

Corrections Special School District Career & Technical Education Center

This is file: /StudentResources/CodeSamples/ArrayExample_4b.html

The search string is: {there is no search string}

Term: April 3 to June 16, 2023

This is option **1**, in which the data is ordered by **Room Number**.

Room #	Instructor	Course	Time Slot
1	Lees	Room Closed	7:10 AM – 8:30 AM
1	Lees	Room Closed	8:40 AM – 10:00 AM
1	Lees	Economics (Mon-Thurs)	11:40 AM – 1:00 PM
1	Lees	Civics	1:10 PM – 2:30 PM
2	Burzycki	Biology IA	7:10 AM – 8:30 AM
2	Burzycki	Skills for Success	8:40 AM – 10:00 AM
2	Burzycki	Physical Sciences 1A	11:40 AM – 1:00 PM
2	Burzycki	Personal Nutrition	1:10 PM – 2:30 PM
3	Latorella	American Literature 1A	7:10 AM – 8:30 AM
3	Latorella	British Literature 1A	8:40 AM – 10:00 AM
3	Latorella	Gothic Literature	11:40 AM – 1:00 PM
3	Latorella	Drama as Literature	1:10 PM – 2:30 PM
4		Room Closed	7:10 AM – 8:30 AM
4		Room Closed	8:40 AM – 10:00 AM
4		Room Closed	11:40 AM – 1:00 PM
4		Room Closed	1:10 PM – 2:30 PM
5		Peer-to-Peer HiSET	7:10 AM – 8:30 AM
5		Peer-to-Peer Study Hall	8:40 AM – 10:00 AM
5		Peer-to-Peer HiSET	11:40 AM – 1:00 PM
5		Peer-to-Peer Study Hall	1:10 PM – 2:30 PM
6	O'Donnell	Safety and Sanitation	7:10 AM – 8:30 AM
6	O'Donnell	Culinary Measurements	8:40 AM – 10:00 AM
6	O'Donnell	Safety and Sanitation	11:40 AM – 1:00 PM

6	O'Donnell	Culinary Measurements	1:10 PM – 2:30 PM
7	Piper	Grammar 1B	7:10 AM – 8:30 AM
7	Piper	<i>Room Closed</i>	8:40 AM – 10:00 AM
7	Piper	Poetry	11:40 AM – 1:00 PM
7	Piper	ESL	1:10 PM – 2:30 PM
8	Mosher	Algebra 1A	7:10 AM – 8:30 AM
8	Mosher	Geometry 1A	8:40 AM – 10:00 AM
8	Mosher	<i>Room Closed</i>	11:40 AM – 1:00 PM
8	Mosher	<i>Room Closed</i>	1:10 PM – 2:30 PM
9		<i>Room Closed</i>	7:10 AM – 8:30 AM
9	Heines	Web Design (Tues/Thurs)	8:40 AM – 10:00 AM
9		<i>Room Closed</i>	11:40 AM – 1:00 PM
9		<i>Room Closed</i>	1:10 PM – 2:30 PM
10	Spires	Marketing II	7:10 AM – 8:30 AM
10	Spires	<i>Room Closed</i>	8:40 AM – 10:00 AM
10	Spires	START I & II	11:40 AM – 1:00 PM
10	Spires	Computer Fundamentals	1:10 PM – 2:30 PM

Corrections Special School District Career & Technical Education Center

This is file: /StudentResources/CodeSamples/ArrayExample_4b.html

The search string is: ?SortColumn=2

Term: April 3 to June 16, 2023

This is option **2**, in which the data is ordered by **Instructor Name**.

Room #	Instructor	Course	Time Slot
4		Room Closed	7:10 AM – 8:30 AM
4		Room Closed	8:40 AM – 10:00 AM
4		Room Closed	11:40 AM – 1:00 PM
4		Room Closed	1:10 PM – 2:30 PM
5		Peer-to-Peer HiSET	7:10 AM – 8:30 AM
5		Peer-to-Peer Study Hall	8:40 AM – 10:00 AM
5		Peer-to-Peer HiSET	11:40 AM – 1:00 PM
5		Peer-to-Peer Study Hall	1:10 PM – 2:30 PM
9		Room Closed	7:10 AM – 8:30 AM
9		Room Closed	11:40 AM – 1:00 PM
9		Room Closed	1:10 PM – 2:30 PM
2	Burzycki	Biology IA	7:10 AM – 8:30 AM
2	Burzycki	Skills for Success	8:40 AM – 10:00 AM
2	Burzycki	Physical Sciences 1A	11:40 AM – 1:00 PM
2	Burzycki	Personal Nutrition	1:10 PM – 2:30 PM
9	Heines	Web Design (Tues/Thurs)	8:40 AM – 10:00 AM
3	Latorella	American Literature 1A	7:10 AM – 8:30 AM
3	Latorella	British Literature 1A	8:40 AM – 10:00 AM
3	Latorella	Gothic Literature	11:40 AM – 1:00 PM
3	Latorella	Drama as Literature	1:10 PM – 2:30 PM
1	Lees	Room Closed	7:10 AM – 8:30 AM
1	Lees	Room Closed	8:40 AM – 10:00 AM
1	Lees	Economics (Mon-Thurs)	11:40 AM – 1:00 PM

