```
1   // Hello.
2   //
3   // This is JSHint, a tool that helps to detect errors and potential
4   // problems in your JavaScript code.
5   //
6   // To start, simply enter some JavaScript anywhere on this page. Your
7   // report will appear on the right side.
8   //
9   // Additionally, you can toggle specific options in the Configure
10  // menu.
11
12  function main() {
13    return 'Hello, World!';
14  }
15
16  main();
17  // Global variables
18
19  let menu = document.getElementById('menu-container').innerHTML;
20  let informationContainer = document.getElementById("information-container");
21  let instructionContainer = document.getElementById("instruction-container");
22  let payContainer = document.getElementById("pay-container");
23  let tabButtons = document.querySelectorAll('.tablinks');
24
25  // Income Tax Bands and rates
26  const taxBand1 = 42000;
27  const lowerRate = 0.20;
28  const higherRate = 0.40;
29
30  // USC Rates
31  const uscRate1 = 0.005;
32  const uscRate2 = 0.02;
33  const uscRate3 = 0.04;
34  const uscRate4 = 0.08;
35  const usc1 = 12012 * uscRate1;
36  const usc2 = 13748 * uscRate2;
37  const usc3 = 44284 * uscRate3;
38
39
40  // Tax credits for 2024
41  const taxCreditSingle = 1875;
42  const taxCreditPaye = 1875;
43  const taxCreditMarried = 1875;
44  const taxCreditSpcc = 1750;
45  const taxCreditRentS = 750;
46  const taxCreditRentM = 1500;
47  const taxCreditHomeCare = 1800;
48  const taxCreditAgeSingle = 245;
49  const taxCreditAgeMarried = 490;
50  let incomeTax;
51  let netIncomeTax;
52  let otherTaxCredits = 0;
53  let reducedTaxCredit = 0;
54
55  let uscDeduction;
56  let prsiDeduction;
57
58  let details = ['Salary', 'Income Tax', 'USC', 'PRSI'];
59  let timePeriods = ['Yearly', 'Monthly', 'Weekly'];
60  let values;
61  let netPay;
62
63  //PRSI rates
64  const prsiRate = 0.04;
65
66
67  // Iteration Statement
68  for (let i = 0; i < tabButtons.length; i++) {
69      tabButtons[i].addEventListener('click', () => {
70          let tabId = tabButtons[i].getAttribute('data-tab');
71          let tabContents = document.querySelectorAll('.tabscontent');
72          //To hide all tabs
73          for (let j = 0; j < tabContents.length; j++) {
74              tabContents[j].style.display = 'none';
75          }
76
77          //Show the active tab
78          document.getElementById(tabId).style.display = 'block';
79          document.getElementById(tabId).style.backgroundColor = '#56BFB5';
80      });
81  }
82
83  // FUNCTIONS
84  // Display menu options in main container
85  window.onload = menuOptions();
86
87  function menuOptions() {
88      document.getElementById("main-container").innerHTML = menu;
89  }
90
91  // Modal container elements
92  let infoButton = document.getElementById("info-button");
93  let infoButtonHide = document.getElementById("information-close");
94  let instructButton = document.getElementById("inst-button");
95  let instructButtonHide = document.getElementById("instruction-close");
96  let calcButton = document.getElementById("calcPay-button");
97  let calcButtonHide = document.getElementById("calc-close");
98
99  // Show-Hide event listeners for main container
100 infoButton.addEventListener('click', showInfoContainer);
101 infoButtonHide.addEventListener('click', hideInfoContainer);
102 instructButton.addEventListener('click', showInstructionContainer);
103 instructButtonHide.addEventListener('click', hideInstructionContainer);
104 calcButtonHide.addEventListener('click', hideCalcsContainer);
105 calcButton.addEventListener('click', showCalcContainer);
106
107
108
109 // Eventlisteners
```

CONFIGURE

Metrics

There are 26 functions in this file.

Function with the largest signature take 4 arguments, while the median is 0.

Largest function has 34 statements in it, while the median is 3.

The most complex function has a cyclomatic complexity value of 23 while the median is 1.

