PHP program to store page views count in SESSION, to increment the count on each refresh, and to show the count on web page.



Objectives of the Experiment:

- To demonstrate the use PHP
- Use PHP

- How To Guide in http://localhost
- Compile and run PHP code




[Author : William Stadtwald Demchick]

Introduction to PHP

[Author : Colin Viebrock]

Introduction to PHP

- Developed by Rasmus Lerdorf in 1994
- PHP is a server-side scripting language, embedded in XHTML pages
- PHP has good support for form processing
- PHP can interface with a wide variety of databases
- Home page : http://php.net/

Introduction to PHP

- Developed by Rasmus Lerdorf in 1994
- PHP is a server-side scripting language, embedded in XHTML pages
- PHP has good support for form processing
- PHP can interface with a wide variety of databases
- Home page : http://php.net/

- Contributions by Rasmus Lerdorf :
  Github https://github.com/rlerdorf
  Home page : https://lerdorf.com/

Introduction to PHP

- Developed by Rasmus Lerdorf in 1994
- PHP is a server-side scripting language, embedded in XHTML pages
- PHP has good support for form processing
- PHP can interface with a wide variety of databases
- Home page : http://php.net/

- Contributions by Rasmus Lerdorf :
  Github https://github.com/rlerdorf
  Home page : https://lerdorf.com/

- Contributing to PHP : http://php.net/get-involved.php

Getting Started

- What is PHP?
- What can PHP do?

[http://php.net/manual]

Getting Started

What is PHP?

- PHP : recursive acronym for PHP: Hypertext Preprocessor
- Widely-used open source general-purpose scripting language
- Suited for web development
- Can be embedded into HTML

[http://php.net/manual/en/getting-started.php]

Getting Started

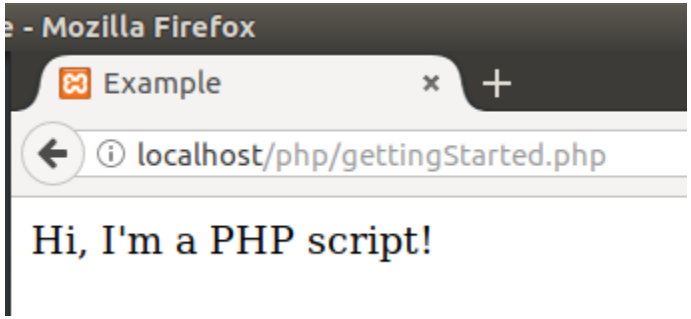
```
<!DOCTYPE HTML>
<html>
    <head>
        <title>Example</title>
    </head>
    <body>




    </body>
</html>
```

Getting Started

```
<!DOCTYPE HTML>
<html>
    <head>
        <title>Example</title>
    </head>
    <body>

        <?php
            echo "Hi, I'm a PHP script!";
        ?>

    </body>
</html>
```

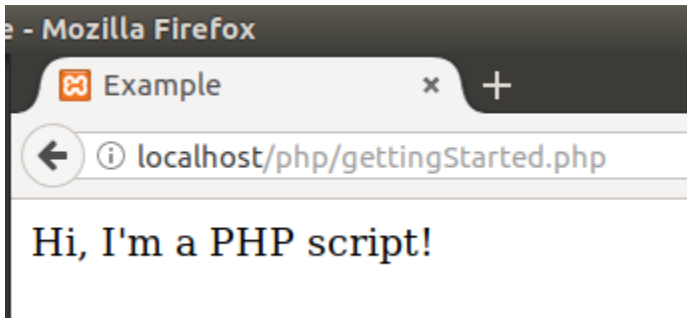
Getting Started

```
<!DOCTYPE HTML>
<html>
    <head>
        <title>Example</title>
    </head>
    <body>

        <?php
            echo "Hi, I'm a PHP script!";
        ?>

    </body>
</html>
```



Getting Started

```
<!DOCTYPE HTML>
<html>
    <head>
        <title>Example</title>
    </head>
    <body>

        <?php
            echo "Hi, I'm a PHP script!";
        ?>

    </body>
</html>
```



view-source:http://localhost/php/gettingStarted.php

```
<!DOCTYPE HTML>
<html>
    <head>
        <title>Example</title>
    </head>
    <body>

        Hi, I'm a PHP script!
    </body>
</html>
```