1	Lees	Civics	1:10 PM – 2:30 PM
8	Mosher	Algebra 1A	7:10 AM – 8:30 AM
8	Mosher	Geometry 1A	8:40 AM – 10:00 AM
8	Mosher	<i>Room Closed</i>	11:40 AM – 1:00 PM
8	Mosher	<i>Room Closed</i>	1:10 PM – 2:30 PM
6	O'Donnell	Safety and Sanitation	7:10 AM – 8:30 AM
6	O'Donnell	Culinary Measurements	8:40 AM – 10:00 AM
6	O'Donnell	Safety and Sanitation	11:40 AM – 1:00 PM
6	O'Donnell	Culinary Measurements	1:10 PM – 2:30 PM
7	Piper	Grammar 1B	7:10 AM – 8:30 AM
7	Piper	<i>Room Closed</i>	8:40 AM – 10:00 AM
7	Piper	Poetry	11:40 AM – 1:00 PM
7	Piper	ESL	1:10 PM – 2:30 PM
10	Spires	Marketing II	7:10 AM – 8:30 AM
10	Spires	<i>Room Closed</i>	8:40 AM – 10:00 AM
10	Spires	START I & II	11:40 AM – 1:00 PM
10	Spires	Computer Fundamentals	1:10 PM – 2:30 PM

Corrections Special School District Career & Technical Education Center

This is file: /StudentResources/CodeSamples/ArrayExample_4b.html

The search string is: ?SortColumn=3

Term: April 3 to June 16, 2023

This is option **3**, in which the data is ordered by **Course Name**.

Room #	Instructor	Course	Time Slot
8	Mosher	Algebra 1A	7:10 AM – 8:30 AM
3	Latorella	American Literature 1A	7:10 AM – 8:30 AM
2	Burzycki	Biology IA	7:10 AM – 8:30 AM
3	Latorella	British Literature 1A	8:40 AM – 10:00 AM
1	Lees	Civics	1:10 PM – 2:30 PM
10	Spires	Computer Fundamentals	1:10 PM – 2:30 PM
6	O'Donnell	Culinary Measurements	8:40 AM – 10:00 AM
6	O'Donnell	Culinary Measurements	1:10 PM – 2:30 PM
3	Latorella	Drama as Literature	1:10 PM – 2:30 PM
7	Piper	ESL	1:10 PM – 2:30 PM
1	Lees	Economics (Mon-Thurs)	11:40 AM – 1:00 PM
8	Mosher	Geometry 1A	8:40 AM – 10:00 AM
3	Latorella	Gothic Literature	11:40 AM – 1:00 PM
7	Piper	Grammar 1B	7:10 AM – 8:30 AM
10	Spires	Marketing II	7:10 AM – 8:30 AM
5		Peer-to-Peer HiSET	7:10 AM – 8:30 AM
5		Peer-to-Peer HiSET	11:40 AM – 1:00 PM
5		Peer-to-Peer Study Hall	8:40 AM – 10:00 AM
5		Peer-to-Peer Study Hall	1:10 PM – 2:30 PM
2	Burzycki	Personal Nutrition	1:10 PM – 2:30 PM
2	Burzycki	Physical Sciences 1A	11:40 AM – 1:00 PM
7	Piper	Poetry	11:40 AM – 1:00 PM
1	Lees	Room Closed	7:10 AM – 8:30 AM

1	Lees	<i>Room Closed</i>	8:40 AM – 10:00 AM
4		<i>Room Closed</i>	7:10 AM – 8:30 AM
4		<i>Room Closed</i>	8:40 AM – 10:00 AM
4		<i>Room Closed</i>	11:40 AM – 1:00 PM
4		<i>Room Closed</i>	1:10 PM – 2:30 PM
7	Piper	<i>Room Closed</i>	8:40 AM – 10:00 AM
8	Mosher	<i>Room Closed</i>	11:40 AM – 1:00 PM
8	Mosher	<i>Room Closed</i>	1:10 PM – 2:30 PM
9		<i>Room Closed</i>	7:10 AM – 8:30 AM
9		<i>Room Closed</i>	11:40 AM – 1:00 PM
9		<i>Room Closed</i>	1:10 PM – 2:30 PM
10	Spires	<i>Room Closed</i>	8:40 AM – 10:00 AM
10	Spires	START I & II	11:40 AM – 1:00 PM
6	O'Donnell	Safety and Sanitation	7:10 AM – 8:30 AM
6	O'Donnell	Safety and Sanitation	11:40 AM – 1:00 PM
2	Burzycki	Skills for Success	8:40 AM – 10:00 AM
9	Heines	Web Design (Tues/Thurs)	8:40 AM – 10:00 AM

Corrections Special School District Career & Technical Education Center

This is file: /StudentResources/CodeSamples/ArrayExample_4b.html

The search string is: ?SortColumn=4

Term: April 3 to June 16, 2023

This is option **4**, in which the data is ordered by **Time Slot**.

