LAB 6: CHUỖI VÀ LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

1. Mục tiêu

- Nắm vững kiến thức về chuỗi và các hàm xử lý chuỗi.
- Hiểu regular expression và áp dụng xử lý các kiểm tra chuỗi phức tạp.

2. Tóm tắt lý thuyết

String:

- Chuỗi là một thành phần rất quan trọng và được sử dụng ở mọi nơi trong các trang web.
- Có rất nhiều hàm trong php được xây dựng để người sử dụng thuận lợi thao tác với chuỗi. Một số hàm thông dụng với String.
  - strlen — Trả về chiều dài của chuỗi. $n = strlen("abc"); //$n=3
  - rtrim — cắt các khoảng trắng (hoặc các ký tự khác) từ cuối chuỗi
  - string rtrim (string $str [, string $character_mask])
  - $str: chuỗi cần cắt, $character_mask: chuỗi ký tự cần bỏ.
  - Hàm trả về chuỗi đã cắt bỏ các ký tự yêu cầu.
  - ltrim — Giống rtrim, nhưng loại bỏ các ký tự bên trái chuỗi.
  - trim — Loại bỏ các ký tự yêu cầu đầu và cuối chuỗi.

```
<?php $s1="abc**";
$s2="   abc   ";
$str1 = trim($s1, "*"); //$str1 ="abc"
$str2= trim($s2); //$str2 = "abc"
?>
```

  - Explode(): Tách chuỗi thành mảng.
    array explode (string $delimiter , string $string).
    $delimiter: chuỗi phân cách.
    $string: chuỗi cần tách
    Kết quả trả về một mảng sau khi tách chuỗi.
  - Implode(): Ghép các phần tử của mảng thành chuỗi.
    String implode (string $glue , array $pieces)
    $glue: Chuỗi kết nối, $pieces: Mảng cần ghép các phần tử
    Kết quả: trả về chuỗi đã gắn kết các phần tử $glue và giữa các phần tử mảng.

```
<?php
$s ="DH-001-ABC";
```

```
$a = array("SP1", "SP2");
$arr = explode("-", $s);
$str = implode("*",$a);
print_r($arr);
echo $str;
?>
```

Kết quả:

```
Array
(
   [0] => DH
   [1] => 001
   [2] => ABC
)
SP1*SP2
```

- str_word_count: Trả về thông tin các từ trong một chuỗi: Đếm số từ trong chuỗi, mảng chứa các từ, các ký vị trí bắt đầu một từ,…
  mixed str_word_count (string $string [, int $format = 0 [, string $charlist]])
  $string: Chuỗi được sử dụng để lấy thông tin. $format: là một số nguyên có các giá trị 0,1, 2.
  0 – Trả về số từ tìm được.
  1: Trả về mảng chứa các từ trong chuỗi.
  2. Trả về mảng kết hợp có index là vị trí của từ, và giá trị là từ tìm được trong chuỗi.

- MD5, SHA1: là các hàm sử dụng để mã hóa một chuỗi thành một chuỗi mới. Các hàm này hay áp dụng trong việc mã hóa chuỗi mật khẩu trước khi lưu vào CSDL.
- number_format — Format một số cùng nhóm các số ở vị trí hàng ngàn.
  number_format ($number , $decimals = 0 , $dec_point = "." , $thousands_sep = ",")
  $number: số được format, $decimals: số ký số thập phân được in, $dec_point: ký tự phân cách phần thập phân và nguyên, ký tự phân cách cho các nhóm số hàng ngàn.

```
<?php
$number = 1234.56;
$number1 = number_format($number);
```

```
// english notation (default)
echo $number1; // 1,235
?>
```