Getting Started

- Instead of lots of commands to output HTML as in PERL PHP pages contain HTML with embedded code that does "something"

```
<!DOCTYPE HTML>
<html>
    <head>
        <title>Example</title>
    </head>
    <body>
        <?php
            echo "Hi, I'm a PHP script!";
        ?>
    </body>
</html>
```

Getting Started

- Instead of lots of commands to output HTML as in PERL PHP pages contain HTML with embedded code that does "something"
- "something" : in this case, output "Hi, I'm a PHP script!"

```
<!DOCTYPE HTML>
<html>
    <head>
        <title>Example</title>
    </head>
    <body>
        <?php
            echo "Hi, I'm a PHP script!";
        ?>
    </body>
</html>
```

What is PHP?

- PHP code is enclosed in special start and end processing instructions
  start **<?php** and
  end **?>**

```
<?php

?>
```

- This allow you to jump into and out of "PHP mode"
- **Everything outside** of a pair of opening and closing tags is **ignored by the PHP parser**
- Allows PHP files to have mixed content, be embedded in HTML documents

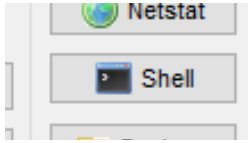
What is PHP?

- PHP code is **executed on** the server
- Generating HTML then sent to the client
- Client would receive the results of running that script, but would not know what the underlying code was

Compile .php

- View / Generate output of .php file
  (Like we check for PERL program **syntax correctness**)

- To run a PHP program
- XAMP installation , Windows : PHP available in
  C:\xampp\php**php.exe** hello.php
  (Or in UNIX like systems , if PHP is installed , directly :)
  php hello.php
  or
- Click Shell in XAMPP Control Panel
  Navigate (change directory , **cd**) to folder where hello.php is saved , then
  php hello.php



Compile .php

- What is present in output after compilation with php parser?

  Entire html code + (embedded with) output of php code

  or

  Only output of php code?

Compile .php

- What is present in output after compilation with **PHP** parser?
  Entire html code + output of php code
  or
  Only output of php code?

- Compile first to check for error
  then
  Access the file through localhost or 127.0.0.1

Compile .php

- What is present in output after compilation with **PHP** parser?
  Entire html code + output of php code
  or
  Only output of php code?

- Compile first to check for error
  then
  Access the file through localhost or 127.0.0.1

- http://localhost/hello.php and file://path/hello.php are different

Compile .php

- What is present in output after compilation with **PHP** parser?
  Entire html code + output of php code
  or
  Only output of php code?
- Compile first to check for error
  then
  Access the file through localhost or 127.0.0.1
- http://localhost/hello.php and file:///filePath/hello.php **are different**
- Just double clicking on the file
  or
  Open with browser will not pass through a PHP parser
  Then .php file will rendered as normal html and **not program**

What can PHP do?

- PHP is mainly focused on server-side scripting
- PHP can do anything any other CGI program can do
- Collect form data
- Generate dynamic page content
- Send and receive cookies
- Outputting images, PDF files, Flash movies

- Support for a wide range of databases

- Text processing
- You also have the choice of using procedural programming or object oriented programming (OOP), or a mixture of them both

What do we need?

- Support for PHP
- All files ending in **.php** are handled by PHP parser
- XAMPP installs PHP parser also
- Put .php in your htdocs directory and the server will automatically parse them for you

hello.php

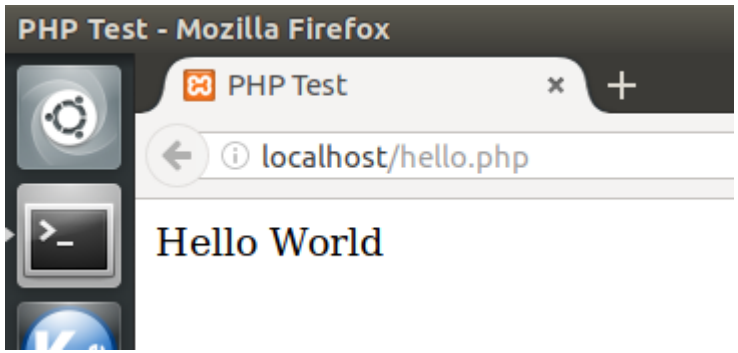
```
<html>
 <head>
  <title>PHP Test</title>
 </head>
 <body>
 <?php echo '<p>Hello World</p>'; ?>
 </body>
</html>
```

