1. Regular Expressions

Sử dụng biểu thức chính quy bạn có thể:

- Tìm kiếm một chuỗi cụ thể bên trong một chuỗi khác
- Thay thế một chuỗi bằng một chuỗi khác
- Chia một chuỗi thành nhiều khối.

Phân loại:

- POSIX Regular Expressions
- PERL Style Regular Expressions

A. POSIX Regular Expressions

Cấu trúc của biểu thức chính quy POSIX không giống với biểu thức số học điển hình: các toán tử được kết hợp để tạo thành các biểu thức phức tạp hơn.

Ví dụ: Biểu thức chính quy đơn giản nhất là một ký tự khớp với một ký tự đơn lẻ, chẳng hạn như *'g'*, bên trong các chuỗi như '*g'*, '*haggle'*, hoặc '*bag'*.

Expressions	Description
[0-9]	Phù hợp với bất kỳ chữ số thập phân từ 0 đến 9.
[a-z]	Phù hợp với bất kỳ ký tự nào từ chữ thường sang chữ hoa chữ z.
[A-Z]	Phù hợp với bất kỳ ký tự nào từ chữ hoa chữ cái A thông qua chữ hoa Z.
[a-Z]	Phù hợp với bất kỳ ký tự từ chữ thường từ a sang chữ hoa Z.
[0-3]	Khớp bất kỳ chữ số thập phân từ 0 đến 3
[b-v]	Khớp bất kỳ ký tự chữ thường nào, từ b đến v
p+	Khớp bất kỳ chuỗi nào có chứa ít nhất một kí tự 'p'
p*	Khớp bất kỳ chuỗi nào không chứa 'p' hoặc nhiều hơn 'p'
p?	Khớp bất kỳ chuỗi nào không chứa 'p' hoặc một số của 'p'.
p{N}	Khớp bất kỳ chuỗi nào chứa một dãy **N** kí tự 'p'
p{2,3}	Phù hợp với bất kỳ chuỗi nào có chứa một dãy gồm hai hoặc ba p
p{2, }	Phù hợp với bất kỳ chuỗi nào có chứa một dãy ít nhất hai p
p$	Phù hợp với bất kỳ chuỗi kết thúc là 'p'

^p	Phù hợp với bất kỳ chuỗi ngoại trừ ‘p’
[^a-zA-Z]	Chỉ cho phép các kí tự ngoài a đến z và A đến Z.
p.p	Phù hợp với bất kỳ chuỗi bắt đầu là ‘p’, tiếp theo là bất kỳ kí tự nào và kết thúc là ‘p’.
^.{2}$	Khớp với bất kỳ chuỗi nào có chứa chính xác hai ký tự.
(.*)	Phù hợp với bất kỳ chuỗi nào được bao trong và
p(hp)*	Phù hợp với bất kỳ chuỗi có chứa một ‘p’ theo sau bởi không hoặc nhiều trường hợp của chuỗi ‘php’.

B. PERL Style Regular Expressions

Meta characters

Sử dụng \ trước các kí tự đặc biệt

Ví dụ: bạn có thể tìm kiếm các khoản tiền lớn bằng cách sử dụng '\d': **/([\d]+)000/**

Trong đó: \d sẽ tìm kiếm bất kỳ kí tự số.

Character	Description
.	1 kí tự duy nhất
+	Lặp lại ký tự hay cụm ký tự đứng trước nó
\s	Một ký tự khoảng trắng (khoảng cách, tab, dòng mới)
\S	Ký tự không chứa khoảng trắng
\d	Một chữ số trong khoảng 0-9
\D	Không phải số
\w	Một ký tự từ, gồm (a-z, A-Z, 0-9, _)
\W	Ngoại trừ (a-z, A-Z, 0-9, _)
[abcd]	Khớp một ký tự đơn trong tập hợp đã cho (Một ký tự duy nhất của: a, b hoặc c)
[^abcd]	Khớp một ký tự bên ngoài tập hợp a b c d
(foo|bar|baz)	Phù hợp bất kỳ lựa chọn nào được chỉ định
^	Bắt đầu dòng
$	Kết thúc dòng
\A	Bắt đầu chuỗi
\z	Kết thúc chuỗi

Modifiers

Modifiers	Description
i	Tìm kiếm không phân biệt hoa, thường
m	Xử lý như chuỗi nhiều dòng
s	Cho phép sử dụng . để khớp ký tự dòng mới

x	Bỏ qua khoảng trắng trong regex (Cho phép bạn sử dụng khoảng trắng)
A	Chỉ khớp ở đầu chuỗi
D	Chỉ khớp ở cuối chuỗi
U	Không lấy thêm data, chỉ kết hợp theo mặc định

PHP's Regexp PERL Compatible Functions

1. **preg_match()**

```
int preg_match (string pattern, string string [, array pattern_array], [, int $flags [, int $offset]]]);
```

Tìm kiếm string trong pattern, trả lại đúng nếu pattern tồn tại, và sai nếu không.