Room #	Instructor	Course	Time Slot
1	Lees	Room Closed	7:10 AM – 8:30 AM
2	Burzycki	Biology IA	7:10 AM – 8:30 AM
3	Latorella	American Literature 1A	7:10 AM – 8:30 AM
4		Room Closed	7:10 AM – 8:30 AM
5		Peer-to-Peer HiSET	7:10 AM – 8:30 AM
6	O'Donnell	Safety and Sanitation	7:10 AM – 8:30 AM
7	Piper	Grammar 1B	7:10 AM – 8:30 AM
8	Mosher	Algebra 1A	7:10 AM – 8:30 AM
9		Room Closed	7:10 AM – 8:30 AM
10	Spires	Marketing II	7:10 AM – 8:30 AM
1	Lees	Room Closed	8:40 AM – 10:00 AM
2	Burzycki	Skills for Success	8:40 AM – 10:00 AM
3	Latorella	British Literature 1A	8:40 AM – 10:00 AM
4		Room Closed	8:40 AM – 10:00 AM
5		Peer-to-Peer Study Hall	8:40 AM – 10:00 AM
6	O'Donnell	Culinary Measurements	8:40 AM – 10:00 AM
7	Piper	Room Closed	8:40 AM – 10:00 AM
8	Mosher	Geometry 1A	8:40 AM – 10:00 AM
9	Heines	Web Design (Tues/Thurs)	8:40 AM – 10:00 AM
10	Spires	Room Closed	8:40 AM – 10:00 AM
1	Lees	Economics (Mon-Thurs)	11:40 AM – 1:00 PM
2	Burzycki	Physical Sciences 1A	11:40 AM – 1:00 PM
3	Latorella	Gothic Literature	11:40 AM – 1:00 PM

4		<i>Room Closed</i>	11:40 AM – 1:00 PM
5		Peer-to-Peer HiSET	11:40 AM – 1:00 PM
6	O'Donnell	Safety and Sanitation	11:40 AM – 1:00 PM
7	Piper	Poetry	11:40 AM – 1:00 PM
8	Mosher	<i>Room Closed</i>	11:40 AM – 1:00 PM
9		<i>Room Closed</i>	11:40 AM – 1:00 PM
10	Spires	START I & II	11:40 AM – 1:00 PM
1	Lees	Civics	1:10 PM – 2:30 PM
2	Burzycki	Personal Nutrition	1:10 PM – 2:30 PM
3	Latorella	Drama as Literature	1:10 PM – 2:30 PM
4		<i>Room Closed</i>	1:10 PM – 2:30 PM
5		Peer-to-Peer Study Hall	1:10 PM – 2:30 PM
6	O'Donnell	Culinary Measurements	1:10 PM – 2:30 PM
7	Piper	ESL	1:10 PM – 2:30 PM
8	Mosher	<i>Room Closed</i>	1:10 PM – 2:30 PM
9		<i>Room Closed</i>	1:10 PM – 2:30 PM
10	Spires	Computer Fundamentals	1:10 PM – 2:30 PM

Line wrap

```

1 <!DOCTYPE html>
2 <html>
3
4 <head>
5   <meta charset="utf-8">
6   <title>Array Example 4b</title>
7
8 <!--
9   http://cssdweb.edu/StudentResources/CodeSamples/ArrayExample_4b.html
10  Copyright (c) 2022-2023 by Jesse Heines. All rights reserved. May be freely
11   copied or excerpted for educational purposes with credit to the author.
12   updated by JMH on June 12, 2022 at 8:06 PM
13   updated by JMH on May 29, 2023 at 8:34 AM
14 -->
15
16 <!-- read the stylesheet for this application -->
17 <link rel="stylesheet" href="ArrayExample_4b.css">
18
19 <script>
20   //// INITIALIZE THE ARRAY WITH THE DATA WE WANT TO WORK WITH
21
22   /* This statement declares variable arrCSSD to be an array. It will actually be a
23    * two-dimensional (2D) array by the time we finish populating it with data. A 2D
24    * array is equivalent to a table with rows and column. Each top level element of
25    * the array represents a row, and the data in the secondary array represents the
26    * columns.
27    */
28   var arrCSSD = new Array() ;
29
30   /* The array data is read from an auxiliary JavaScript file so that it can be easily
31    * changed from one semester to another. Putting the data in an auxiliary file
32    * separates the program LOGIC from the DATA that the program works on. This is
33    * analogous to specifying all program layout via CSS to separate program CONTENT from
34    * RENDITION.
35    *
36    * The statements in the auxiliary JavaScript file "push" onto the array, which means
37    * they add data at the end of the array. Each statement pushes an entire subarray
38    * onto the top level array, resulting in a data structure with the 4 "columns". These
39    * columns are, in order:
40    *   | Room # | Instructor Name | Course Name | Time Slot |
41    * where "Day Period" is:
42    *   "A" for 7:10- 8:30
43    *   "B" for 8:40-10:00
44    *   "C" for 10:10-11:30
45    *   "D" for 11:40- 1:00
46    *   "E" for 1:10- 2:30
47    */
48 </script>
49
50 <!-- load the data for this application -->
51 <script src="ArrayExample_4b_2023-04.js"></script>
52
53 <script>
54   //// SET THE DESIRED SORT COLUMN FROM THE URL SEARCH STRING
55
56   // nSortColumn is the number of the column by which the data will be sorted. The
57   // default is 1, but this can be changed by clicking a different column header cell.
58   var nSortColumn = 1 ;
59
60   // The following code determines whether a query (search) string has been supplied
61   // and, if so, whether that query string contains a SortColumn parameter and a legal
62   // value (1-4). If it does, the value of the SortColumn parameter replaces the
63   // nSortColumn value so that the table data is sorted as the user desires.
64   // Note that for simplicity, the following code does not contain a lot of error
65   // checking. In a real application you have to have a lot of error checking whenever

