

# Develop the Income Tax Calculator

Create a web-based Income Tax Calculator that allows users to input their annual income and calculate their tax liability based on specified tax brackets.

## Requirements:

### 1. HTML Structure:

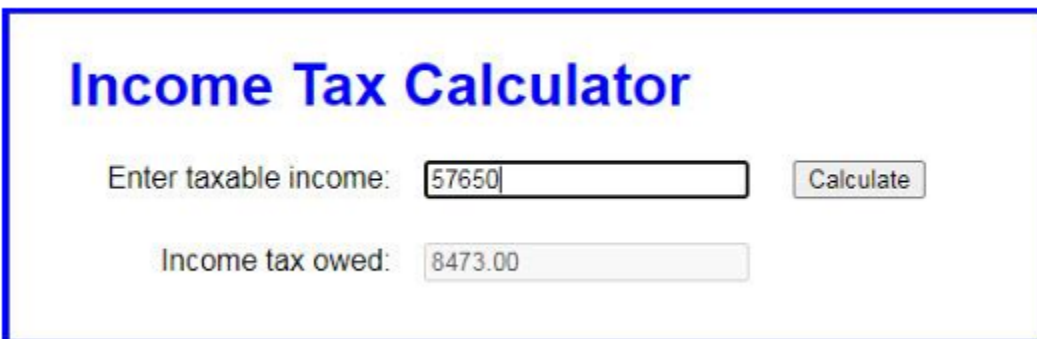
- An input field for entering the annual income.
- A button to calculate the tax.
- A display area to show the calculated tax.
- Optional: Add fields for tax deductions or credits.

### 2. JavaScript Functionality:

- Use function expressions and arrow functions to handle calculations.
- Implement event listeners for button clicks.
- Use conditional statements to apply different tax rates based on income brackets.

### 3. CSS Styling:

- Apply basic styling to create a user-friendly interface. The design is flexible and can be modified as needed.



The screenshot shows a web-based Income Tax Calculator interface. It features a title "Income Tax Calculator" in a large, bold, blue font. Below the title, there is a label "Enter taxable income:" followed by a text input field containing the value "57650". To the right of the input field is a button labeled "Calculate". Below the input field, there is a label "Income tax owed:" followed by a text input field displaying the calculated value "8473.00". The entire interface is enclosed in a blue border.

## Tax Brackets (Example):

- 0% for income up to \$10,000
- 10% for income from \$10,001 to \$40,000
- 20% for income from \$40,001 to \$100,000
- 30% for income over \$100,000

### Deliverables:

- A zipped folder containing:
  - HTML file
  - CSS file
  - JavaScript file

### Bonus Challenges (Optional):

- Allow users to input deductions and adjust the taxable income.
- Add a reset button to clear the input and results.
- Implement local storage to save previous calculations.

### Assignments Grading Rubric

	<b>Unsatisfactory ( ≤ 40% of the points)</b>	<b>Satisfactory (60% of the points)</b>	<b>Good (80% of the points)</b>	<b>Excellent (100% of the points)</b>
<b>Requirements and Delivery (4 points)</b>	<ul style="list-style-type: none"><li>• Completed less than 70% of the requirements.</li><li>• Delivered on time but not in correct format.</li></ul>	<ul style="list-style-type: none"><li>• Completed between 70- 80% of the requirements.</li><li>• Delivered on time, and in correct format.</li></ul>	<ul style="list-style-type: none"><li>• Completed between 80-90% of the requirements.</li><li>• Delivered on time, and in correct format.</li></ul>	<ul style="list-style-type: none"><li>• Completed between 90-100% of the requirements.</li><li>• Delivered on time, and in correct format.</li></ul>
<b>Coding Standards (2 points)</b>	<ul style="list-style-type: none"><li>• No name, date, or assignment title included</li><li>• Poor use of white space (indentation, blank lines).</li><li>• Disorganized and messy</li><li>• Poor use of variables (many ambiguous naming).</li></ul>	<ul style="list-style-type: none"><li>• Includes name, assignment title.</li><li>• White space makes program fairly easy to read.</li><li>• Organized work.</li><li>• Good use of variables (few unambiguous naming).</li></ul>	<ul style="list-style-type: none"><li>• Includes name, and assignment title.</li><li>• Good use of white space.</li><li>• Organized work.</li><li>• Good use of variables (no global variables, unambiguous naming)</li></ul>	<ul style="list-style-type: none"><li>• Includes name, and assignment title.</li><li>• Excellent use of white space.</li><li>• Creatively organized work.</li><li>• Excellent use of variables (no global variables, unambiguous naming).</li></ul>

<b>Runtime (2 points)</b>	<ul style="list-style-type: none"> <li>• Does not execute due to errors.</li> <li>• User prompts are misleading or non-existent.</li> <li>• No testing has been completed, or no input validation.</li> </ul>	<ul style="list-style-type: none"> <li>• Executes without errors.</li> <li>• User prompts contain little information, poor design.</li> <li>• Some testing or input validation has been completed.</li> </ul>	<ul style="list-style-type: none"> <li>• Executes without errors.</li> <li>• User prompts are understandable, minimum use of symbols or spacing in output.</li> <li>• Most testing or input validation completed</li> </ul>	<ul style="list-style-type: none"> <li>• Executes without errors excellent user prompts, good use of symbols, spacing in output.</li> <li>• Thorough and organized testing or input validation has been completed.</li> </ul>
<b>Documentation (2 points)</b>	<ul style="list-style-type: none"> <li>• Very limited or no documentation included.</li> <li>• Documentation does not help the reader understand the code.</li> </ul>	<ul style="list-style-type: none"> <li>• Basic documentation has been completed including descriptions of all class variables.</li> <li>• Purpose is noted for each function</li> </ul>	<ul style="list-style-type: none"> <li>• Clearly documented including descriptions of all class variables.</li> <li>• Specific purpose is noted for each function and control structure.</li> </ul>	<ul style="list-style-type: none"> <li>• Clearly and effectively documented including descriptions of all class variables.</li> <li>• Specific purpose noted for each function, control structure, input requirements, and output results.</li> </ul>