- (If hello.php is saved in htdocs , XAMPP Apache started, then)
  http://localhost/hello.php or http://127.0.0.1/hello.php

hello.php

```
<html>
 <head>
  <title>PHP Test</title>
 </head>
 <body>
 <?php echo '<p>Hello World</p>'; ?>
 </body>
</html>
```

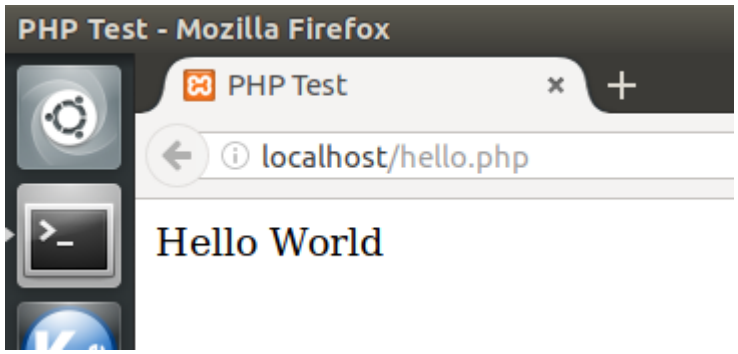
- http://localhost/hello.php or http://127.0.0.1/hello.php



hello.php

```
<html>
 <head>
  <title>PHP Test</title>
 </head>
 <body>
 <?php echo '<p>Hello World</p>'; ?>
 </body>
</html>
```

- http://localhost/hello.php

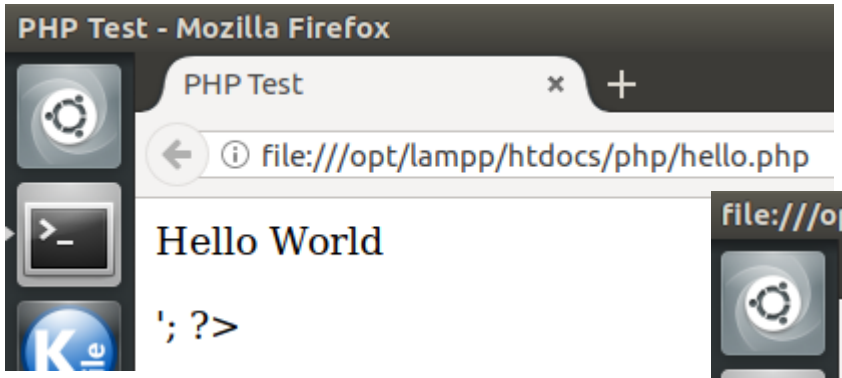


hello.php

- If you just call up the file from your file system

  Or

  Just double clicking on the php file (or right click, open with browser) , **then** it will **not be parsed** by PHP



PHP Presentation System

http://talks.php.net/

Overview of PHP

PHP has typical scripting language characteristics

- Dynamic typing, untyped variables
- Associative arrays
- Pattern matching
- Extensive libraries

Dynamic typing, untyped variables

- The type of a variable is not usually set by the programmer
- Rather, **type** is decided at runtime by PHP depending on the **context** in which that variable is used

- PHP supports ten primitive types

- Four scalar types: boolean , integer , float , string
- Four compound types: array , object , callable , iterable
- And finally two special types: resource , NULL

- To forcibly convert a variable to a certain type, either **cast** the variable or use the **settype()** function on it