```

```
66 // you get input from the user to ensure that the data he or she enters is valid. In
67 // many of the production quality programs that I have written personally, the amount
68 // of error-checking code EXCEEDS the amount of processing code!
69 // The following code could also be made more efficient, but I have written it our
70 // step-by-step for clarity.
71
72 // (1) get the search (query) string, which is the part of the URL that begins with a
73 // question mark (?)
74 var strSearchString = window.location.search ;
75
76 // (2) if the search string exists, process it, otherwise the following code is skipped
77 // and the default sort column defined above is used
78 if ( strSearchString != "" ) {
79
80     // (3) call the built-in URLSearchParams function to parse the query string into an
81     // data structure of key/value pairs, where the key is the name of the parameter
82     // and the value is the string assigned to that parameter
83     var urlParams = new URLSearchParams( strSearchString ) ;
84
85     // (4) retrieve the value we want: the one associated with key "SortColumn"
86     var strSortColumn = urlParams.get( "SortColumn" ) ;
87
88     // (5) CRITICALLY IMPORTANT! Values extracted from query strings are ALWAYS STRINGS.
89     // For use in our program, we must therefore convert string variable strSortColumn
90     // to a number (variable nSortColumn) using the built-in eval function.
91     nSortColumn = eval( strSortColumn ) ;
92 }
93
94
95 //// COMPARISON FUNCTIONS TO SORT THE TABLE DATA ON THE DESIRED COLUMN
96
97 // Note: The following 4 functions could be combined into a single function with some
98 // additional programming, but I have left them as separate functions for clarity. We
99 // could also write additional functions that sort the specified columns in descending
100 // order.
101
102 // This function compares the first (0th) items in the subarrays, which are the
103 // ROOM NUMBERS. It returns -1 if the 1st item should come before the 2nd. +1 if the
104 // 1st item should come after the 2nd, 0 if their current order is OK.
105 var fnRoomAscending = function( a, b ) {
106     if ( a[0] < b[0] ) {
107         return -1 ;
108     } else if ( a[0] > b[0] ) {
109         return +1 ;
110     } else {
111         return 0 ;
112     }
113 }
114
115 // This function compares the second (index 1) items in the subarrays, which are the
116 // INSTRUCTOR NAMES. It returns -1 if the 1st item should come before the 2nd. +1 if
117 // the 1st item should come after the 2nd, 0 if their current order is OK.
118 var fnNameAscending = function( a, b ) {
119     if ( a[1] < b[1] ) {
120         return -1 ;
121     } else if ( a[1] > b[1] ) {
122         return +1 ;
123     } else {
124         return 0 ;
125     }
126 }
127
128 // This function compares the third (index 2) items in the subarrays, which are the
129 // COURSE NAMES. It returns -1 if the 1st item should come before the 2nd. +1 if the
130 // 1st item should come after the 2nd, 0 if their current order is OK.
131 var fnCourseAscending = function( a, b ) {
```



```

197 // https://www.tutorialspoint.com/How-to-add-rows-to-a-table-using-JavaScript-DOM
198 var DisplayTable = function( nSortColumn ) {
199     // Loop through the entire array, displaying the subarray in each top-level element
200     // in a separate row.
201     for ( var k = 0 ; k < arrCSSD.length ; k++ ) {
202         let table = document.getElementById( "master" ) ;
203         // insertRow parameter -1 causes the new row is to be added at the end of the table
204         let row = table.insertRow( -1 ) ;
205         row.insertCell().innerHTML = arrCSSD[k][0] ;
206         row.insertCell().innerHTML = arrCSSD[k][1] ;
207         // if a room is closed, display that message in italics
208         if ( arrCSSD[k][2] == "Room Closed" ) {
209             row.insertCell().innerHTML = "<em>" +arrCSSD[k][2] + "</em>" ;
210         } else {
211             row.insertCell().innerHTML = arrCSSD[k][2] ;
212         }
213         row.insertCell().innerHTML = TimeSlot( arrCSSD[k][3] ) ;
214     }
215
216     // Highlight the header cell of the column on which the data is sorted.
217     // Be careful! The getElementsByTagName function returns a *collection*, which is
218     // similar to, but not quite the same as an *array*!
219     var colHeaderCells = document.getElementsByTagName("th") ;
220     colHeaderCells[nSortColumn-1].setAttribute(
221         "style", "background-color: lightgreen ; color: black" ) ;
222 }
223
224
225 //// FUNCTION TO RELOAD THE PAGE WITH A SPECIFIED SORT COLUMN
226
227 // This function is executed when the user clicks a column header cell. The parameter
228 // passed to this function is the number of the column on which to sort the table.
229 var ReloadPage = function( nSortColumn ) {
230     // The following statement returns:
231     //     "/StudentResources/CodeSamples/ArrayExample_4b.html"
232     var strThisPagePath = window.location.pathname ;
233
234     // To this we want to add a parameter, which is part of the search (or query) string.
235     // A question mark begins the search string part of a URL.
236     // Each parameter is then written in name=value format (without quotes).
237     // Thus, what we want is the page path returned by the above statement, then a
238     // question mark, then the search parameter name ("SortColumn") followed by an
239     // equals sign (=) and finally followed by the parameter value, which in our
240     // program is the number of the desired sort column passed to this function.
241     var strFullPagePath = strThisPagePath + "?SortColumn=" + nSortColumn ;
242
243     // The final step is to replace the current page with the full path that we just
244     // constructed. In our program, this is the same page that we are currently viewing,
245     // but with a parameter specifying the desired sort column.
246     window.location.replace( strFullPagePath ) ;
247 }
248 </script>
249
250 </head>
251 <body>
252
253 <!-- display the page title and semester dates -->
254 <h2 id="title">Corrections Special School District<br>
255 Career &amp; Technical Education Center</h2>
256
257
258 <!-- display the name of this HTML file and its search string -->
259 <p><em>This is file:&nbsp;&nbsp;&nbsp;</em><strong> <span id="filename"></span></strong><br>
260 <em>The search string is:&nbsp;&nbsp;&nbsp;</em><strong> <span id="searchstring"></span></strong></p>
261
262 <!-- display the term for which data has been loaded -->

