Common PHP Bugs

This is a list of common PHP bugs that students (and TA's/instructors) often run into in their programs, as well as some debugging tips to find them! We highly recommend keeping a list of your own common bugs as you're writing your first PHP web services, as well as how you found each bug! We'll be working on adding to this document as we learn more PHP features.

The Bug: . vs. +
Explanation: Remember that to concatenate Strings in PHP, we use the . operator instead of +.
Example (Partial) Buggy Code:

```
$first = "Mowgli";
$last = "Hovik";
$fullname = $first + " " + $last;
echo "Hello " + $first + " " + $last + "!";
```

Fixed Code:

```
$first = "Mowgli";
$last = "Hovik";
$fullname = $first + " " + $last;
echo "Hello " . $first . " " . $last . "!";  # or echo "Hello {$first} {$last}!"
```

Debugging Tip: Run your PHP programs often! This bug can be subtle, especially if you have added a lot of code that could have other bugs.

The Bug: Forgetting `<?php` and `?>`
Explanation: Every PHP program needs to start with `<?php` and end with `?>`
Example (Full) Buggy Program:

```
echo “Hello world!”;
```

Fixed Code:

```
<?php
echo “Hello world!”;
?>
```

Debugging Tip: When you try to run your PHP program without these symbols, you will probably see the source code output rather than the result of the program executing.

The Bug: No new lines output despite having “\n” in echo/print statements.
Explanation: By default, PHP programs output to the body of an HTML page. Remember that HTML ignores whitespace characters and new-lines (unless it’s in a <pre> tag)! When outputting plain text, we need to specify the content type with the PHP header function.
Example (Partial) Buggy Code:

```
echo "Hello world!\n";
echo "PHP is awesome!";
```

Fixed Code:

```
header(“Content-type: text/plain”);
echo "Hello world!\n";
echo "PHP is awesome!";
```

Debugging Tip: Run your PHP programs often! This bug can be subtle, especially if you have added a lot of code that could have other bugs.

The Bug: Missing $ with loop variables.
Explanation: There’s a point where we can write for loops in our sleep in our favorite language. But PHP is unique in requiring $ before *every* reference to a variable. This includes for loop counters like i!
Example (Partial) Buggy Code:

```
for ($i = 0; i < 10; i++) {
  echo “{$i} squared is “ . ($i * $i) . “!\n”;
}
```

Fixed Code:

```
for ($i = 0; $i < 10; $i++) {
  echo “{$i} squared is “ . ($i * $i) . “!\n”;
}
```

Debugging Tip: Run your PHP programs often! This bug can be subtle, especially if you have added a lot of code that could have other bugs.

The Bugs: foreach loop syntax

- Reversing the order of $arr and $item in foreach loops.
- Using for instead for foreach

Explanation: foreach loops are very handy ways to loop through arrays (especially associative arrays with non-integer keys) in PHP! Their syntax is a bit new at first, so it's not uncommon to put the $arr and $item in the incorrect order, leading to some subtle PHP bugs.

Example (Partial) Buggy Codes:

```
foreach ($item as $arr) {
  ...
}

for ($arr as $item) {
  ...
}
```

Fixed Code:

```
foreach ($arr as $item) {
  ...
}
```

Debugging Tip: Run your PHP programs often! This bug can be subtle, especially if you have added a lot of code that could have other bugs.

The Bug: Using a GET parameter without checking if it has been passed as a query parameter.
Explanation: When writing web services, we can't assume the client will always pass all of the required GET parameters. If you access a $_GET parameter in PHP without it being defined (it will evaluate to NULL), you won't get an error until you try to use it where NULL can't be used. Sometimes, you won't get an error at all, such as printing out a message with the value of the $_GET parameter you assume was passed. Remember to always check whether $_GET parameters are set **as early as possible in your PHP web services.**

Example (Full) Buggy Program:

```
<?php
  header("Content-type: text/plain");
  $name = $_GET["name"];
  echo "Hello {$name}!";
?>
```

Fixed Code:

```
<?php
  header("Content-type: text/plain");
  if (isset($_GET["name"])) {
    $name = $_GET["name"];
    echo "Hello {$name}!";
  } else {
    echo "Missing required 'name' query parameter.";
  }
?>
```

Debugging Tip: Get in the habit early on with using isset to check your $_GET parameters *before* using them!
