

```
1: <!--
2:
3: ajax1.html
4:
5: Gets stock quote from quotel.php via Ajax, displaying result with alert().
6:
7: Computer Science 50
8: David J. Malan
9:
10: -->
11:
12: <!DOCTYPE html>
13:
14: <html>
15:   <head>
16:     <script type="text/javascript">
17:       // <![CDATA[
18:
19:         // an XMLHttpRequest
20:         var xhr = null;
21:
22:         /*
23:          * void
24:          * quote()
25:          *
26:          * Gets a quote.
27:          */
28:         function quote()
29:         {
30:           // instantiate XMLHttpRequest object
31:           try
32:           {
33:             xhr = new XMLHttpRequest();
34:           }
35:           catch (e)
36:           {
37:             xhr = new ActiveXObject("Microsoft.XMLHTTP");
38:           }
39:
40:           // handle old browsers
41:           if (xhr == null)
42:           {
43:             alert("Ajax not supported by your browser!");
44:             return;
45:           }

```

```
46:
47:         // construct URL
48:         var url = "quotel.php?symbol=" + document.getElementById("symbol").value;
49:
50:         // get quote
51:         xhr.onreadystatechange = handler;
52:         xhr.open("GET", url, true);
53:         xhr.send(null);
54:       }
55:
56:
57:       /*
58:        * void
59:        * handler()
60:        *
61:        * Handles the Ajax response.
62:        */
63:       function handler()
64:       {
65:         // only handle loaded requests
66:         if (xhr.readyState == 4)
67:         {
68:           // display response if possible
69:           if (xhr.status == 200)
70:             alert(xhr.responseText);
71:           else
72:             alert("Error with Ajax call!");
73:         }
74:       }
75:
76:     // ]]>
77:   </script>
78:   <title>ajax1</title>
79: </head>
80: <body>
81:   <form onsubmit="quote(); return false;">
82:     Symbol: <input id="symbol" type="text">
83:     <br><br>
84:     <input type="submit" value="Get Quote">
85:   </form>
86: </body>
87: </html>

```

```
1: <!--
2:
3: ajax2.html
4:
5: Gets stock quote from quotel.php via Ajax, embedding result in page itself.
6:
7: Computer Science 50
8: David J. Malan
9:
10: -->
11:
12: <!DOCTYPE html>
13:
14: <html>
15:   <head>
16:     <script type="text/javascript">
17:       // <![CDATA[
18:
19:         // an XMLHttpRequest
20:         var xhr = null;
21:
22:         /*
23:          * void
24:          * quote()
25:          *
26:          * Gets a quote.
27:          */
28:         function quote()
29:         {
30:           // instantiate XMLHttpRequest object
31:           try
32:           {
33:             xhr = new XMLHttpRequest();
34:           }
35:           catch (e)
36:           {
37:             xhr = new ActiveXObject("Microsoft.XMLHTTP");
38:           }
39:
40:           // handle old browsers
41:           if (xhr == null)
42:           {
43:             alert("Ajax not supported by your browser!");
44:             return;
45:           }

```

```
46:
47:         // construct URL
48:         var url = "quotel.php?symbol=" + document.getElementById("symbol").value;
49:
50:         // get quote
51:         xhr.onreadystatechange = handler;
52:         xhr.open("GET", url, true);
53:         xhr.send(null);
54:       }
55:
56:
57:         /*
58:          * void
59:          * handler()
60:          *
61:          * Handles the Ajax response.
62:          */
63:         function handler()
64:         {
65:           // only handle loaded requests
66:           if (xhr.readyState == 4)
67:           {
68:             // embed response in page if possible
69:             if (xhr.status == 200)
70:               document.getElementById("price").innerHTML = xhr.responseText;
71:             else
72:               alert("Error with Ajax call!");
73:           }
74:         }
75:
76:       // ]]>
77:     </script>
78:     <title>ajax2</title>
79:   </head>
80:   <body>
81:     <form onsubmit="quote(); return false;">
82:       Symbol: <input id="symbol" type="text">
83:       <br>
84:       Price: <span id="price"><b>to be determined</b></span>
85:       <br><br>
86:       <input type="submit" value="Get Quote">
87:     </form>
88:   </body>
89: </html>

```

```
1: <!--
2:
3: ajax3.html
4:
5: Gets stock quote (plus day's low and high) from quote2.php via Ajax,
6: embedding result in page itself after indicating progress with an
7: animated GIF.