```

```

263 <h3 id="semester"><em>Term:</em>&nbsp;<span id="term"></span></h3>
264
265 <!-- display the skeleton of the message stating how the data is sorted -->
266 <p>This is option <span id="OptionNumber"></span>, in which the data is ordered by
267 <span id="OptionName"></span>.</p>
268
269 <script>
270 // the following code completes the above message stating how the data is sorted
271 document.getElementById( "OptionNumber" ).innerHTML = nSortColumn ;
272 switch ( nSortColumn ) {
273     case 1 : // sorted on the Room Number
274         document.getElementById( "OptionName" ).innerHTML = "Room Number" ;
275         break ;
276     case 2 : // sorted on the Instructor Name
277         document.getElementById( "OptionName" ).innerHTML = "Instructor Name" ;
278         break ;
279     case 3 : // sorted on the Course name
280         document.getElementById( "OptionName" ).innerHTML = "Course Name" ;
281         break ;
282     case 4 : // sorted on the Time Slot
283         document.getElementById( "OptionName" ).innerHTML = "Time Slot" ;
284         break ;
285 }
286 </script>
287
288 <!-- This is the table of room numbers, instructor names, course names, and time slots. -->
289 <table id="master" cellpadding="5" cellspacing="0" border="1">
290 <tr>
291 <!--
292     Note that each table header cell has an onclick parameter. The value of this
293     parameter is the name of a function that will be called when that table cell is
294     clicked. To that function we pass the number of the column by which we want the
295     table data to be sorted.
296 -->
297 <th onclick="ReloadPage( 1 )">Room #</th>
298 <th onclick="ReloadPage( 2 )">Instructor</th>
299 <th onclick="ReloadPage( 3 )">Course</th>
300 <th onclick="ReloadPage( 4 )">Time Slot</th>
301 </tr>
302
303 </table>
304
305 <script>
306 // insert the string showing the name of the current file
307 document.getElementById( "filename" ).innerHTML = location.pathname ;
308
309 // insert the string showing the search string
310 if ( location.search == "" ) {
311     document.getElementById( "searchstring" ).innerHTML = "{there is no search string}" ;
312 } else {
313     document.getElementById( "searchstring" ).innerHTML = location.search ;
314 }
315
316 // insert the string showing the term for this data
317 document.getElementById( "term" ).innerHTML = strTerm ;
318
319 // we are now finally ready to display the table! :)
320 DisplayTable( nSortColumn ) ;
321 </script>
322
323 </body>
324
325 </html>
326