- htmlentities (string)— chuyển tất cả các ký tự liên quan đến html (>, <) sang dạng thực thể. Hàm này thường sử dụng cho việc xử lý dữ liệu liên quan đến html khi nhập vào database
- strip_tags($string, $allow_tags): Loại bỏ các thẻ HTML hoặc PHP ra khỏi chuỗi, thường được sử dụng để xử lý dữ liệu do người dùng nhập trước khi lưu trữ database, hiển thị văn bản dạng trích dẫn.
- $allow_tags: Các thẻ cho phép giữ lại Ví dụ: Loại bỏ các thẻ HTML ra khỏi chuỗi $str, cho phép giữ lại các tag trong $allow_tags.
- NL2br($str) — Thay thế ký tự xuống dòng (\n) bằng thẻ xuống dòng (
) trong chuỗi $str. Hàm này đường dùng để hiển thị nội dung của file văn bản trên trình duyệt web.
- substr — Hàm lấy một phần của chuỗi.
  substr($string, $start, $length)
  Trả về chuỗi con trong $string tính từ vị trí $start và lấy $lenght ký tự. Ký tự đầu tiên trong chuỗi bắt đầu tính từ vị trí 0.
- str_shuffle ($str): Đảo ngẫu nhiên các ký tự trong chuỗi $str. Kết quả trả về chuỗi đã đảo ngẫu nhiên.
- Strtolower($str): Chuyển chuỗi sang lowercase của chuỗi $str. Hàm trả về kết quả đã xử lý.
- strtoupper ($str): Chuyển chuỗi sang uppercase (chữ hoa)
- ucwords ($str): Viết hoa ở mỗi ký tự đầu tiên của từ.
  $str: Chuỗi cần xử lý. Hàm trả về chuỗi sau khi đã xử lý
- ucfirst ($str): Viết hoa từ đầu tiên của câu trong chuỗi
  $str: chuỗi cần xử lý. Hàm trả về kết quả sau khi xử lý

Biểu thức chính quy - regular expression:

- Biểu thức chính quy (regular expression- Regex): là một chuỗi miêu tả một bộ các chuỗi khác, theo những quy tắc cú pháp nhất định.
- Biểu thức chính quy thường được dùng trong các trình biên tập văn bản và các tiện ích tìm kiếm và xử lý văn bản dựa trên các mẫu được quy định.
- Sử dụng biểu thức chính qui đơn giản hơn nhiều trong lập trình và quá trình xử lý văn bản, và có những vấn đề sẽ không thể giải quyết được nếu không sử dụng regxp.
- Regular expression là một công cụ mạnh mẽ trong việc thao tác và trích xuất văn bản trên máy tính. Do đó nắm vững các biểu thức chính quy sẽ giúp tiết kiệm nhiều thời gian và công sức.

- Trong php, ta hay sử dụng các hàm: preg_match, preg_match_all và preg_replace để kiểm tra, tách dữ liệu và thay thế chuỗi.
- Các hàm hay sử dụng với Regula Expression:
    - Hàm preg_match: sử dụng để kiểm tra và tách dữ liệu của một chuỗi. preg_match($pattern, $string[, $match]);
      $pattern: Mẫu tìm kiếm, là một chuỗi được bắt đầu và kết thúc bằng ký tự: / . $string: Chuỗi input được sử dụng cho tìm kiếm. $match: Nếu $match được cung cấp, thì đây là mảng nhận kết quả trả về. Nếu có, ta có thể thấy các kết quả trong mảng $match[0], $match[1],…
      Hàm tìm kiếm trong $string theo mẫu $pattern. Nếu kết quả tìm thấy, hợp lệ, hàm trả về true và các kết quả đặt trong $match. Nếu không tìm thấy: trả về false
    - Hàm: preg_replace: sử dụng để tìm kiếm và thay thế dữ liệu.
      preg_replace ($pattern , $replacement , $subject [, $limit = -1 [, &$count]])
      $pattern: Mẫu cần tìm kiếm (mảng hay chuỗi). $replacement : giá trị thay thế (mảng hay chuỗi). $subject: mảng hay chuỗi để tìm kiếm và thay thế. $limit: số lần tìm kiếm thay thế. Default = -1: no limit
      $count: số lần được thay thế.
    - Hàm preg_match_all: giống hàm preg_match, nhưng trả về tất cả các kết quả trong $match.
- Pattern: là một chuỗi biểu thức có quy tắc (Regular expression - Regex) để mô tả những chuỗi khác.
    - Một pattern là chuỗi bắt đầu và kết thúc bằng ký tự /.
    - Trong regex có 2 loại ký tự: Ký tự thông thường: tập các chữ cái …, ký tự đặc biệt: những ký tự có ý nghĩa khác với giá trị của nó. ^, :, +, …
    - Các ký tự đặc biệt được gọi là các metacharacter. [, \, ^, $, ., |, ?, *, +, (,),… Nếu muốn sử dụng các ký tự đặc biệt này giống như những ký tự thông thường, ta thêm ký tự backslash (\) vào trước ký tự đó.
    - Biểu thức qui tắc sẽ so sánh các ký tự phân biệt các giá trị ký tự hoa và thường. Để so sánh không phân biệt chữ hoa và thường, ta sử dụng ký tự i sau chuỗi pattern. Ví dụ: $pattern="/dog/i" sẽ không phân biệt hoa-thường khi so sánh (match) mẫu.
    - Các ký tự đặc biệt và ý nghĩa