One warning

69 Functions declared within loops referencing an outer scoped variable may lead to confusing semantics. (tabButtons, i)

```
110  document.getElementById('calculateButton-adv').addEventListener('click', calculateSalaryAd
111  document.getElementById('calculateButton').addEventListener('click', calculateSalary);
112  document.addEventListener('DOMContentLoaded', function() {
113      const singleInput = document.getElementById('single');
114      const marriedInput = document.getElementById('married');
115      const spouseIncomeInput = document.getElementById('spouse-income');
116      const ageInput = document.getElementById('age');
117
118      singleInput.addEventListener('change', function() {
119          spouseIncomeInput.disabled = true;
120          updateTaxCredits(); // Update tax credits whenever singleInput Changes
121      });
122
123      marriedInput.addEventListener('change', function() {
124          spouseIncomeInput.disabled = false;
125      });
126
127
128      ageInput.addEventListener('change', updateTaxCredits);
129
130  });
131
132  // To show/hide info container
133  // Show information container
134  function showInfoContainer() {
135      informationContainer.style.display = 'block';
136  }
137
138  // Hide information container
139  function hideInfoContainer() {
140      informationContainer.style.display = 'none';
141  }
142
143  // To show/hide instructions container
144  // Instruction container show
145  function showInstructionContainer() {
146      instructionContainer.style.display = 'block';
147  }
148
149  // Instruction container hide
150  function hideInstructionContainer() {
151      instructionContainer.style.display = 'none';
152  }
153
154  // To show calculate pay container occupying replacing all elements in the main-container
155  function showCalcContainer() {
156      payContainer.style.display = 'block';
157  }
158
159  // to hide pay container
160  function hideCalcsContainer() {
161      payContainer.style.display = 'none';
162      document.getElementById('calcResultsContainer').innerHTML = '';
163      document.getElementById('salary').value = '';
164  }
165
166  function validateYearlySalary(yearlySalary) {
167      // Validate the salary input to not accept 0 and negative values
168      if (yearlySalary < 0) {
169          alert('Salary cannot be negative.');
170          return false;
171      } else if (!yearlySalary) {
172          alert('Please enter your salary.');
173          return false;
174      }
175      return true;
176  }
177
178  // Calculation Section
179  function calculateUSCDeduction(yearlySalary) {
180
181      //USC Calculation
182      if (yearlySalary <= 13000) {
183          uscDeduction = 0;
184      } else if (yearlySalary > 12012 && yearlySalary <= 25760) {
185          uscDeduction = Math.round(usc1 + ((yearlySalary - 12012) * uscRate2));
186      } else if (yearlySalary > 25760 && yearlySalary <= 70044) {
187          uscDeduction = Math.round(usc1 + usc2 + ((yearlySalary - 25760) * uscRate3));
188      } else {
189          uscDeduction = Math.round(usc1 + usc2 + usc3 + ((yearlySalary - 70044) * uscRate4)
190      }
191
192      return uscDeduction;
193  }
194
195  function calculatePRSIDeduction(yearlySalary) {
196      /**Calculate PRSI deductions
197      PRSI is nil if the yearly salary is less than or equal to €18,304 per year or €352 per
198      **/
199      if (yearlySalary <= 18304) {
200          prsiDeduction = 0;
201      } else {
202          prsiDeduction = Math.round(yearlySalary * prsiRate);
203      }
204
205      return prsiDeduction;
206  }
207
208  function calculateIncomeTax(yearlySalary) {
209
210  // Calculate income tax
211      if (yearlySalary <= 18750) {
212          incomeTax = 0;
213      } else if (yearlySalary > 18750 && yearlySalary <= taxBand1) {
214          incomeTax = Math.round((yearlySalary * lowerRate));
215      } else {
216          incomeTax = Math.round(((taxBand1 * lowerRate) + ((yearlySalary - taxBand1) * high
217      }
218
219      return incomeTax;
220  }
```

Metrics

There are 26 functions in this file.

Function with the largest signature take 4 arguments, while the median is 0.

Largest function has 34 statements in it, while the median is 3.

The most complex function has a cyclomatic complexity value of 23 while the median is 1.