VD:

```
<?php
   $line = "Vi is the greatest word processor ever created!";
   // perform a case-Insensitive search for the word "Vi"

   if (preg_match("/Vi/i", $line, $match)) :
      print "Match found!";
      endif;
?>
```

ÁP DỤNG:

1. Viết biểu thức chính quy kiểm tra chuỗi “Hello” có tồn tại trong chuỗi “Hello World!” không?

2. Viết biểu thức chính quy kiểm tra chuỗi “Hello” có xuất hiện ở vị trí đầu tiên trong chuỗi “Hello World!” không?
3. Viết biểu thức chính quy kiểm tra chuỗi có phải là số và có 3 chữ số hay không
4. Viết biểu thức chính quy kiểm tra một địa chỉ email:
   - Emai gồm các kí tự a-z, A-Z, 0-9 và giới hạn từ 3 - 50 kí tự
   - Tiếp theo bao gồm @
   - Tên domain gồm các kí tự a-z, A-Z, 0-9 và giới hạn từ 3 - 10 kí tự
   - Tiếp theo là .com hoặc .abc ….
5. Viết biểu thức chính quy kiểm tra một URL: ví dụ: http://domain.com hoặc https://domain.com

6. Viết biểu thức chính quy kiểm tra password thỏa mãn điều kiện:
   - Kí tự đầu tiên phải là chữ in hoa
   - Các kí tự tiếp theo được phép từ a-z A-Z và 0-9
   - Độ dài từ 6 - 8 kí tự

2. preg_match_all()

```
int preg_match_all (string pattern, string string, array pattern_array [, int order]);
```

Khớp tất cả các lần xuất hiện của mẫu trong chuỗi.

Trả về số lần khớp.

VD:

```php
<?php
   $userinfo = "Name: <b>John Poul</b> <br> Title: <b>PHP Guru</b>";
   preg_match_all ("/<b>(.*)<\/b>/U", $userinfo, $pat_array);

   print $pat_array[0][0]." <br> ".$pat_array[0][1]."\n";
?>
```

a. preg_replace()

```
mixed preg_replace (mixed pattern, mixed replacement, mixed string [, int limit [, int
&$count]] );
```

Tìm kiếm string được chỉ định bởi pattern và thay thế `pattern` nếu tìm thấy

Sau khi thay thế đã xảy ra, chuỗi sửa đổi sẽ được trả lại.

Nếu không tìm thấy kết quả phù hợp, chuỗi sẽ không thay đổi.

VD:

```php
<?php
   $copy_date = "Copyright 1999";
   $copy_date = preg_replace("([0-9]+)", "2000", $copy_date);

   print $copy_date;
?>
```

b. preg_split()

```
array preg_split (string pattern, string string [, int limit [, int flags]]);
```

Chia một string thành các phần tử khác nhau, các ranh giới của mỗi phần tử dựa trên sự xuất hiện của pattern trong chuỗi.

Limit: Nếu giới hạn tham số đầu vào tùy chọn được chỉ định, sau đó chỉ giới hạn số lượng các chuỗi con được trả về.

Flags:

- PREG_SPLIT_NO_EMPTY: chỉ những phần không rỗng sẽ được trả về
- PREG_SPLIT_DELIM_CAPTURE: biểu thức trong ngoặc đơn sẽ được bắt và trả về
- PREG_SPLIT_OFFSET_CAPTURE : chuỗi phụ bù cũng sẽ được trả về.

VD:

```php
<?php
   $ip = "123.456.789.000"; // some IP address
   $iparr = split ("/\./", $ip);

   print "$iparr[0] <br />";
   print "$iparr[1] <br />" ;
   print "$iparr[2] <br />"  ;
   print "$iparr[3] <br />"  ;
?>
```

c. preg_grep()

```
array preg_grep ( string $pattern, array $input [, int $flags] );
```

Trả về mảng bao gồm các phần tử của mảng đầu vào phù hợp với pattern đã cho.

Nếu $flags được đặt thành PREG_GREP_INVERT, hàm này trả về các phần tử của mảng đầu vào *không khớp* với pattern đã cho.

VD:

```php
<?php
   $foods = array("pasta", "steak", "fish", "potatoes");

   // find elements beginning with "p", followed by one or more letters.
   $p_foods = preg_grep("/p(\w+)/", $foods);

   print "Found food is " . $p_foods[0];
   print "Found food is " . $p_foods[1];
?>
```