Variables

- A variable that has not been assigned a value is unbound and has the value NULL
- NULL is coerced to 0 if a number is needed
- NULL is coerced to empty string if a string is needed
- Both the coercions count as boolean FALSE
- Characters in PHP are one byte
- The boolean type has two values :TRUE and FALSE

PHP Syntax

- PHP statements are terminated with **semicolons ;**
- All variable names in PHP begin with **$**
- Variable names are **case sensitive**
- One line comments can begin with **#** or **//** and continue to the end of the line
- Multi-line comments begin with **/*** and end with ***/**
- Curly braces are used to create compound statements

Dynamic typing, untyped variables

```php
<?php

   $bool   = TRUE;    // a boolean
   $str1   = "foo";   // a string
   $str2   = 'foo';   // a string
   $int    = 12;      // an integer

?>
```

```
php types.php
```

Dynamic typing, untyped variables

```php
<?php

   $bool  = TRUE;   // a boolean
   $str1  = "foo";  // a string
   $str2  = 'foo';  // a string
   $int   = 12;     // an integer

   echo gettype($bool);  // prints out ?
   echo gettype($str1);  // prints out ?

?>
```

Dynamic typing, untyped variables

```php
<?php

  $bool  = TRUE;   // a boolean
  $str1  = "foo";  // a string
  $str2  = 'foo';  // a string
  $int   = 12;     // an integer

  echo "\n int = $int ";

  if ( is_int ($int) )
    {
      $int += 4;
    }

  echo "\n int = $int ";

?>
```

Associative arrays

- Zero indexed

```php
<?php

  //Creates an array with elements.
  $theVariable = array("A", "B", "C");

  print "\n theVariable[1] = $theVariable[1]";

?>
```

[MATLAB , Octave array index begin with 1]

Associative arrays

```php
<?php

  //Creating Associaive array.
  $theVariable = array(
                          1 => "http://duckduckgo.com",
                          2 => "http://google.com");

  print "\n theVariable[1] = $theVariable[1]";

?>
```

Associative arrays

```php
<?php

  //Creating Associaive array with named keys
  $theVariable = array
                 (
                  "google"     => "http//google.com",
                  "duckduckgo"=> "http://duckduckgo.com");

  $searchEngine="google";
  print "\n google = $theVariable[$searchEngine]";

?>
```

[Author : Image : AKBYS]

Pattern matching

- Parle deals with **par**sing and **le**xing
- Parle pattern matching , PCRE : Regular Expressions (Perl-Compatible)
- Parser is LALR(1)

```php
<?php


use Parle\Token;
use Parle\Lexer;
use Parle\LexerException;

<?php


use Parle\{Parser, ParserException, Lexer, Token};
```

Extensive libraries

- ImageWorkshop : manipulate images with layers
- Goutte : scraping websites and extracting data
- Mustache : templating language
- Omnipay : payment processing library for PHP
- HTMLPurifier (on github) : HTML filtering library
- phpgeo : calculating distances between geographic coordinates

phpgeo

Operations

Arithmetic Operators and Expressions

- PHP supports the usual operators supported by the C / C++ / Java family

String Operations

- String catenation is indicated with a period
- Characters are accessed in a string with a subscript enclosed in **curly braces**

- The C printf function is also available

- identity operator **===**

Form Handling

- The values from forms can be accessed in PHP using the $_POST and $_GET arrays

Locking Files

- flock function will lock a named file
-

PHP, function documentation

- Doc , help

Cookies

- Cookies are a mechanism for storing data in the remote browser and thus tracking or identifying return users
- HTTP is a stateless protocol
- Server treats each request as completely separate from any other
- A shopping cart is an object that must be maintained across numerous requests and responses
- The mechanism of cookies can be used to help maintain state by storing some information on the browser system

[http://sg2.php.net/manual/en/features.cookies.php]

Cookies

- A cookie is a **key/value** pair that is keyed to the domain of the server
- This key/value pair is sent along with any request made by the browser of the same server
- A cookie has a lifetime which specifies a time at which the cookie is deleted from the browser