```

Line wrap

```
1 /** http://cssdweb.edu/StudentResources/CodeSamples/ArrayExample_4b.css
2  * maintained by Jesse M. Heines, jesse@jesseheines.com
3  * Copyright (c) 2023 by Jesse Heines. All rights reserved. May be freely
4  * copied or excerpted for educational purposes with credit to the author.
5  * updated by JMH on May 29, 2023 at 8:04 AM
6  */
7
8 h2#title {                               /* main page title */
9     color: blue ;
10    margin-bottom: 0.2em ;
11 }
12 h3#semester {                           /* page subtitle (semester dates) */
13     color: blue ;
14     margin-top: 0 ;
15 }
16 table#master tr th {                   /* formatting for the table header row */
17     background-color: black ;
18     color: white ;
19 }
20 table#master tr th:hover {             /* formatting change when a header cell is moused over */
21     background-color: yellow ;
22     color: black ;
23     cursor: pointer ;
24 }
25 /* Note that the nth-child pseudo-selector's argument begins at 1, not 0. (Sigh.) */
26 table#master tr td:nth-child(1) {     /* formatting for column 1 */
27     text-align: center ;
28 }
29 table#master tr td:nth-child(4) {     /* formatting for column 4 */
30     text-align: right ;
31 }
32 table#timeslot tr td:nth-child(1),    /* formatting for the Time Slot subtable */
33 table#timeslot tr td:nth-child(3) {  /* that is, the data in column 4 */
34     width: 4.2em ;
35     text-align: right ;
36 }
37 #OptionNumber, #OptionName {         /* formatting for the option chosen message */
38     font-weight: bold ;
39     background-color: lightgreen ;
40     padding: 0 0.5em 0 0.5em ;       /* top right bottom left */
41     border-radius: 0.35em ;
42 }
43
```

```

/** http://cssdweb.edu/StudentResources/CodeSamples/ArrayExample_4b_2022-04.js
 * maintained by Jesse M. Heines
 * Copyright (c) 2023 by Jesse Heines. All rights reserved. May be freely
 * copied or excerpted for educational purposes with credit to the author.
 * updated by JMH on May 29, 2023 at 8:12 AM
 *
 * The array data in this file is for the term that began in April 2022, which
 * is why 2022-04 is part of the file name.
 *
 * This data is read from an auxiliary JavaScript file so that it can be easily
 * changed from one semester to another. Putting the data in an auxiliary file
 * separates the program LOGIC from the DATA that the program works on. This is
 * analogous to specifying all program layout via CSS to separate program CONTENT
 * from RENDITION.
 *
 * The statements in the auxiliary JavaScript file "push" onto the array, which
 * means they add data at the end of the array. Each statement pushes an entire
 * subarray onto the top level array, resulting in a data structure with the 4
 * "columns". These columns are, in order:
 * | Room # | Instructor Name | Course Name | Time Slot |
 * where "Day Period" is:
 * "A" for 7:10- 8:30
 * "B" for 8:40-10:00
 * "C" for 10:10-11:30
 * "D" for 11:40- 1:00
 * "E" for 1:10- 2:30
 */

```

```
var strTerm = "April 4 to June 17, 2022" ;
```

```

arrCSSD.push( [ 1, "Lees", "Economics", "A" ] );
arrCSSD.push( [ 1, "Lees", "Geography", "B" ] );
arrCSSD.push( [ 2, "Burzycki", "Biology IA", "D" ] );
arrCSSD.push( [ 2, "Burzycki", "Biology IB", "E" ] );
arrCSSD.push( [ 3, "Latorella", "World Literature IA", "A" ] );
arrCSSD.push( [ 3, "Latorella", "Special Units", "B" ] );
arrCSSD.push( [ 3, "Latorella", "American Literature IA", "D" ] );
arrCSSD.push( [ 3, "Latorella", "Tolkien", "E" ] );
arrCSSD.push( [ 4, "Jones", "CDL Prep.", "B" ] );
arrCSSD.push( [ 4, "Jones", "CDL Prep.", "D" ] );
arrCSSD.push( [ 6, "Mosher", "Algebra IA", "A" ] );
arrCSSD.push( [ 6, "Mosher", "Computation for Business", "B" ] );
arrCSSD.push( [ 7, "Piper", "Short Story", "D" ] );
arrCSSD.push( [ 7, "Piper", "Mythology", "E" ] );
arrCSSD.push( [ 8, "Kroll", "Purchasing", "D" ] );
arrCSSD.push( [ 8, "Kroll", "Nutrition for Culinary", "E" ] );
arrCSSD.push( [ 10, "Spires", "Small Business Management", "A" ] );
arrCSSD.push( [ 10, "Spires", "Advanced Microsoft Applications", "D" ] );
arrCSSD.push( [ 10, "Spires", "Career Awareness", "E" ] );

