PHP IV

Regex in PHP

- PHP used to support two regular expression engines
  - POSIX
  - PCRE
- POSIX is deprecated as of PHP 5.3
- Because of this, all regular epxression functions you will use start with the prefix `preg_`
- All regex strings need a delimiter inside the quotes
  - `"/regex/"`

Matching

- PHP has three functions that perform some type of matching
  - `preg_match(regex,string,[captures])` - Return if the regex is found in the string, optionally capturing groups into provided array
  - `preg_match_all(regex,string,[captures])` - Finds all matches, storing them in the provided capture variable, and returns the number of matches found
  - `preg_grep(regex,array)` - Returns an array with all elements of the input array that match the regular expression

In [ ]:

```
$str = <<< END
The phone number for the CSEE Department is 410-455-3500.
The fax number is 410-455-3969.
END;

preg_match("/\d{3}-\d{3}-\d{4}/",$str);
```

In [ ]:

```
preg_match("/(\d{3})-(\d{3})-(\d{4})/",$str,$matches);
echo $matches[0];
echo $matches[1];
echo $matches[2];
echo $matches[3];
```

In [ ]:
```
preg_match_all("/(\d{3})-(\d{3})-(\d{4})/",$str);
```

In [ ]:

```
$n_matches = preg_match_all("/(\d{3})-(\d{3})-(\d{4})/",$str,
        $matches);
for($i = 0; $i < $n_matches; $i++){
    echo $matches[0][$i];
    echo $matches[1][$i];
    echo $matches[2][$i];
    echo $matches[3][$i];
}
```

In [ ]:
```
$strs = array('One','Bone','Phone','Tome','Rome');
preg_grep('/one/i',$strs)
```

In [ ]:

```
$strs = array('a'=>'One','b'=>'Bone',
              'c' => 'Phone','d' => 'Tome','e' => 'Rome');
preg_grep('/one/i',$strs)
```

In [ ]:
```
$strs = array('One','Bone','Phone','Tome','Rome');
preg_grep('/one/i',$strs,PREG_GREP_INVERT)
```

Regex Practice

- Find all URLs in a text string and print them

In [ ]:

```
$text = <<< END
<footer role="contentinfo" id="umbc-footer">

<div id="umbc-footer-logo" class="first-child">[<span class="first-child last-chil
d">UMBC:
An Honors University in Maryland</span>](https://www.umbc.edu "UMBC: An Honors Uni
versity in Maryland")</div>

<div id="usmd-footer-logo">[<span class="first-child last-child">Univeristy
System of Maryland</span>](http://www.usmd.edu/ "Univeristy System of Maryland")</
div>

<nav id="umbc-footer-nav" style="clear:both;">
    * [About UMBC](https://about.umbc.edu)
    * [Contact Us](mailto:homepage@umbc.edu)
    * [Equal Opportunity](https://www.umbc.edu/go/equal-opportunity)
    * Community: [<span class="none first-child last-child">Facebook</span>](http
s://www.facebook.com/umbcpage)
                 [<span class="none first-child last-child">Twitter</span>](htt
p://twitter.com/umbc)
                 [<span class="none first-child last-child">YouTube</span>](htt
p://www.youtube.com/user/UMBCtube)
                 [<span class="none first-child last-child">Retriever Net</span>]
(http://alumni.umbc.edu/)
</nav>

<div id="umbc-footer-info" class="last-child">© University of Maryland, Baltimor
e County <span class="bullet first-child">•</span> 1000 Hilltop C
ircle <span class="bullet last-child">•</span> Baltimore, MD 21250</div>

</footer>
END;
```

Replacing

- PHP has several varities of a replace function
- The main one is

```
preg_replace(regex,replacement,string)
```

  - This replaces all occurences of the pattern in the string with the replacement
  - Regex, replacement, and string can all be arrays of strings
- There other varities are similar but rather than a replacement string, they take a call back function

In [ ]:

```
$str = <<< END
Go directly to Jail.
Do not pass Go,
do not collect $200.
END;

$str = preg_replace('/(do not)/i', '<b>$1</b>', $str);
echo $str;
```

In [ ]:

```
$chipmunks = array("Alvin","Simon","Theodore");
$chipmunks = preg_replace('/^(.*)$/',"$1 Seville",$chipmunks);
foreach($chipmunks as $chip){
    echo $chip;
}
```

In [ ]:

```
$bad_words = array("/Fooey/i", "/Blurgh/i",
"/Smurf/i", "/Flipping/i");
$str = <<<HERE
Oh fooey, stop saying smurf, we are trying to
              air this on flipping network tv.
HERE;

preg_replace($bad_words,"*******", $str);
```

In [ ]:

```
$bad_words = array("/Fooey/i", "/Blurgh/i",
            "/Smurf/i", "/Flipping/i");
$replacements = array("F***","B****","S***","F*******");
$str = <<<HERE
Oh fooey, stop saying smurf, we are trying to air this on flipping network tv.
HERE;

preg_replace($bad_words,$replacements, $str);
```