Cookies

- Cookies are only returned to the server that created them
- Cookies can be used to determine usage patterns that might not otherwise be ascertained by a server
- Browsers generally allow users to limit how cookies are used
- Browsers usually allow users to remove all cookies currently stored by the browser
- Systems that depend on cookies will fail if the browser refuses to store them

PHP Support for Cookies

- PHP provides the **setcookie** function to set a cookie in a HTTP response
- So setcookie( ) must be called before any output is sent to the browser
  [Like header( ) in PERL]

- Any cookies sent to server from the client will automatically be included into a **$_COOKIE** auto-global array

PHP Support for Cookies

- PHP provides the setcookie function to set a cookie in a HTTP response
  - First parameter is the cookie's name
  - Second, optional, parameter gives the cookie's value
  - Third, optional, parameter gives the expiration
- The cookie must be set before setting content type and before providing any other output

9. Pseudo Code / Outline of the Algorithm

```
<?php

  setcookie("name","value",time()+$int);
  /*name is your cookie's name
  value is cookie's value
  $int is time of cookie expires*/

  // time() returns ?
  // time() in PERL ?

?>
```

[http://sg2.php.net/manual/en/function.time.php]

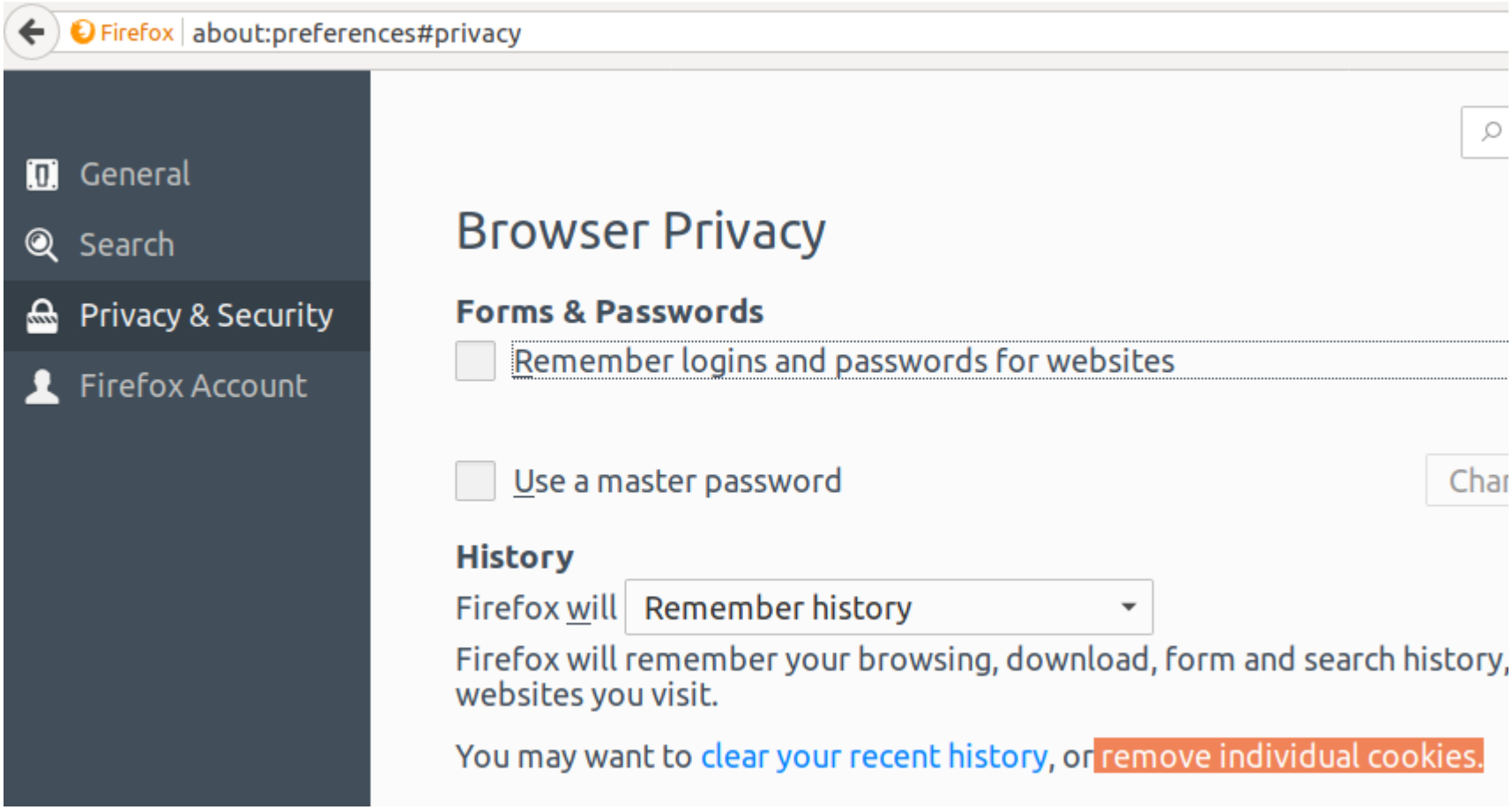
9. Pseudo Code / Outline of the Algorithm

```
<?php

  setcookie("name","value",time()+$int);
  /*name is your cookie's name
  value is cookie's value
  $int is time of cookie expires*/

  // time() Returns the current time measured in the
  //          number of seconds since the Unix Epoch

  // And Unix Epoch is ?



?>
```