One warning

69 Functions declared within loops referencing an outer scoped variable may lead to confusing semantics. (tabButtons, i)

```
221
222 function calculateSalary() {
223     const yearlySalary = parseFloat(document.getElementById('salary').value);
224
225     // Validate the salary input
226     if (!validateYearlySalary(yearlySalary)) {
227         return;
228     }
229
230     // Calculate Income Tax
231     incomeTax = calculateIncomeTax(yearlySalary);
232
233     // Calculate Net Income tax
234     if (incomeTax == 0) {
235         netIncomeTax = 0;
236     } else {
237         netIncomeTax = incomeTax - taxCreditSingle - taxCreditPaye;
238     }
239
240     //Calculate USC Deductions
241     uscDeduction = calculateUSCDeduction(yearlySalary);
242
243     //Calculate PRSI Deductions
244     prsiDeduction = calculatePRSIDeduction(yearlySalary);
245
246     calculateMonthlyWeekly(yearlySalary, netIncomeTax, uscDeduction, prsiDeduction);
247
248     //function to create the result table
249     createResultTable(details, timePeriods, values, netPay);
250
251 }
252
253 function updateTaxCredits() {
254     const singleInput = document.getElementById('single');
255     const marriedInput = document.getElementById('married');
256     const ageInput = document.getElementById('age');
257     const depInput = document.getElementById('depyes');
258     const rentInput = document.getElementById('rentyes');
259     const spouseIncomeInput = document.getElementById('spouse-income');
260     let spouseIncome = parseInt(spouseIncomeInput.value);
261     const age = parseFloat(ageInput.value);
262
263     // Reset otherTaxCredits
264     otherTaxCredits = 0;
265
266     if (singleInput.checked && parseInt(ageInput.value) >= 65) {
267         otherTaxCredits += taxCreditAgeSingle;
268     }
269
270     if (singleInput.checked && depInput.checked) {
271         otherTaxCredits += taxCreditSpcc;
272     }
273     if (singleInput.checked && rentInput.checked) {
274         otherTaxCredits += taxCreditRentS;
275     }
276     if (marriedInput.checked) {
277         otherTaxCredits += taxCreditMarried;
278     }
279     if (marriedInput.checked && parseInt(ageInput.value) >= 65) {
280         otherTaxCredits += taxCreditAgeMarried;
281     }
282     if (marriedInput.checked && rentInput.checked) {
283         otherTaxCredits += taxCreditRentM;
284     }
285
286     if (marriedInput.checked && depInput.checked) {
287         if (parseInt(spouseIncomeInput.value) >= 0 && parseInt(spouseIncomeInput.value) <=
288             otherTaxCredits += taxCreditHomeCare;
289         } else if (parseInt(spouseIncomeInput.value) > 7200 && parseInt(spouseIncomeInput.
290             reducedTaxCredit = taxCreditHomeCare - ((parseInt(spouseIncomeInput.value)-720
291             otherTaxCredits += reducedTaxCredit;
292         }
293     }
294
295     // Validate age input
296     if(isNaN(age) || age < 14 || age > 110) {
297         alert('Please enter a valid age between 14 to 110.');
298         return;
299     }
300
301     // Validate spouse income
302     if (isNaN(spouseIncome) || spouseIncome < 0) {
303         alert('Spouse income must not be a negative value.');
304         spouseIncome = 0;
305         spouseIncomeInput.value = spouseIncome; // Update the input field
306         return false;
307     }
308
309 }
310
311 function calculatePension(yearlySalary) {
312     const pensionInput = document.getElementById('pension');
313     const pensionPercentage = parseFloat(pensionInput.value);
314
315     if (isNaN(pensionPercentage)) {
316         alert('Please enter a valid pension contribution percentage.');
317         return;
318     } else if(pensionPercentage < 0 || pensionPercentage > 40 ) {
319         alert('Invalid pension contribution.');
320         return;
321     }
322
323     return (yearlySalary * pensionPercentage) / 100;
324
325 }
326
327 function calculateSalaryAdv() {
328     const yearlySalary = parseFloat(document.getElementById('salary-adv').value);
329
330     // Validate the salary input
331     if (!validateYearlySalary(yearlySalary)) {
```

JSHint version 2.13.6 (https://github.com/jshint/j

Metrics

There are 26 functions in this file.

Function with the largest signature take 4 arguments, while the median is 0.

Largest function has 34 statements in it, while the median is 3.

The most complex function has a cyclomatic complexity value of 23 while the median is 1.