In [ ]:
```
$lines = file('sherlock.txt');
preg_replace('/Sherlock Holmes/','Bond, James Bond',$lines);
```

In [ ]:

```
$lines = file('sherlock.txt');
$names = array('/Sherlock Holmes/','/Henry Baker/',
               '/Mr[s]\. \w+/');
preg_replace($names,'REDACTED',$lines);
```

Replacement Practice

- Replace all social security numbers in text with * for the first 5 numbers, and print the last 4 numberse

In [ ]:

```
$ssns = <<<END
078-05-1120 is no longer issued because so many people used it as their
own. Other fake social security numbers include 000-11-1111, and 123-00-1111.
Pretty much any number with all zeros in one location isn't a real number, such
as 1234-56-0000

END;
```

Quoting

- The `preg_quote(string)` function will place a backslash before any special character
- This can be useful if you want to do a find operation using user input
  - Assuming you want basic find, not regex find

In [ ]:
```
$string_to_search_for = "The cost of the vase was $100!!";
preg_quote($string_to_search_for);
```

Globbing

- Globbing uses shell style regular expressions
  - `*` is all matches, don't need `.*`
- The `glob` function returns a list of full file names

In [ ]:
```
glob("/home/bryan/Teaching/CMSC433/*.html")
```

Sorting

- PHP has numerous built-in routines to sort arrays, based on
  - Ascending or Descending order
  - By Key or by Value
  - By a user defined functions
- All sorts in PHP are done **in place**

Sorting the Standard Array

- The two simplest sort functions are
  - `sort`
  - `rsort`
- These functions sort an array by value, completely ignoring any key associations that are there

In [ ]:

```
$number_array = array(6,7,8,4,2,6,7,5,4,3);
sort($number_array);
$number_array;
```

In [ ]:

```
$number_array = array(6,7,8,4,2,6,7,5,4,3);
rsort($number_array);
$number_array;
```

In [ ]:

```
$string_array = array("asdf","rteu","qerw","vbnm");
sort($string_array);
$string_array;
```

In [ ]:

```
$string_array = array("asdf","rteu","qerw","vbnm");
rsort($string_array);
$string_array;
```

In [ ]:

```
$assoc_array = array('key1' => 'tiger',
                     'key2' => 'lion',
                     'k3' => 'cheeta');
sort($assoc_array);
$assoc_array;
```

In [ ]:

```
$mixed_array = array(5,6,4,3,1,"asdf","rteu",4,"qerw","vbnm","1");
sort($mixed_array);
$mixed_array;
```

In [ ]:

```
$mixed_array = array(5,6,4,3,1,"asdf","rteu",4,"qerw",
                     "vbnm","1");
sort($mixed_array,SORT_STRING);
$mixed_array;
```

Maintaining Association

- To sort by value, but keep thet associated keys, use one of the `a*sort()` functions
  - `asort()`
  - `arsort()`

In [ ]:

```
$assoc_array = array('key1' => 'tiger',
                     'key2' => 'lion',
                     'k3' => 'cheeta');
asort($assoc_array);
$assoc_array;
```

In [ ]:

```
$assoc_array = array('key1' => 'tiger', 'key2' => 'lion',
                     'k3' => 'cheeta');
arsort($assoc_array);
$assoc_array;
```

In [ ]:

```
$number_array = array(6,7,8,4,2,6,7,5,4,3);
asort($number_array);
$number_array;
```

Sorting by Key

- To sort by key, use one of the `k*sort()` functions
  - `ksort()`
  - `krsort()`
- There is no `aksort()` function
  - It isn't needed, `k*sort()` mantains associativity

In [ ]:

```
$cats = array('asia' => 'tiger',
              'africa' => 'lion',
              'south_america' => 'jaguar',
              'north_america'=>'bobcat');
ksort($cats);
$cats;
```

In [ ]:

```
$cats = array('asia' => 'tiger',
              'africa' => 'lion',
              'south_america' => 'jaguar',
              'north_america'=>'bobcat');
krsort($cats);
$cats;
```

In [ ]:

```
$number_array = array(6,7,8,4,2,6,7,5,4,3);
ksort($number_array);
$number_array;
```

User-Defined Sort

- To use a custom sort function, call one of the `u*sort` functions
  - `usort()`
  - `uasort()`
  - `uksort()`
- Each of these functions takes an array and a comparator function

In [ ]:

```
$semesters = array('Fall 2009','Spring 2010',
                   'Winter 2010','Summer 2010',
                   'Fall 2010');
shuffle($semesters);
$semesters;
```