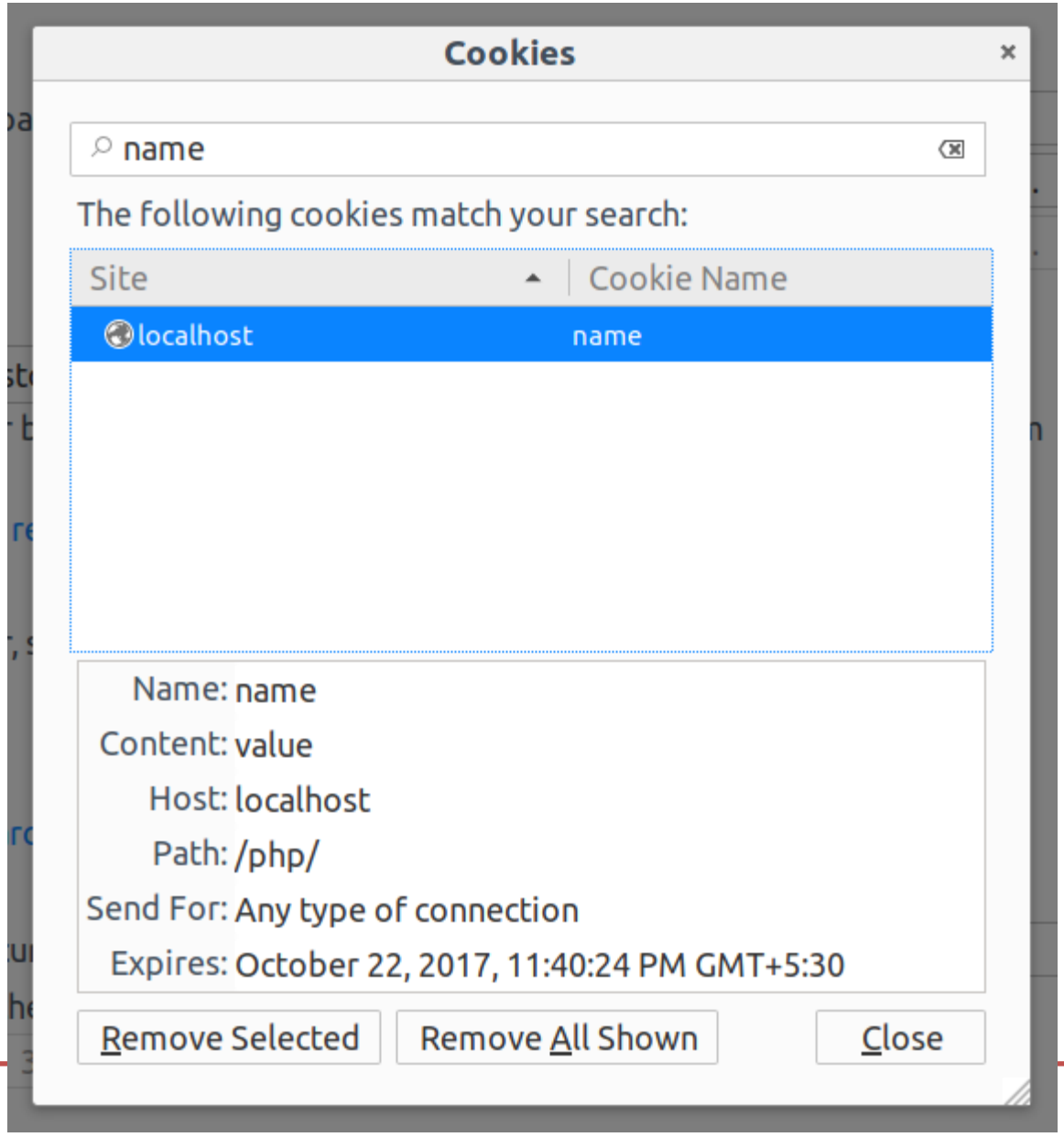
9. Pseudo Code / Outline of the Algorithm

```
<?php

  setcookie("name","value",time()+$int);
  /*name is your cookie's name
  value is cookie's value
  $int is time of cookie expires*/

  // time() Returns the current time measured in the
  //          number of seconds since the Unix Epoch

  // And Unix Epoch is (January 1 1970 00:00:00 GMT)

  // And yes, same return value as time() in PERL

?>
```