8:
9: Computer Science 50
10: David J. Malan
11:
12: -->
13:
14: <!DOCTYPE html>
15:
16: <html>
17:   <head>
18:     <script type="text/javascript">
19:       // <![CDATA[
20:
21:         // an XMLHttpRequest
22:         var xhr = null;
23:
24:         /*
25:          * void
26:          * quote()
27:          *
28:          * Gets a quote.
29:          */
30:         function quote()
31:         {
32:           // instantiate XMLHttpRequest object
33:           try
34:           {
35:             xhr = new XMLHttpRequest();
36:           }
37:           catch (e)
38:           {
39:             xhr = new ActiveXObject("Microsoft.XMLHTTP");
40:           }
41:
42:           // handle old browsers
43:           if (xhr == null)
44:           {
45:             alert("Ajax not supported by your browser!");
```

```
46:             return;
47:           }
48:
49:           // construct URL
50:           var url = "quote2.php?symbol=" + document.getElementById("symbol").value;
51:
52:           // show progress
53:           document.getElementById("progress").style.display = "block";
54:
55:           // get quote
56:           xhr.onreadystatechange = handler;
57:           xhr.open("GET", url, true);
58:           xhr.send(null);
59:         }
60:
61:
62:         /*
63:          * void
64:          * handler()
65:          *
66:          * Handles the Ajax response.
67:          */
68:         function handler()
69:         {
70:           // only handle requests in "loaded" state
71:           if (xhr.readyState == 4)
72:           {
73:             // hide progress
74:             document.getElementById("progress").style.display = "none";
75:
76:             // embed response in page if possible
77:             if (xhr.status == 200)
78:               document.getElementById("quote").innerHTML = xhr.responseText;
79:             else
80:               alert("Error with Ajax call!");
81:           }
82:         }
83:
84:       // ]]>
85:     </script>
86:     <title>ajax3</title>
87:   </head>
88:   <body>
89:     <form onsubmit="quote(); return false;">
90:       Symbol: <input id="symbol" type="text">
```

```
91:     <br><br>
92:     <div id="progress" style="display: none">
93:         
94:         <br><br>
95:     </div>
96:     <div id="quote"></div>
97:     <br><br>
98:     <input type="submit" value="Get Quote">
99: </form>
100: </body>
101: </html>
```

```
1: <!--
2:
3: ajax4.html
4:
5: Gets stock quote from quotel.php via Ajax, displaying result with alert().
6: Implements handler as an anonymous function.
7:
8: Computer Science 50
9: David J. Malan
10:
11: -->
12:
13: <!DOCTYPE html>
14:
15: <html>
16:   <head>
17:     <script type="text/javascript">
18:       // <![CDATA[
19:
20:         // an XMLHttpRequest
21:         var xhr = null;
22:
23:         /*
24:          * void
25:          * quote()
26:          *
27:          * Gets a quote.
28:          */
29:         function quote()
30:         {
31:           // instantiate XMLHttpRequest object
32:           try
33:           {
34:             xhr = new XMLHttpRequest();
35:           }
36:           catch (e)
37:           {
38:             xhr = new ActiveXObject("Microsoft.XMLHTTP");
39:           }
40:
41:           // handle old browsers
42:           if (xhr == null)
43:           {
44:             alert("Ajax not supported by your browser!");
45:             return;
```

```
46:         }
47:
48:         // construct URL
49:         var url = "quote1.php?symbol=" + document.getElementById("symbol").value;
50:
51:         // get quote
52:         xhr.onreadystatechange = function () {
53:             // only handle loaded requests
54:             if (xhr.readyState == 4)
55:             {
56:                 // display response if possible
57:                 if (xhr.status == 200)
58:                     alert(xhr.responseText);
59:                 else
60:                     alert("Error with Ajax call!");
61:             }
62:         };
63:         xhr.open("GET", url, true);
64:         xhr.send(null);
65:     }
66:
67:     // ]]>
68: </script>
69: <title>ajax4</title>
70: </head>
71: <body>
72:     <form onsubmit="quote(); return false;">
73:         Symbol: <input id="symbol" type="text">
74:         <br><br>
75:         <input type="submit" value="Get Quote">
76:     </form>
77: </body>
78: </html>
```

```
1: <!--
2:
3: ajax5.html
4:
5: Gets stock quote (plus day's low and high) from quote3.php via Ajax,
6: embedding (JSON) result in page itself.