One warning

69 Functions declared within loops referencing an outer scoped variable may lead to confusing semantics. (tabButtons, i)

```
332         return;
333     }
334
335     // Calculate pension contribution
336     const pensionContribution = calculatePension(yearlySalary);
337
338     // Calculate USC Deductions
339     uscDeduction = calculateUSCDeduction(yearlySalary);
340
341     // Calculate PRSI Deductions
342     prsiDeduction = calculatePRSIDeduction(yearlySalary);
343
344     updateTaxCredits();
345
346     // Calculate total tax credits
347     const totalTaxCredits = taxCreditSingle + taxCreditPaye + otherTaxCredits;
348     const taxableIncome = yearlySalary - pensionContribution;
349
350     // Calculate Income Tax
351     incomeTax = calculateIncomeTax(taxableIncome);
352
353      // Calculate Net Income tax
354      if (incomeTax == 0) {
355         netIncomeTax = 0;
356     } else {
357         netIncomeTax = incomeTax - totalTaxCredits;
358     }
359
360     if (netIncomeTax < 0) {
361         netIncomeTax = 0;
362     }
363
364     calculateMonthlyWeekly(yearlySalary, netIncomeTax, uscDeduction, prsiDeduction);
365
366     // function to create the result table
367     createResultTableAdv(details, timePeriods, values, netPay, pensionContribution);
368
369 }
370
371
372 function calculateMonthlyWeekly(yearlySalary, netIncomeTax, uscDeduction, prsiDeduction){
373     // Calculate Gross Monthly and Weekly Gross Salary
374     const grossMonthlySalary = Math.round(yearlySalary / 12);
375     const grossWeeklySalary = Math.round(yearlySalary / 52);
376
377     // Monthly and weekly calculations
378      const monthlyUsc = Math.round(uscDeduction / 12);
379      const monthlyPrsi = Math.round(prsiDeduction / 12);
380      const weeklyUsc = Math.round(uscDeduction / 52);
381      const weeklyPrsi = Math.round(prsiDeduction / 52);
382      const monthlyTax = Math.round(netIncomeTax / 12);
383      const weeklyTax = Math.round(netIncomeTax / 52);
384      //const monthlyPension = Math.round(pensionContribution/12);
385      //const weeklyPension = Math.round(pensionContribution/52);
386
387     // Calculate net salary after deductions
388     const netYearlySalary = yearlySalary - netIncomeTax - uscDeduction - prsiDeduction;
389     const netMonthlySalary = Math.round(netYearlySalary / 12);
390     const netWeeklySalary = Math.round(netYearlySalary / 52);
391
392     // Data for the table
393     values = [yearlySalary, grossMonthlySalary, grossWeeklySalary, netIncomeTax, monthlyTa
394     netPay = [netYearlySalary, netMonthlySalary, netWeeklySalary];
395
396 }
397
398 // Table for Basic
399 function createResultTable(details, timePeriods, values, netPay) {
400     const table = ResultTable(details, timePeriods, values, netPay);
401     displayResultTable(table);
402 }
403
404 // Table for Advance
405 function createResultTableAdv(details, timePeriods, values, netPay) {
406     const table = ResultTable(details, timePeriods, values, netPay);
407     displayResultTableAdv(table);
408 }
409
410 function ResultTable(details, timePeriods, values, netPay) {
411     const table = document.createElement('table');
412     const thead = document.createElement('thead');
413     const tbody = document.createElement('tbody');
414
415     // Create table headers
416     const headerRow = thead.insertRow(0);
417     headerRow.insertCell(0).textContent = 'Details'; // Label for details column
418     for (let i = 0; i < timePeriods.length; i++) {
419         const headerCell = headerRow.insertCell(i + 1);
420         headerCell.textContent = timePeriods[i];
421     }
422
423     // Create rows
424     for (let i = 0; i < details.length; i++) {
425         const row = tbody.insertRow(i);
426         const detailsCell = row.insertCell(0);
427         detailsCell.textContent = details[i];
428
429         for (let j = 0; j < timePeriods.length; j++) {
430             const valueCell = row.insertCell(j + 1);
431             valueCell.textContent = `€${values[i * timePeriods.length + j].toLocaleString(
432         }
433     }
434
435     // Create a row for net pay values
436     const netPayRow = tbody.insertRow(details.length);
437     netPayRow.insertCell(0).textContent = 'Net Pay';
438     for (let i = 0; i < netPay.length; i++) {
439         const cell = netPayRow.insertCell(i + 1);
440         cell.textContent = `€${netPay[i].toLocaleString()}`;
441     }
442
```

Metrics

There are 26 functions in this file.

Function with the largest signature take 4 arguments, while the median is 0.

Largest function has 34 statements in it, while the median is 3.

The most complex function has a cyclomatic complexity value of 23 while the median is 1.

One warning

69 Functions declared within loops referencing an outer scoped variable may lead to confusing semantics. (tabButtons, i)

```
443         table.appendChild(thead);
444         table.appendChild(tbody);
445 
446         return table;
447     }
448 
449 
450     function displayResultTable(table){
451         const resultsContainer = document.getElementById('calcResultsContainer');
452 
453         // Clear existing content
454         resultsContainer.innerHTML = '';
455 
456         // Append the generated table
457         resultsContainer.appendChild(table);
458     }
459 
460     function displayResultTableAdv(table) {
461         const resultsContainer = document.getElementById('calcResultsContainerAdv');
462 
463         // Clear existing content
464         resultsContainer.innerHTML = '';
465 
466         // Append the generated table
467         resultsContainer.appendChild(table);
468     }
```

Metrics

There are 26 functions in this file.

Function with the largest signature take 4 arguments, while the median is 0.

Largest function has 34 statements in it, while the median is 3.

The most complex function has a cyclomatic complexity value of 23 while the median is 1.

One warning

69 Functions declared within loops referencing an outer scoped variable may lead to confusing semantics. (tabButtons, i)