9. Pseudo Code / Outline of the Algorithm



9. Pseudo Code / Outline of the Algorithm



9. Pseudo Code / Outline of the Algorithm

- Where is it saved on hard disk?

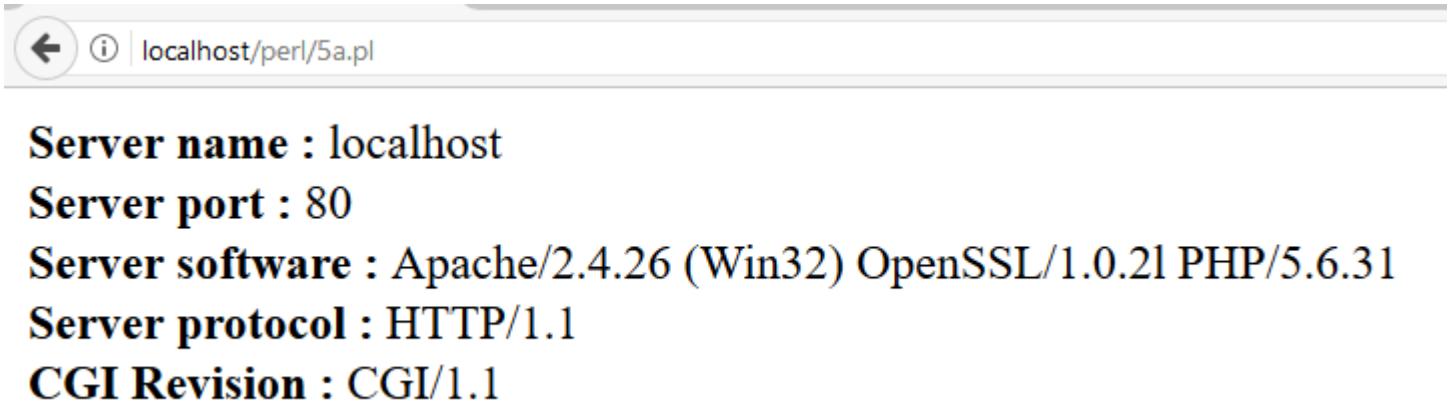
9. Pseudo Code / Outline of the Algorithm

date( )