Ký tự	Ý nghĩa
^	Bắt đầu một chuỗi
$	Kết thúc chuỗi

.	Đại diện cho một ký tự
*	Lặp lại ký tự hay cụm ký tự đứng trước nó (>=0) Ví dụ: a* : nghĩa là lặp lại chữ a từ 0 tới n lần. -123a: -123aaa -123
+	Lặp lại ký tự hay cụm ký tự đứng trước nó (>=1) Ví dụ: $pattern="/a+/" : nghĩa là lặp lại ký tự a và phải có ít nhất là một ký tự a. - 123a: 1(đúng) - 123aa: 1(đúng) - 123 : 0 (sai). vì không có chữ a nào được lặp. $patterm="/ax+/" : biểu diễn cho các chuỗi: ax, axx, axxxx - nhưng không thể là "a" vì + đại diện cho >=1 ký tự.
?	Tồn tại hay không tồn tại ký tự hay cụm ký tự đứng trước nó: $pattern="/ab? /" :biểu diễn cho: a, ab (b có thể có hoặc không)
\	Dấu \ đi kèm với 1 meta symbol sẽ làm mất ý nghĩa của meta symbol đó - trả về symbol bình thường. $pattern="/a\?b/": Tìm chuỗi chứa chuỗi con "a?b" vì ? Là ký tự đặc biệt, nên đặt \? để nó trở thành ký tự thường.
\d	Biểu diễn cho một số bất kỳ: 0-9: $p ="/^\d/"; Tìm chuỗi bắt đầu bằng một ký tự số
\D	Biểu diễn một ký tự không phải số (ngược với \d):
\w	Biểu diễn một ký tự bất kỳ: a-z, A-Z, 0-9
\W	Biểu diễn một ký tự đặc biệt (không phải a-z, A-Z, 0-9)
\s+	Có ít nhất một khoảng trắng
()	Gom các ký tự thành một nhóm: $pattern ="/\w\w(123)/"; Tìm các chuỗi có chứa chuỗi 123 và trước chuỗi 123 là 2 ký tự. Ví dụ chuỗi: "ab0q123d4
{}	Lặp ký tự hay cụm ký tự trước đó với 1 số lần cho trước. {n}: n lần {n, }. Lặp lại >= n lần {n, m} Lặp lại từ n đến m lần Ví dụ: $pattern="/\w\w(\d\d){2,4}/"; Tìm các chuỗi bắt đầu bằng 2 ký tự (số hay a-z), sau đó là 2 ký tự số lặp lại từ 2 đến 4 lần. Hợp lệ: "ab0q1253d4"
|	Toán tử or, để lựa chọn hoặc cái này hoặc cái kia.
[]	Chỉ đoạn ký tự cho phép [a-z]: các ký tự thường từ a đến z [A-Z]: Các ký tự hoa từ A đến Z [0-9]: các ký tự số Ví dụ: $p ="/^(\(0[1-9]{1,2}\))?[0-9]{7,8}/";

	Thích hợp với: (08)66780052, 66780052, (056) 3821013

3. Ví dụ

3.1 Ví dụ lab6_1.php: Kiểm tra dữ liệu nhập vào từ form với các hàm đơn giản xử lý về chuỗi.

- Tên đăng nhập: có độ dài tối thiểu 6 ký tự (không tính ký tự khoảng trắng ở 2 đầu chuỗi).
- Mật khẩu: độ dài tối thiểu là 8 và 2 trường: mật khẩu và mật khẩu nhập lại phải bằng nhau.
- Họ tên: Phải có ít nhất 2 từ.

