

```
1. #!/usr/bin/env php
2. <?php
3.
4.     // ask user for an integer
5.     $n = readline("I'd like an integer please: ");
6.
7.     // analyze user's input
8.     if ($n > 0)
9.     {
10.         printf("You picked a positive number!\n");
11.     }
12.     else if ($n == 0)
13.     {
14.         printf("You picked zero!\n");
15.     }
16.     else
17.     {
18.         printf("You picked a negative number!\n");
19.     }
20.
21. ?>
```

```
1. <?php
2.
3.     /**
4.      * froshims-0.php
5.      *
6.      * David J. Malan
7.      * malan@harvard.edu
8.      *
9.      * Implements a registration form for Frosh IMs.
10.     * Submits to register-0.php.
11.     */
12.
13. ?>
14.
15. <!DOCTYPE html>
16.
17. <html>
18.     <head>
19.         <title>Frosh IMs</title>
20.     </head>
21.     <body style="text-align: center;">
22.         <h1>Register for Frosh IMs</h1>
23.         <form action="register-0.php" method="post">
24.             Name: <input name="name" type="text" />
25.             <br />
26.             <input name="captain" type="checkbox" /> Captain?
27.             <br />
28.             <input name="comfort" type="radio" value="less" /> Less Comfortable
29.             <input name="comfort" type="radio" value="more" /> More Comfortable
30.             <br />
31.             Dorm:
32.             <select name="dorm">
33.                 <option value=""></option>
34.                 <option value="Apley Court">Apley Court</option>
35.                 <option value="Canaday">Canaday</option>
36.                 <option value="Grays">Grays</option>
37.                 <option value="Greenough">Greenough</option>
38.                 <option value="Hollis">Hollis</option>
39.                 <option value="Holworthy">Holworthy</option>
40.                 <option value="Hurlbut">Hurlbut</option>
41.                 <option value="Lionel">Lionel</option>
42.                 <option value="Matthews">Matthews</option>
43.                 <option value="Mower">Mower</option>
44.                 <option value="Pennypacker">Pennypacker</option>
45.                 <option value="Stoughton">Stoughton</option>
46.                 <option value="Straus">Straus</option>
47.                 <option value="Thayer">Thayer</option>
48.                 <option value="Weld">Weld</option>
```

```
49.         <option value="Wigglesworth">Wigglesworth</option>
50.     </select>
51.     <br/>
52.     <input type="submit" value="Register"/>
53. </form>
54. </body>
55. </html>
```

```
1. <?php
2.
3.  /**
4.   * froshims-1.php
5.   *
6.   * David J. Malan
7.   * malan@harvard.edu
8.   *
9.   * Implements a registration form for Frosh IMs.
10.  * Submits to register-1.php.
11.  *
12.  * Demonstrates Bootstrap (http://getbootstrap.com/).
13.  */
14.
15. ?>
16.
17. <!DOCTYPE html>
18.
19. <html>
20.   <head>
21.     <link href="bootstrap/css/bootstrap.min.css" rel="stylesheet"/>
22.     <title>Frosh IMs</title>
23.   </head>
24.   <body style="margin: 20px;">
25.     <h1>Register for Frosh IMs</h1>
26.     <form action="register-1.php" method="post">
27.       <fieldset>
28.         <label>Name</label>
29.         <input name="name" type="text"/>
30.         <label class="checkbox">
31.           <input name="captain" type="checkbox"/> Captain?
32.         </label>
33.         <label class="radio">
34.           <input name="comfort" type="radio" value="less"/> Less Comfortable
35.         </label>
36.         <label class="radio">
37.           <input name="comfort" type="radio" value="more"/> More Comfortable
38.         </label>
39.         <label>
40.           <select name="dorm">
41.             <option value=""></option>
42.             <option value="Apley Court">Apley Court</option>
43.             <option value="Canaday">Canaday</option>
44.             <option value="Grays">Grays</option>
45.             <option value="Greenough">Greenough</option>
46.             <option value="Hollis">Hollis</option>
47.             <option value="Holworthy">Holworthy</option>
48.             <option value="Hurlbut">Hurlbut</option>
```

```
49.         <option value="Lionel">Lionel</option>
50.         <option value="Matthews">Matthews</option>
51.         <option value="Mower">Mower</option>
52.         <option value="Pennypacker">Pennypacker</option>
53.         <option value="Stoughton">Stoughton</option>
54.         <option value="Straus">Straus</option>
55.         <option value="Thayer">Thayer</option>
56.         <option value="Weld">Weld</option>
57.         <option value="Wigglesworth">Wigglesworth</option>
58.     </select>
59. </label>
60.     <button class="btn btn-default" type="submit">Register</button>
61. </fieldset>
62. </form>
63. </body>
64. </html>
```

```
1. <?php
2.