- P
- m is month
-

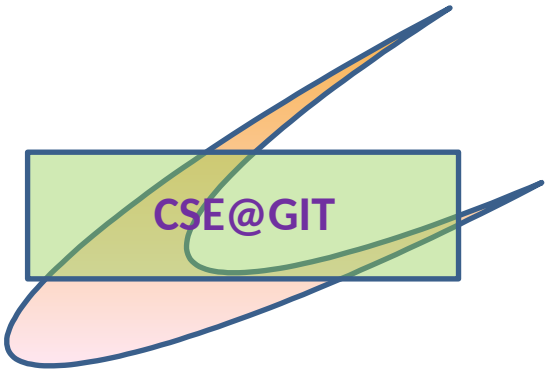
[http://sg2.php.net/manual/en/function.date.php]

Sample Run


localhost/perl/5a.pl
Server name : localhost
Server port : 80
Server software : Apache/2.4.26 (Win32) OpenSSL/1.0.2l PHP/5.6.31
Server protocol : HTTP/1.1
CGI Revision : CGI/1.1

Experiment No. 10

Problem Definition: 10. Write a PHP program to store page views count in SESSION, to increment the count on each refresh, and to show the count on web page.



Session Tracking

- Some applications need to keep track of a session
- Sessions are represented internally in PHP with a session id
- A session consists of **key/value** pairs
- A session can be initialized or retrieved by using the **session_start** function

  Guess what is it saved as

  An array , good.

- This function retrieves **$_SESSION**, an array containing the **key/value** pairs for each cookie in the current request

Session Tracking

- session_start()
- isset()
- $_SESSION[]
-

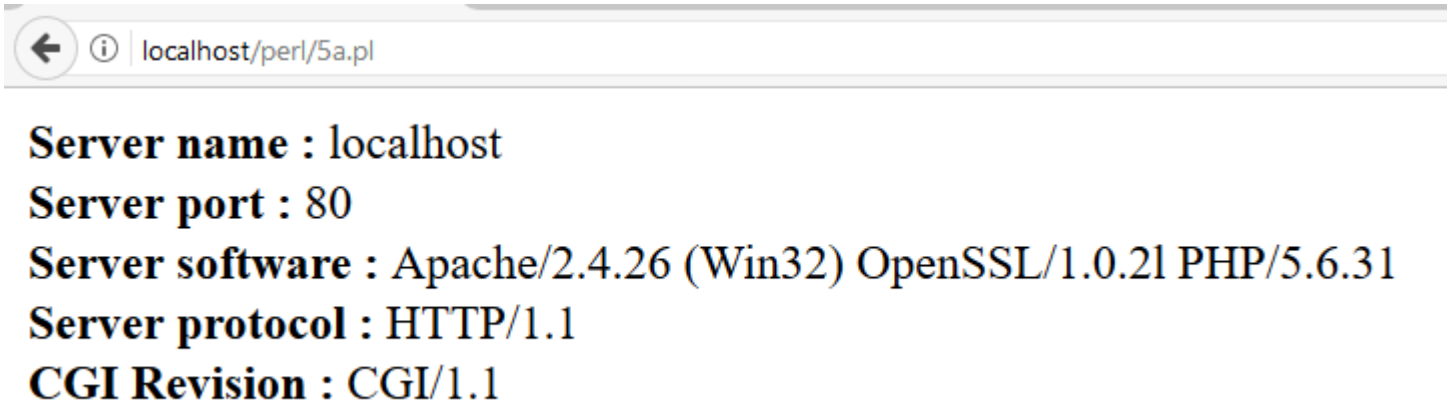
Session Log , .log file

- session_start()
- isset()
- $_SESSION[]
-

10. Pseudo Code / Outline of the Algorithm

```
print "<br/> <b> Server name :</b> ", server_name() ,
      "<br/> <b> Server port :</b> ", server_port(),
      "<br/> <b> Server software :</b> ", server_software(),
      "<br/> <b> Server protocol :</b> ", server_protocol();
```

Sample Run


localhost/perl/5a.pl
Server name : localhost
Server port : 80
Server software : Apache/2.4.26 (Win32) OpenSSL/1.0.2l PHP/5.6.31
Server protocol : HTTP/1.1
CGI Revision : CGI/1.1

Learning Outcomes of the Experiment

At the end of the session, students should be able to :

1) Experiment with the database connections, query using Perl [L3]

Acknowledgement ECE