7:
8: Computer Science 50
9: David J. Malan
10:
11: -->
12:
13: <!DOCTYPE html>
14:
15: <html>
16:     <head>
17:         <script type="text/javascript">
18:             // 
19:
20:                 // an XMLHttpRequest
21:                 var xhr = null;
22:
23:                 /*
24:                 * void
25:                 * quote()
26:                 *
27:                 * Gets a quote.
28:                 */
29:                 function quote()
30:                 {
31:                     // instantiate XMLHttpRequest object
32:                     try
33:                     {
34:                         xhr = new XMLHttpRequest();
35:                     }
36:                     catch (e)
37:                     {
38:                         xhr = new ActiveXObject("Microsoft.XMLHTTP");
39:                     }
40:
41:                     // handle old browsers
42:                     if (xhr == null)
43:                     {
44:                         alert("Ajax not supported by your browser!");
45:                         return;</pre></div>
```

```
46:     }
47:
48:     // construct URL
49:     var url = "quote3.php?symbol=" + document.getElementById("symbol").value;
50:
51:     // get quote
52:     xhr.onreadystatechange = function() {
53:
54:         // only handle requests in "loaded" state
55:         if (xhr.readyState == 4)
56:         {
57:             // embed response in page if possible
58:             if (xhr.status == 200)
59:             {
60:                 var quote = eval("(" + xhr.responseText + ")");
61:                 document.getElementById("price").innerHTML = quote.price;
62:                 document.getElementById("high").innerHTML = quote.high;
63:                 document.getElementById("low").innerHTML = quote.low;
64:             }
65:             else
66:                 alert("Error with Ajax call!");
67:         }
68:
69:     };
70:     xhr.open("GET", url, true);
71:     xhr.send(null);
72: }
73:
74:
75: /*
76:  * void
77:  * handler()
78:  *
79:  * Handles the Ajax response.
80:  */
81: function handler()
82: {
83: }
84:
85: // ]]>
86: </script>
87: <title>ajax5</title>
88: </head>
89: <body>
90: <form onsubmit="quote(); return false;">
```

```
91:     Symbol: <input id="symbol" type="text">
92:     <br><br>
93:     Price: <span id="price"></span>
94:     <br>
95:     High: <span id="high"></span>
96:     <br>
97:     Low: <span id="low"></span>
98:     <br>
99:     <br><br>
100:     <input type="submit" value="Get Quote">
101:     </form>
102: </body>
103: </html>
```

```
1: <?
2:
3:  /**
4:   * quote1.php
5:   *
6:   * Outputs price of given symbol as text/html.
7:   *
8:   * Computer Science 50
9:   * David J. Malan
10:  */
11:
12:  // get quote
13:  $handle = @fopen("http://download.finance.yahoo.com/d/quotes.csv?s={$_GET["symbol"]}&f=e1l1", "r");
14:  if ($handle !== FALSE)
15:  {
16:      $data = fgetcsv($handle);
17:      if ($data !== FALSE && $data[0] == "N/A")
18:          print($data[1]);
19:      fclose($handle);
20:  }
21: ?>
```

```
1: <?
2:
3:  /**
4:   * quote2.php
5:   *
6:   * Outputs price, low, and high of given symbol as text/html, after
7:   * inserting an artificial delay.