3.     /**
4.      * froshims-2.php
5.      *
6.      * David J. Malan
7.      * malan@harvard.edu
8.      *
9.      * Implements a registration form for Frosh IMs.
10.     * Submits to register-2.php.
11.     */
12.
13. ?>
14.
15. <!DOCTYPE html>
16.
17. <html>
18.     <head>
19.         <title>Frosh IMs</title>
20.     </head>
21.     <body style="text-align: center;">
22.         <h1>Register for Frosh IMs</h1>
23.         <form action="register-2.php" method="post">
24.             Name: <input name="name" type="text" />
25.             <br />
26.             <input name="captain" type="checkbox" /> Captain?
27.             <br />
28.             <input name="comfort" type="radio" value="less" /> Less Comfortable
29.             <input name="comfort" type="radio" value="more" /> More Comfortable
30.             <br />
31.             Dorm:
32.             <select name="dorm">
33.                 <option value=""></option>
34.                 <option value="Apley Court">Apley Court</option>
35.                 <option value="Canaday">Canaday</option>
36.                 <option value="Grays">Grays</option>
37.                 <option value="Greenough">Greenough</option>
38.                 <option value="Hollis">Hollis</option>
39.                 <option value="Holworthy">Holworthy</option>
40.                 <option value="Hurlbut">Hurlbut</option>
41.                 <option value="Lionel">Lionel</option>
42.                 <option value="Matthews">Matthews</option>
43.                 <option value="Mower">Mower</option>
44.                 <option value="Pennypacker">Pennypacker</option>
45.                 <option value="Stoughton">Stoughton</option>
46.                 <option value="Straus">Straus</option>
47.                 <option value="Thayer">Thayer</option>
48.                 <option value="Weld">Weld</option>
```

```
49.         <option value="Wigglesworth">Wigglesworth</option>
50.     </select>
51.     <br/>
52.     <input type="submit" value="Register"/>
53. </form>
54. </body>
55. </html>
```

```
1. <?php
2.
3.  /**
4.   * froshims-3.php
5.   *
6.   * David J. Malan
7.   * malan@harvard.edu
8.   *
9.   * Implements a registration form for Frosh IMs.
10.  * Submits to register-3.php.
11.  */
12.
13. ?>
14.
15. <!DOCTYPE html>
16.
17. <html>
18.   <head>
19.     <title>Frosh IMs</title>
20.   </head>
21.   <body style="text-align: center;">
22.     <h1>Register for Frosh IMs</h1>
23.     <form action="register-3.php" method="post">
24.       Name: <input name="name" type="text" />
25.       <br />
26.       <input name="captain" type="checkbox" /> Captain?
27.       <br />
28.       <input name="comfort" type="radio" value="less" /> Less Comfortable
29.       <input name="comfort" type="radio" value="more" /> More Comfortable
30.       <br />
31.       Dorm:
32.       <select name="dorm">
33.         <option value=""></option>
34.         <option value="Apley Court">Apley Court</option>
35.         <option value="Canaday">Canaday</option>
36.         <option value="Grays">Grays</option>
37.         <option value="Greenough">Greenough</option>
38.         <option value="Hollis">Hollis</option>
39.         <option value="Holworthy">Holworthy</option>
40.         <option value="Hurlbut">Hurlbut</option>
41.         <option value="Lionel">Lionel</option>
42.         <option value="Matthews">Matthews</option>
43.         <option value="Mower">Mower</option>
44.         <option value="Pennypacker">Pennypacker</option>
45.         <option value="Stoughton">Stoughton</option>
46.         <option value="Straus">Straus</option>
47.         <option value="Thayer">Thayer</option>
48.         <option value="Weld">Weld</option>
```



```
49.         <option value="Wigglesworth">Wigglesworth</option>
50.     </select>
51.     <br/>
52.     <input type="submit" value="Register"/>
53. </form>
54. </body>
55. </html>
```

```
1. <?php
2.
3.     /**
4.      * register-0.php
5.      *
6.      * David J. Malan
7.      * malan@harvard.edu
8.      *
9.      * Dumps contents of $_POST.
10.     */
11.
12. ?>
13.
14. <!DOCTYPE html>
15.