```

```

/** http://cssdweb.edu/StudentResources/CodeSamples/ArrayExample_4b_2023-04.js
 * maintained by Jesse M. Heines
 * Copyright (c) 2023 by Jesse Heines. All rights reserved. May be freely
 * copied or excerpted for educational purposes with credit to the author.
 * updated by JMH on June 1, 2023 at 4:49 PM
 *
 * The array data in this file is for the term that began in April 2023, which
 * is why 2023-04 is part of the file name.
 *
 * This data is read from an auxiliary JavaScript file so that it can be easily
 * changed from one semester to another. Putting the data in an auxiliary file
 * separates the program LOGIC from the DATA that the program works on. This is
 * analogous to specifying all program layout via CSS to separate program CONTENT
 * from RENDITION.
 *
 * The statements in the auxiliary JavaScript file "push" onto the array, which
 * means they add data at the end of the array. Each statement pushes an entire
 * subarray onto the top level array, resulting in a data structure with the 4
 * "columns". These columns are, in order:
 * | Room # | Instructor Name | Course Name | Time Slot |
 * where "Day Period" is:
 * "A" for 7:10- 8:30
 * "B" for 8:40-10:00
 * "C" for 10:10-11:30
 * "D" for 11:40- 1:00
 * "E" for 1:10- 2:30
 */

```

```
var strTerm = "April 3 to June 16, 2023" ;
```

```

arrCSSD.push( [ 1, "Lees", "Room Closed", "A" ] );
arrCSSD.push( [ 1, "Lees", "Room Closed", "B" ] );
arrCSSD.push( [ 1, "Lees", "Economics (Mon-Thurs)", "D" ] );
arrCSSD.push( [ 1, "Lees", "Civics", "E" ] );
arrCSSD.push( [ 2, "Burzycki", "Biology 1A", "A" ] );
arrCSSD.push( [ 2, "Burzycki", "Skills for Success", "B" ] );
arrCSSD.push( [ 2, "Burzycki", "Physical Sciences 1A", "D" ] );
arrCSSD.push( [ 2, "Burzycki", "Personal Nutrition", "E" ] );
arrCSSD.push( [ 3, "Latorella", "American Literature 1A", "A" ] );
arrCSSD.push( [ 3, "Latorella", "British Literature 1A", "B" ] );
arrCSSD.push( [ 3, "Latorella", "Gothic Literature", "D" ] );
arrCSSD.push( [ 3, "Latorella", "Drama as Literature", "E" ] );
arrCSSD.push( [ 4, "", "Room Closed", "A" ] );
arrCSSD.push( [ 4, "", "Room Closed", "B" ] );
arrCSSD.push( [ 4, "", "Room Closed", "D" ] );
arrCSSD.push( [ 4, "", "Room Closed", "E" ] );
arrCSSD.push( [ 5, "", "Peer-to-Peer Hi SET", "A" ] );
arrCSSD.push( [ 5, "", "Peer-to-Peer Study Hall", "B" ] );
arrCSSD.push( [ 5, "", "Peer-to-Peer Hi SET", "D" ] );
arrCSSD.push( [ 5, "", "Peer-to-Peer Study Hall", "E" ] );
arrCSSD.push( [ 6, "O'Donnell", "Safety and Sanitation", "A" ] );
arrCSSD.push( [ 6, "O'Donnell", "Culinary Measurements", "B" ] );
arrCSSD.push( [ 6, "O'Donnell", "Safety and Sanitation", "D" ] );
arrCSSD.push( [ 6, "O'Donnell", "Culinary Measurements", "E" ] );
arrCSSD.push( [ 7, "Piper", "Grammar 1B", "A" ] );
arrCSSD.push( [ 7, "Piper", "Room Closed", "B" ] );
arrCSSD.push( [ 7, "Piper", "Poetry", "D" ] );
arrCSSD.push( [ 7, "Piper", "ESL", "E" ] );
arrCSSD.push( [ 8, "Mosher", "Algebra 1A", "A" ] );
arrCSSD.push( [ 8, "Mosher", "Geometry 1A", "B" ] );
arrCSSD.push( [ 8, "Mosher", "Room Closed", "D" ] );