8:   *
9:   * Computer Science 50
10:  * David J. Malan
11:  */
12:
13:
14:  // pretend server is slow
15:  sleep(5);
16:
17:  // try to get quote
18:  $handle = @fopen("http://download.finance.yahoo.com/d/quotes.csv?s={$_GET["symbol"]}&f=e1l1hg", "r");
19:  if ($handle !== FALSE)
20:  {
21:      $data = fgetcsv($handle);
22:      if ($data !== FALSE && $data[0] == "N/A")
23:      {
24:          print("Price: {$data[1]}");
25:          print("<br>");
26:          print("High: {$data[2]}");
27:          print("<br>");
28:          print("Low: {$data[3]}");
29:      }
30:      fclose($handle);
31:  }
32: ?>
```

```
1: <?
2:
3:  /**
4:   * quote3.php
5:   *
6:   * Outputs price, low, and high of given symbol as JSON.
7:   *
8:   * Computer Science 50
9:   * David J. Malan
10:  */
11:
12:  // try to get quote
13:  $quote = array();
14:  $handle = @fopen("http://download.finance.yahoo.com/d/quotes.csv?s={$_GET["symbol"]}&f=e1l1hg", "r");
15:  if ($handle !== FALSE)
16:  {
17:      $data = fgetcsv($handle);
18:      if ($data !== FALSE && $data[0] == "N/A")
19:      {
20:          $quote["price"] = $data[1];
21:          $quote["high"] = $data[2];
22:          $quote["low"] = $data[3];
23:      }
24:      fclose($handle);
25:  }
26:  header("Content-type: application/json");
27:  print(json_encode($quote));
28: ?>
```

```
1: <?
2:
3:  /**
4:   * dump.php
5:   *
6:   * Dumps HTTP requests.
7:   *
8:   * Computer Science 50
9:   * David J. Malan
10:  */
11:
12: ?>
13:
14: <!DOCTYPE html>
15:
16: <html>
17:   <head>
18:     <title>dump</title>
19:   </head>
20:   <body>
21:     <pre><? print_r($_GET); ?></pre>
22:   </body>
23: </html>
```

```
1: <!--
2:
3: form1.html
4:
5: A form without client-side validation.
6:
7: Computer Science 50
8: David J. Malan
9:
10: -->
11:
12: <!DOCTYPE html>
13:
14: <html>
15:   <head>
16:     <title>form1</title>
17:   </head>
18:   <body>
19:     <form action="dump.php" method="get">
20:       Email: <input name="email" type="text">
21:       <br>
22:       Password: <input name="password1" type="password">
23:       <br>
24:       Password (again): <input name="password2" type="password">
25:       <br>
26:       I agree to the terms and conditions: <input name="agreement" type="checkbox">
27:       <br><br>
28:       <input type="submit" value="Submit">
29:     </form>
30:   </body>
31: </html>
```

```
1: <!--
2:
3: form2.html
4:
5: A form with client-side validation.
6:
7: Computer Science 50
8: David J. Malan
9:
10: -->
11:
12: <!DOCTYPE html>
13:
14: <html>
15:   <head>
16:     <script type="text/javascript">
17:       // <![CDATA[
18:
19:         function validate()
20:         {
21:           if (document.forms.registration.email.value == "")
22:           {
23:             alert("You must provide an email address.");
24:             return false;
25:           }
26:           else if (document.forms.registration.password1.value == "")
27:           {
28:             alert("You must provide a password.");
29:             return false;
30:           }
31:           else if (document.forms.registration.password1.value != document.forms.registration.password2.value)
32:           {
33:             alert("You must provide the same password twice.");
34:             return false;
35:           }
36:           else if (!document.forms.registration.agreement.checked)
37:           {
38:             alert("You must agree to our terms and conditions.");
39:             return false;
40:           }
41:           return true;
42:         }
43:
44:       // ]]>
45:     </script>
```

```
46: <title>form2</title>
47: </head>
48: <body>
49: <form action="dump.php" method="get" name="registration" onsubmit="return validate();">
50:   Email: <input name="email" type="text">
51:   <br>
52:   Password: <input name="password1" type="password">
53:   <br>
54:   Password (again): <input name="password2" type="password">
55:   <br>
56:   I agree to the terms and conditions: <input name="agreement" type="checkbox">
57:   <br><br>
58:   <input type="submit" value="Submit">
59: </form>
60: </body>
61: </html>
```

```
1: <!--
2:
3: form3.html
4:
5: A form with client-side validation demonstrating "this" keyword.