16. <html>
17.     <head>
18.         <title>Frosh IMs</title>
19.     </head>
20.     <body>
21.         <pre><?php print_r($_POST); ?></pre>
22.     </body>
23. </html>
```

```
1. <?php
2.
3.     /**
4.      * register-1.php
5.      *
6.      * David J. Malan
7.      * malan@harvard.edu
8.      *
9.      * Implements a registration form for Frosh IMs. Redirects
10.     * user to froshims-1.php upon error.
11.     */
12.
13.     // validate submission
14.     if (empty($_POST["name"]) || empty($_POST["comfort"]) || empty($_POST["dorm"]))
15.     {
16.         header("Location: froshims-1.php");
17.         exit;
18.     }
19.
20. ?>
21.
22. <!DOCTYPE html>
23.
24. <html>
25.     <head>
26.         <title>Frosh IMs</title>
27.     </head>
28.     <body>
29.         You are registered! (Well, not really.)
30.     </body>
31. </html>
```

```
1. <?php
2.
3.     /**
4.      * register-2.php
5.      *
6.      * Computer Science 50
7.      * David J. Malan
8.      *
9.      * Implements a registration form for Frosh IMs.  Informs user of
10.     * any errors.
11.     */
12.
13. ?>
14.
15. <!DOCTYPE html>
16.
17. <html>
18.     <head>
19.         <title>Frosh IMs</title>
20.     </head>
21.     <body>
22.         <?php if (empty($_POST["name"]) || empty($_POST["comfort"]) || empty($_POST["dorm"])): ?>
23.             You must provide your name, comfort, and dorm!  Go <a href="froshims-2.php">back</a>.
24.         <?php else: ?>
25.             You are registered!  (Well, not really.)
26.         <?php endif ?>
27.     </body>
28. </html>
```

```
1. <?php
2.
3.  /**
4.   * register-3.php
5.   *
6.   * Computer Science 50
7.   * David J. Malan
8.   *
9.   * Implements a registration form for Frosh IMS. Reports registration
10.  * via email. Redirects user to froshims-3.php upon error.
11.  */
12.
13. // require PHPMailer
14. require("libphp-phpmailer/class.phpmailer.php");
15.
16. // validate submission
17. if (!empty($_POST["name"]) && !empty($_POST["comfort"]) && !empty($_POST["dorm"]))
18. {
19.     // instantiate mailer
20.     $mail = new PHPMailer();
21.
22.     // use SMTP
23.     $mail->IsSMTP();
24.     $mail->Host = "smtp.gmail.com";
25.     $mail->Password = "crimson50";
26.     $mail->Port = 587;
27.     $mail->SMTPAuth = true;
28.     $mail->SMTPSecure = "tls";
29.     $mail->Username = "jharvard@cs50.net";
30.
31.     // set From:
32.     $mail->SetFrom("jharvard@cs50.net");
33.
34.     // set To:
35.     $mail->AddAddress("jharvard@cs50.net");
36.
37.     // set Subject:
38.     $mail->Subject = "registration";
39.
40.     // set body
41.     $mail->Body =
42.         "This person just registered:\n\n" .
43.         "Name: " . $_POST["name"] . "\n" .
44.         "Captain: " . $_POST["captain"] . "\n" .
45.         "Comfort: " . $_POST["comfort"] . "\n" .
46.         "Dorm: " . $_POST["dorm"];
47.
48.     // send mail
```

```
49.     if ($mail->Send() == false)
50.     {
51.         print($mail->ErrInfo);
52.         exit;
53.     }
54. }
55. else
56. {
57.     header("Location: froshims-3.php");
58.     exit;
59. }
60. ?>
61.
62. <!DOCTYPE html>
63.
64. <html>
65.     <head>
66.         <title>Frosh IMs</title>
67.     </head>
68.     <body>
69.         You are registered! (Really.)
70.     </body>
71. </html>
```

```
1. #!/usr/bin/env php
2. <?php
3.
4.     printf("hello, world\n");
5.
6. ?>
```

```
1. <?php
2.
3.  /**
4.   * dictionary.php
5.   *
6.   * David J. Malan
7.   * malan@harvard.edu
8.   *
9.   * Implements a dictionary in a (non-object-oriented) way that
10.  * mimics Problem Set 6's implementation in C. However, an
11.  * object-oriented design would be better in PHP.
12.  */
13.
14. // size of dictionary
15. $size = 0;
16.
17. // hash table
18. $table = [];
19.
20. /**
21.  * Returns true if word is in dictionary else false.