```



```
arrCSSD.push( [ 8, "Mosher", "Room Closed", "E" ] );
arrCSSD.push( [ 9, "", "Room Closed", "A" ] );
arrCSSD.push( [ 9, "Hei nes", "Web Design (Tues/Thurs)", "B" ] );
arrCSSD.push( [ 9, "", "Room Closed", "D" ] );
arrCSSD.push( [ 9, "", "Room Closed", "E" ] );
arrCSSD.push( [ 10, "Spi res", "Marketing II", "A" ] );
arrCSSD.push( [ 10, "Spi res", "Room Closed", "B" ] );
arrCSSD.push( [ 10, "Spi res", "START I & II", "D" ] );
arrCSSD.push( [ 10, "Spi res", "Computer Fundamental s", "E" ] );
```

```

/** http://cssdweb.edu/StudentResources/CodeSamples/ArrayExample_4b_2023-07.js
 * maintained by Jesse M. Heines
 * Copyright (c) 2023 by Jesse Heines. All rights reserved. May be freely
 * copied or excerpted for educational purposes with credit to the author.
 * updated by JMH on June 1, 2023 at 4:49 PM
 *
 * The array data in this file is for the term that began in July 2023, which
 * is why 2023-07 is part of the file name.
 *
 * This data is read from an auxiliary JavaScript file so that it can be easily
 * changed from one semester to another. Putting the data in an auxiliary file
 * separates the program LOGIC from the DATA that the program works on. This is
 * analogous to specifying all program layout via CSS to separate program CONTENT
 * from RENDITION.
 *
 * The statements in the auxiliary JavaScript file "push" onto the array, which
 * means they add data at the end of the array. Each statement pushes an entire
 * subarray onto the top level array, resulting in a data structure with the 4
 * "columns". These columns are, in order:
 * | Room # | Instructor Name | Course Name | Time Slot |
 * where "Day Period" is:
 * "A" for 7:10- 8:30
 * "B" for 8:40-10:00
 * "C" for 10:10-11:30
 * "D" for 11:40- 1:00
 * "E" for 1:10- 2:30
 */

```

```
var strTerm = "July 6 to September 14, 2023" ;
```

```

arrCSSD.push( [ 1, "Lees", "World Geography", "A" ] ) ;
arrCSSD.push( [ 1, "Lees", "Modern World History", "B" ] ) ;
arrCSSD.push( [ 1, "Lees", "US History 1A", "D" ] ) ;
arrCSSD.push( [ 1, "Lees", "Psychology", "E" ] ) ;
arrCSSD.push( [ 2, "Burzycki", "Biology 1B", "A" ] ) ;
arrCSSD.push( [ 2, "Burzycki", "Physical Sciences 1B", "B" ] ) ;
arrCSSD.push( [ 2, "Burzycki", "Bunk to 5K", "D" ] ) ;
arrCSSD.push( [ 2, "Burzycki", "Skills for Success", "E" ] ) ;
arrCSSD.push( [ 3, "Latorella", "Anime English", "A" ] ) ;
arrCSSD.push( [ 3, "Latorella", "American Western", "B" ] ) ;
arrCSSD.push( [ 3, "", "Room Closed", "D" ] ) ;
arrCSSD.push( [ 3, "Latorella", "20th Century Literature", "E" ] ) ;
arrCSSD.push( [ 4, "", "Room Closed", "A" ] ) ;
arrCSSD.push( [ 4, "", "Room Closed", "B" ] ) ;
arrCSSD.push( [ 4, "", "Room Closed", "D" ] ) ;
arrCSSD.push( [ 4, "", "Room Closed", "E" ] ) ;
arrCSSD.push( [ 5, "", "Peer-to-Peer HI SET", "A" ] ) ;
arrCSSD.push( [ 5, "", "College Lab", "B" ] ) ;
arrCSSD.push( [ 5, "", "Peer-to-Peer HI SET", "D" ] ) ;
arrCSSD.push( [ 5, "", "College Lab", "E" ] ) ;
arrCSSD.push( [ 6, "Owens", "Nutrition for Culinary", "A" ] ) ;
arrCSSD.push( [ 6, "Owens", "Customer Service", "B" ] ) ;
arrCSSD.push( [ 6, "Owens", "Nutrition for Culinary", "D" ] ) ;
arrCSSD.push( [ 6, "Owens", "Customer Service", "E" ] ) ;
arrCSSD.push( [ 7, "", "Room Closed", "A" ] ) ;
arrCSSD.push( [ 7, "", "Room Closed", "B" ] ) ;
arrCSSD.push( [ 7, "", "Room Closed", "D" ] ) ;
arrCSSD.push( [ 7, "", "Room Closed", "E" ] ) ;
arrCSSD.push( [ 8, "Mosher", "Geometry 1B", "A" ] ) ;
arrCSSD.push( [ 8, "Mosher", "Room Closed", "B" ] ) ;
arrCSSD.push( [ 8, "Mosher", "Computation for Business", "D" ] ) ;

```

```
arrCSSD.push( [ 8, "Mosher", "Pre-Algebra", "E" ] );  
arrCSSD.push( [ 9, "", "Room Closed", "A" ] );  
arrCSSD.push( [ 9, "", "Room Closed", "B" ] );  
arrCSSD.push( [ 9, "Latorella", "Writing 1A", "D" ] );  
arrCSSD.push( [ 9, "", "Room Closed", "E" ] );  
arrCSSD.push( [ 10, "Daynard", "Small Business Management", "A" ] );  
arrCSSD.push( [ 10, "Daynard", "Accounting I", "B" ] );  
arrCSSD.push( [ 10, "Daynard", "Career Awareness", "D" ] );  
arrCSSD.push( [ 10, "Daynard", "Personal Finance", "E" ] );
```