6:
7: Computer Science 50
8: David J. Malan
9:
10: -->
11:
12: <!DOCTYPE html>
13:
14: <html>
15:   <head>
16:     <script type="text/javascript">
17:       // <![CDATA[
18:
19:         function validate(f)
20:         {
21:           if (f.email.value == "")
22:           {
23:             alert("You must provide an email address.");
24:             return false;
25:           }
26:           else if (f.password1.value == "")
27:           {
28:             alert("You must provide a password.");
29:             return false;
30:           }
31:           else if (f.password1.value != f.password2.value)
32:           {
33:             alert("You must provide the same password twice.");
34:             return false;
35:           }
36:           else if (!f.agreement.checked)
37:           {
38:             alert("You must agree to our terms and conditions.");
39:             return false;
40:           }
41:           return true;
42:         }
43:
44:       // ]]>
45:     </script>
```

```
46: <title>form3</title>
47: </head>
48: <body>
49: <form action="dump.php" method="get" onsubmit="return validate(this);">
50:   Email: <input name="email" type="text">
51:   <br>
52:   Password: <input name="password1" type="password">
53:   <br>
54:   Password (again): <input name="password2" type="password">
55:   <br>
56:   I agree to the terms and conditions: <input name="agreement" type="checkbox">
57:   <br><br>
58:   <input type="submit" value="Submit">
59: </form>
60: </body>
61: </html>
```

```
1: <!--
2:
3: form4.html
4:
5: A form with client-side validation demonstrating disabled property.
6:
7: Computer Science 50
8: David J. Malan
9:
10: -->
11:
12: <!DOCTYPE html>
13:
14: <html>
15:   <head>
16:     <script type="text/javascript">
17:       // <![CDATA[
18:
19:         function toggle()
20:         {
21:           if (document.forms.registration.button.disabled)
22:             document.forms.registration.button.disabled = false;
23:           else
24:             document.forms.registration.button.disabled = true;
25:         }
26:
27:         function validate()
28:         {
29:           if (document.forms.registration.email.value == "")
30:           {
31:             alert("You must provide an email address.");
32:             return false;
33:           }
34:           else if (document.forms.registration.password1.value == "")
35:           {
36:             alert("You must provide a password.");
37:             return false;
38:           }
39:           else if (document.forms.registration.password1.value != document.forms.registration.password2.value)
40:           {
41:             alert("You must provide the same password twice.");
42:             return false;
43:           }
44:           else if (!document.forms.registration.agreement.checked)
45:           {
```

```
46:         alert("You must agree to our terms and conditions.");
47:         return false;
48:     }
49:     return true;
50: }
51:
52: // ]]>
53: </script>
54: <title>form4</title>
55: </head>
56: <body>
57:     <form action="dump.php" method="get" name="registration" onsubmit="return validate();" >
58:         Email: <input name="email" type="text">
59:         <br>
60:         Password: <input name="password1" type="password">
61:         <br>
62:         Password (again): <input name="password2" type="password">
63:         <br>
64:         I agree to the terms and conditions: <input name="agreement" onclick="toggle();" type="checkbox">
65:         <br><br>
66:         <input disabled="disabled" name="button" type="submit" value="Submit">
67:     </form>
68: </body>
69: </html>
```

```
1: <!--
2:
3: form5.html
4:
5: A form with client-side validation demonstrating regular expressions.