22.  */
23. function check($word)
24. {
25.     global $table;
26.     if (isset($table[strtolower($word)]))
27.     {
28.         return true;
29.     }
30.     else
31.     {
32.         return false;
33.     }
34. }
35.
36. /**
37.  * Loads dictionary into memory. Returns true if successful else false.
38.  */
39. function load($dictionary)
40. {
41.     global $table, $size;
42.     if (!file_exists($dictionary) && is_readable($dictionary))
43.     {
44.         return false;
45.     }
46.     foreach (file($dictionary) as $word)
47.     {
48.         $table[chop($word)] = true;
```



```
49.         $size++;
50.     }
51.     return true;
52. }
53.
54. /**
55.  * Returns number of words in dictionary if loaded else 0 if not yet loaded.
56.  */
57. function size()
58. {
59.     global $size;
60.     return $size;
61. }
62.
63. /**
64.  * Unloads dictionary from memory. Returns true if successful else false.
65.  */
66. function unload()
67. {
68.     return true;
69. }
70.
71. ?>
```

```
1. #!/usr/bin/env php
2. <?php
3.
4. /*****
5.  * speller.php
6.  *
7.  * David J. Malan
8.  * malan@harvard.edu
9.  *
10. * Implements a spell-checker.
11. *****/
12.
13. require("dictionary.php");
14.
15. // maximum length for a word
16. // (e.g., pneumonoultramicroscopicsilicovolcanoconiosis)
17. define("LENGTH", 45);
18.
19. // default dictionary
20. define("WORDS", "/home/cs50/pset5/dictionaries/large");
21.
22. // check for correct number of args
23. if ($argc != 2 && $argc != 3)
24. {
25.     print("Usage: speller [dictionary] text\n");
26.     return 1;
27. }
28.
29. // benchmarks
30. $time_load = 0.0; $time_check = 0.0; $time_size = 0.0; $time_unload = 0.0;
31.
32. // determine dictionary to use
33. $dictionary = ($argc == 3) ? $argv[1] : WORDS;
34.
35. // load dictionary
36. $before = microtime(true);
37. $loaded = load($dictionary);
38. $after = microtime(true);
39.
40. // abort if dictionary not loaded
41. if (!$loaded)
42. {
43.     print("Could not load $dictionary.\n");
44.     return 1;
45. }
46.
47. // calculate time to load dictionary
48. $time_load = $after - $before;
```

```
49.
50. // try to open file
51. $file = ($argc == 3) ? $argv[2] : $argv[1];
52. $fp = fopen($file, "r");
53. if ($fp === false)
54. {
55.     print("Could not open $file.\n");
56.     return 1;
57. }
58.
59. // prepare to report misspellings
60. printf("\nMISSPELLED WORDS\n\n");
61.
62. // prepare to spell-check
63. $word = "";
64. $index = 0; $mispellings = 0; $words = 0;
65.
66. // spell-check each word in file
67. for ($c = fgetc($fp); $c !== false; $c = fgetc($fp))
68. {
69.     // allow alphabetical characters and apostrophes (for possessives)
70.     if (preg_match("/[a-zA-Z]/", $c) || ($c == "'" && $index > 0))
71.     {
72.         // append character to word
73.         $word .= $c;
74.         $index++;
75.
76.         // ignore alphabetical strings too long to be words
77.         if ($index >= LENGTH)
78.         {
79.             // consume remainder of alphabetical string
80.             while (($c = fgetc($fp)) !== false && preg_match("/[a-zA-Z]/", $c));
81.
82.             // prepare for new word
83.             $index = 0; $word = "";
84.         }
85.     }
86.
87.     // ignore words with numbers (like MS Word)
88.     else if (ctype_digit($c))
89.     {
90.         // consume remainder of alphabetical string
91.         while (($c = fgetc($fp)) !== false && preg_match("/[a-zA-z0-9]/", $c));
92.
93.         // prepare for new word
94.         $index = 0; $word = "";
95.     }
96.
```

```
97.         // we must have found a whole word
98.     else if ($index > 0)
99.     {
100.         // update counter
101.         $words++;
102.
103.         // check word's spelling
104.         $before = microtime(true);
105.         $misspelled = !check($word);
106.         $after = microtime(true);
107.
108.         // update benchmark
109.         $time_check += $after - $before;
110.
111.         // print word if misspelled
112.         if ($misspelled)
113.         {
114.             print("$word\n");
115.             $mispellings++;
116.         }
117.
118.         // prepare for next word
119.         $index = 0; $word = "";
120.     }
121. }
122.
123. // close file
124. fclose($fp);
125.