In [ ]:

```
usort($semesters, function($a, $b) {
        $map = array('Winter'=>1, 'Spring'=>2,
                     'Summer'=>3, 'Fall'=>4);
        $aParts = explode(' ', $a);
        $bParts = explode(' ', $b);
        $result = $aParts[1] - $bParts[1];
        return $result == 0 ? $map[$aParts[0]] - $map[$bParts[0]] : $result;
    });
$semesters
```

In [ ]:

```
$cats = array('asia' => 'tiger',
              'africa' => 'lion',
              'south_america' => 'jaguar',
              'north_america'=>'bobcat');
uasort($cats,function($a,$b){
    return strcmp(substr($a,-1),substr($b,-1));
});
$cats;
```

In [ ]:

```
$cats = array('asia' => 'tiger',
              'africa' => 'lion',
              'south_america' => 'jaguar',
              'north_america'=>'bobcat');
uksort($cats,function($a,$b){
    return strlen($b) - strlen($a);
});
$cats;
```

Sorting Practice

- Given an array of of associative arrays, each representing facts about a movie, sort by average box office total per day of in theaters
- The array looks like this:

In [ ]:
```
$movies = array('Avengers' => array('box'=> 250000000, 'rating' => 'PG-13', 'days'
 => 3),
                'I Feel Pretty' => array('box' => 29000000 , 'rating' => 'PG-13',
 'days' => 10),
                'Sherlock Gnomes' => array('box' => 40000000, 'rating' => 'PG' ,
 'days' => 38),
                'Peter Rabbit' => array('box' => 114000000, 'rating' => 'PG', 'day
s'=>80 ),
                'A Quiet Place' => array('box' => 148173301, 'rating' => 'PG-13',
 'days'=>24 ),
                'Black Panther' => array('box' => 688000000, 'rating' => 'PG-13',
 'days'=>73 )
                );
```

Network Access

- PHP has excellent support for performing operations over a network
- Today we will look at two methods
  - Accessing content over a network using file IO
  - Using the cURL library (almost always isntalled)

Treating a URL like a file name

- All of PHP's file functions can use a URL inplace of a file name
  - You can't write to a URL unless you are using something like FTP and have permission
- You must include the scheme so PHP can handle it correctly
  - `http://`
  - `ftp://`
  - `file://`

In [ ]:
```
$website = file_get_contents("http://umbc.edu");
$website;
```

In [ ]:

```
preg_match_all("/<a href=\"(.*?)\">(.*?)<\/a>/",$website,
                $matches);
$matches[1];
```

In [ ]:

```
$website_fp = fopen('http://umbc.edu','r');
while($line = fgets($website_fp)){
        echo htmlspecialchars($line);
}
```

cURL

- cURl is short for Client URL
- The basic use of cURL is the following steps
  - Get a curl object using `curl_init()`
  - Set options on the curl object using `curl_setopt()`
  - Execute the curl request using `curl_exec()`
  - Close the curl session using `curl_close()`

cURL init

- `curl_init` can be called in one of two ways
  - With no parameters, to create an empty curl object
  - With a URL, which is the URL you will be interacting with

cURL Options

- The main functionaility of cURL are its numerous options which allow you to perform almost any kind of network request
- A full list of the predefined constants can be found at http://php.net/manual/en/curl.constants.php
  - The ones for setting options start with `CURLOPT_`
- To set an option use

```
curl_setopt(curl_object, OPTION_CONSTANT, value);
```

In [ ]:

```
$ch = curl_init('http://umbc.edu');
$result = curl_exec($ch);
curl_close($ch);
```

In [ ]:

```
$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, "http://umbc.edu");
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
$result =  curl_exec($ch);
curl_close($ch);
$result;
```

In [ ]:

```
$lat = 39.2556015207;
$long = -76.71099543;
$radius= 700;
if($radius > 10000){
        $radius = 10000;
}
$curl = curl_init("https://en.wikipedia.org/w/api.php?action=query&prop=coordinate
s|pageimages|pageterms|info|extracts&colimit=100&piprop=thumbnail&pithumbsize=144&
pilimit=50&wbptterms=description&generator=geosearch&ggscoord=${lat}|${long}&ggsra
dius=${radius}&ggslimit=100&ggsprop=type&format=json&inprop=url");
curl_setopt($curl, CURLOPT_USERAGENT, 'UMBC_CMSC_433_Project/1.0 (https://www.cse
e.umbc.edu/~bwilk1/433/; bwilk1@umbc.edu)');
curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
$data = curl_exec($curl);
echo print_r(json_decode($data),true);
curl_close($curl);
```

Get Info About Request

- After executing, you can retrieve additional information from the curl handle
  - Headers
  - Time information
  - IP information
- The `curl_getinfo` function can be passed an curl handle, and optionally flags to denote what info to get
  - If no flags are set, all the information is returned

In [ ]:

```
$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, "https://umbc.edu");
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
$r1 = curl_exec($ch);
$result = curl_getinfo($ch);
curl_close($ch);
$result;
```