6:
7: Computer Science 50
8: David J. Malan
9:
10: -->
11:
12: <!DOCTYPE html>
13:
14: <html>
15:   <head>
16:     <script type="text/javascript">
17:       // <![CDATA[
18:
19:         function validate()
20:         {
21:           if (!document.forms.registration.email.value.match(/.+@.+\.edu$/))
22:           {
23:             alert("You must provide a .edu email address.");
24:             return false;
25:           }
26:           else if (document.forms.registration.password1.value == "")
27:           {
28:             alert("You must provide a password.");
29:             return false;
30:           }
31:           else if (document.forms.registration.password1.value != document.forms.registration.password2.value)
32:           {
33:             alert("You must provide the same password twice.");
34:             return false;
35:           }
36:           else if (!document.forms.registration.agreement.checked)
37:           {
38:             alert("You must agree to our terms and conditions.");
39:             return false;
40:           }
41:           return true;
42:         }
43:
44:       // ]]>
45:     </script>
```

```
46: <title>form5</title>
47: </head>
48: <body>
49: <form action="dump.php" method="get" name="registration" onsubmit="return validate();">
50:   Email: <input name="email" type="text">
51:   <br>
52:   Password: <input name="password1" type="password">
53:   <br>
54:   Password (again): <input name="password2" type="password">
55:   <br>
56:   I agree to the terms and conditions: <input name="agreement" type="checkbox">
57:   <br><br>
58:   <input type="submit" value="Submit">
59: </form>
60: </body>
61: </html>
```

```
1: <?
2:
3:   /*****
4:   * dictionary.php
5:   *
6:   * Computer Science 50
7:   * David J. Malan
8:   *
9:   * Implements a dictionary.
10:  *****/
11:
12:
13:   // size of dictionary
14:   $size = 0;
15:
16:   // dictionary
17:   $dictionary = array();
18:
19:
20:   /*
21:   * bool
22:   * check($word)
23:   *
24:   * Returns TRUE if word is in dictionary else FALSE.
25:   */
26:
27:   function check($word)
28:   {
29:     global $dictionary;
30:     if ($dictionary[strtolower($word)])
31:       return TRUE;
32:     else
33:       return FALSE;
34:   }
35:
36:
37:   /*
38:   * bool
39:   * load($dict)
40:   *
41:   * Loads dict into memory. Returns TRUE if successful else FALSE.
42:   */
43:
44:   function load($dict)
45:   {
```

```

46:     global $dictionary, $size;
47:     if (!file_exists($dict) && !is_readable($dict))
48:         return FALSE;
49:     foreach (file($dict) as $word)
50:     {
51:         $dictionary[chop($word)] = TRUE;
52:         $size++;
53:     }
54:     return TRUE;
55: }
56:
57:
58: /*
59:  * int
60:  * size()
61:  *
62:  * Returns number of words in dictionary if loaded else 0 if not yet loaded.
63:  */
64:
65: function size()
66: {
67:     global $size;
68:     return $size;
69: }
70:
71:
72: /*
73:  * int
74:  * unload()
75:  *
76:  * Unloads dictionary from memory. Returns TRUE if successful else FALSE.
77:  */
78:
79: function unload()
80: {
81:     return TRUE;
82: }
83:
84: ?>

```

```

1: #!/usr/bin/php
2: <?
3:
4:     /*****
5:     * speller.php
6:     *
7:     * Computer Science 50
8:     * David J. Malan
9:     *
10:    * Implements a spell-checker.