126. // determine dictionary's size
127. $before = microtime(true);
128. $n = size();
129. $after = microtime(true);
130.
131. // calculate time to determine dictionary's size
132. $time_size = $after - $before;
133.
134. // unload dictionary
135. $before = microtime(true);
136. $unloaded = unload();
137. $after = microtime(true);
138.
139. // abort if dictionary not unloaded
140. if (!$unloaded)
141. {
142.     print("Could not load $dictionary.\n");
143.     return 1;
144. }
```

```
145. // calculate time to determine dictionary's size
146. $time_unload = $after - $before;
147.
148. // report benchmarks
149. printf("\nWORDS MISPELLED:      %d\n", $mispellings);
150. printf("WORDS IN DICTIONARY:  %d\n", $n);
151. printf("WORDS IN TEXT:        %d\n", $words);
152. printf("TIME IN load:         %.2f\n", $time_load);
153. printf("TIME IN check:        %.2f\n", $time_check);
154. printf("TIME IN size:         %.2f\n", $time_size);
155. printf("TIME IN unload:       %.2f\n", $time_unload);
156. printf("TOTAL TIME:          %.2f\n\n", $time_load + $time_check + $time_size + $time_unload);
157.
158. ?>
```

```
1. #!/usr/bin/env php
2. <?php
3.
4.     $x = 2;
5.     printf("x is now %d\n", $x);
6.     printf("Cubing...\n");
7.     $x = cube($x);
8.     printf("Cubed!\n");
9.     printf("x is now %d\n", $x);
10.
11.     /**
12.      * Cubes argument.
13.      */
14.     function cube($n)
15.     {
16.         return $n * $n * $n;
17.     }
18.
19. ?>
```

```
1. <?php
2.
3.     require("libphp-phpmailer/class.phpmailer.php");
4.
5.     // open file
6.     $handle = fopen("TODO", "r");
7.     if ($handle == false)
8.     {
9.         print("could not open file\n");
10.        exit(1);
11.    }
12.
13.    // array to hold email addresses
14.    $addresses = [];
15.
16.    // iterate over rows in file
17.    while ($row = fgetcsv($handle))
18.    {
19.        // name is in first column
20.        $name = $row[0];
21.
22.        // number is in third column
23.        $number = $row[2];
24.
25.        // carrier is in fourth column
26.        $carrier = $row[3];
27.
28.        // determine address
29.        if ($carrier == "AT&T")
30.        {
31.            array_push($addresses, "{$number}@txt.att.net");
32.        }
33.        else if ($carrier == "Sprint")
34.        {
35.            array_push($addresses, "{$number}@messaging.sprintpcs.com");
36.        }
37.        else if ($carrier == "T-Mobile")
38.        {
39.            array_push($addresses, "{$number}@tmomail.net");
40.        }
41.        else if ($carrier == "Verizon")
42.        {
43.            array_push($addresses, "{$number}@vtext.com");
44.        }
45.        else if ($carrier == "Virgin Mobile")
46.        {
47.            array_push($addresses, "{$number}@vmobl.com");
48.        }

```

```
49.     }
50.
51.     // close file
52.     fclose($handle);
53.
54.     // instantiate mailer
55.     $mail = new PHPMailer();
56.
57.     // configure mailer
58.     // http://phpmailer.worxware.com/index.php?pg=methods
59.     // http://phpmailer.worxware.com/index.php?pg=properties
60.     // https://www.google.com/settings/u/0/security/lesssecureapps
61.     $mail->IsSMTP();
62.     $mail->Host = "smtp.gmail.com";
63.     $mail->Password = "TODO";
64.     $mail->Port = 587;
65.     $mail->SMTPAuth = true;
66.     $mail->SMTPDebug = 1;
67.     $mail->SMTPSecure = "tls";
68.     $mail->Username = "TODO";
69.
70.     // set From:
71.     $mail->SetFrom("bot@cs50.net");
72.
73.     // set body
74.     $mail->Body = "Miss you! love, CS50 Bot";
75.
76.     // iterate over email addresses
77.     for ($i = 0, $n = count($addresses); $i < $n; $i++)
78.     {
79.         // add email address to To: field
80.         $mail->addAddress($addresses[$i]);
81.
82.         // send email
83.         if ($mail->Send())
84.         {
85.             print("Sent text #{$i}.\n");
86.         }
87.         else
88.         {
89.             print($mail->ErrorInfo);
90.         }
91.
92.         // clear To: field
93.         $mail->ClearAddresses();
94.     }
95.
96.     ?>
```