Nếu kết quả nhập vào hợp lệ, in ra thông tin đã nhập với họ tên có các ký tự đầu tiên là ký tự hoa, ngược lại báo lỗi.

3.2 Ví dụ lab6_2.php: Chứa một form để nhập file và gửi lên server. Server nhận file và in ra phần mở rộng của file và một mật khẩu ngẫu nhiên. Ví dụ này chứa:

- Hàm nhận vào một chuỗi là tên một file. Hàm sẽ trả về phần mở rộng của file này.
- Hàm phát sinh ra một chuỗi là mật khẩu ngẫu nhiên. Hàm này nhận vào một số nguyên dương n và trả về một chuỗi có giá trị là n ký tự ngẫu nhiên trong chuỗi: $s="abcdefghijkmlnopqrstuvxyz0123456789";

3.3 Ví dụ lab6_3.php: Kiểm tra dữ liệu nhập từ form sử dụng regular expression, với các điều kiện kiểm tra nâng cao.

a. Username: Độ dài tối thiểu là 6 và chỉ chứa các ký tự số, dấu chấm, alphabet
b. Email: Phải là email có định dạng hợp lệ.

4. Vận dụng

4.1 Sửa lại ví dụ lab6_1.php để:

- In ra mật khẩu đã được mã hóa bằng MHA1 hoặc kết hợp cả 2 giải thuật: SHA1 và MD5.
- Thêm textArea name=thong_tin và:
  - Loại bỏ các thẻ HTML trong field thong_tin
  - Thay thế các ký hiệu xuống hàng (\n) bằng các thẻ
 trong chuỗi nhận từ field thongtin và xuất kết quả đã xử lý ra màn hình.
  - Thêm ký tự \ trước các ký tự nháy đơn (‘) trong nội dung thong_tin khi gửi lên server.

- Loại bỏ các ký tự (\) trước các ký tự đặc biệt của chuỗi. (‘).

4.2 Sửa lại lab6_3.php. Thêm các thông tin nhập vào trong form và viết các hàm sử dụng Regular expression kiểm tra dữ liệu nhập của các thông tin sau.

- Mật khẩu phải có tối thiểu 8 ký tự và phải có ít nhất một ký tự số, một ký tự hoa, một ký tự thường.
- Số điện thoại: phải nhập số.
- Ngày sinh: Phải là có định dạng ngày (dd/mm/yyyy hoặc dd-mm-yyyy).

4.3 Lấy và phân tích dữ liệu từ các website khác.

a. Bật chức năng allow_url_fopen trong file php.ini: allow_url_fopen = On
b. Sử dụng hàm : file_get_contents(url) để đọc dữ liệu html từ website khác. (ví dụ url là: http://thethao.vnexpress.net/)
c. In kết quả ra màn hình và phân tích cấu trúc của nội dung này.

```
<?php
$content = file_get_contents("http://thethao.vnexpress.net/");
echo $content;
?>
```

d. Phân tích cấu trúc để lọc và in ra các thông tin chọn lọc (các thông tin nằm trong các thẻ …
   - Xây dựng pattern và sử dụng hàm preg_match_all để lấy dữ liệu.
   - In dữ liệu ra màn hình bằng hàm print_r.

```
<?php
$content = file_get_contents("http://thethao.vnexpress.net/");
//echo $content;
$pattern='/<div class="title_news">.*<\/div>/imsU';
preg_match_all($pattern, $content, $arr);
print_r($arr);
?>
```

e. Duyệt qua mảng trên, để trích lấy dữ liệu cần thiết và đưa vào hiển thị trong table.

5. Nâng cao

5.1 Tìm kiếm và thay thế dữ liệu:

a. Lọc tất cả các link trong trang web.
b. Lọc tất cả các địa chỉ email, điện thoại trong các trang web.
c. Kiểm tra tên hình ảnh được đưa vào có đúng qui định hay không.

5.2 Hãy phân tích một số trang báo điện tử, sử dụng kỹ thuật quét tin ở phần trên và hãy quét, in tiêu đề các tin mục **xã hội** lên trình duyệt web.

a. http://news.zing.vn/
b. http://dantri.com.vn/
c. http://vietnamnet.vn/