11:    *****/
12:
13: require("dictionary.php");
14:
15: // suppress notices and warnings
16: error_reporting(E_ALL ^ E_NOTICE);
17:
18:
19: // maximum length for a word
20: // (e.g., pneumoultramicroscopicosilicovolcanoconiosis)
21: define("LENGTH", 45);
22:
23: // default dictionary
24: define("WORDS", "/home/cs50/pub/share/pset6/dict/words");
25:
26: // check for correct number of args
27: if ($argc != 2 && $argc != 3)
28: {
29:     print("Usage: speller.php [dict] file\n");
30:     return 1;
31: }
32:
33: // benchmarks
34: $ti_load = 0.0; $ti_check = 0.0; $ti_size = 0.0; $ti_unload = 0.0;
35:
36: // determine dictionary to use
37: $dict = ($argc == 3) ? $argv[1] : WORDS;
38:
39: // load dictionary
40: $before = microtime(TRUE);
41: $loaded = load($dict);
42: $after = microtime(TRUE);
43:
44: // abort if dictionary not loaded
45: if (!$loaded)

```

```

46:     {
47:         print("Could not load $dict.\n");
48:         return 2;
49:     }
50:
51:     // calculate time to load dictionary
52:     $ti_load = $after - $before;
53:
54:     // try to open file
55:     $file = ($argc == 3) ? $argv[2] : $argv[1];
56:     $fp = fopen($file, "r");
57:     if ($fp === FALSE)
58:     {
59:         print("Could not open $file.\n");
60:         return 3;
61:     }
62:
63:     // prepare to report misspellings
64:     printf("\nMISPELLED WORDS\n\n");
65:
66:     // prepare to spell-check
67:     $word = "";
68:     $index = 0; $misspellings = 0; $words = 0;
69:
70:     // spell-check each word in file
71:     for ($c = fgetc($fp); $c != FALSE; $c = fgetc($fp))
72:     {
73:         // allow alphabetical characters and apostrophes (for possessives)
74:         if (preg_match("/[a-zA-Z]'", $c) || ($c == "'" && $index > 0))
75:         {
76:             // append character to word
77:             $word .= $c;
78:             $index++;
79:
80:             // ignore alphabetical strings too long to be words
81:             if ($index >= LENGTH)
82:             {
83:                 // consume remainder of alphabetical string
84:                 while (($c = fgetc($fp)) != FALSE && preg_match("/[a-zA-Z]'", $c));
85:
86:                 // prepare for new word
87:                 $index = 0; $word = "";
88:             }
89:         }
90:     }

```

```

91:         // ignore words with numbers (like MS Word)
92:         else if (ctype_digit($c))
93:         {
94:             // consume remainder of alphabetical string
95:             while (($c = fgetc($fp)) != FALSE && preg_match("/[a-zA-z0-9]'", $c));
96:
97:             // prepare for new word
98:             $index = 0; $word = "";
99:         }
100:
101:     // we must have found a whole word
102:     else if ($index > 0)
103:     {
104:         // update counter
105:         $words++;
106:
107:         // check word's spelling
108:         $before = microtime(TRUE);
109:         $misspelled = !check($word);
110:         $after = microtime(TRUE);
111:
112:         // update benchmark
113:         $ti_check += $after - $before;
114:
115:         // print word if misspelled
116:         if ($misspelled)
117:         {
118:             print("$word\n");
119:             $misspellings++;
120:         }
121:
122:         // prepare for next word
123:         $index = 0; $word = "";
124:     }
125: }
126:
127: // close file
128: fclose($fp);
129:
130: // determine dictionary's size
131: $before = microtime(TRUE);
132: $n = size();
133: $after = microtime(TRUE);
134:
135: // calculate time to determine dictionary's size

```

```
136:     $ti_size = $after - $before;
137:
138:     // unload dictionary
139:     $before = microtime(TRUE);
140:     $unloaded = unload();
141:     $after = microtime(TRUE);
142:
143:     // abort if dictionary not unloaded
144:     if (!$unloaded)
145:     {
146:         print("Could not load $dict.\n");
147:         return 5;
148:     }
149:     // calculate time to determine dictionary's size
150:     $ti_unload = $after - $before;
151:
152:     // report benchmarks
153:     printf("\nWORDS MISPELLED:      %d\n", $mispellings);
154:     printf("WORDS IN DICTIONARY:  %d\n", $n);
155:     printf("WORDS IN FILE:           %d\n", $words);
156:     printf("TIME IN load:             %.2f\n", $ti_load);
157:     printf("TIME IN check:           %.2f\n", $ti_check);
158:     printf("TIME IN size:            %.2f\n", $ti_size);
159:     printf("TIME IN unload:          %.2f\n", $ti_unload);
160:     printf("TOTAL TIME:              %.2f\n\n", $ti_load + $ti_check + $ti_size + $ti_unload);
161:
162: ?>
```