Evaluation Tasks for Adserver Javascript Developer Position

In order to evaluate the skills needed for the Adserver Javascript developer position, we request applicants to provide solutions for the following tasks, together with several test scenarios that prove their solution.

Efficient Integer Conversion

Create a function called, that accepts a positive integer and converts it into 1 in the most effective way possible. The function should return the number of steps executed. You can use only these three possible operations to convert the integer into 1:

1. Add One
2. Remove One
3. Divide by 2 (only if the current amount is even)

For example:
solution(4) # Should return 2 (using 2 steps to convert 4 into 1): 4 -> 2 -> 1
solution(15) # Should return 5 (using 5 steps to convert 15 into 1): 15 -> 16 -> 8 -> 4 -> 2 -> 1

Escape a labyrinth

You have a labyrinth. The map is represented as a matrix of 0s and 1s, where 0s are passable space and 1s are impassable walls. The door out of the labyrinth is at the top left (0,0) and the door into the labyrinth is at the bottom right (w-1,h-1).

Write a function solution(map) that generates the length of the shortest path from the entry point to the exit, where you are allowed to remove one wall as part of your strategy. The path length is the total number of nodes you pass through, counting both the entrance and exit. The starting and ending positions are always passable (0). The map will always be solvable, though you may or may not need to remove a wall to solve it. The height and width of the map can be from 2 to 20. Moves can only be made in cardinal directions; no diagonal moves are allowed.

For example:
Your code should pass the following test cases. Note that it may also be run against other test cases not shown here.

Input with matrix [[0, 0, 0, 0, 0, 0], [1, 1, 1, 1, 1, 0], [0, 0, 0, 0, 0, 0], [0, 1, 1, 1, 1, 1], [0, 1, 1, 1, 1, 1], [0, 0, 0, 0, 0, 0]]) Should Output: 11

Input with matrix [[0, 1, 1, 0], [0, 0, 0, 1], [1, 1, 0, 0], [1, 1, 1, 0]]) Should Output: 7

JavaScript DOM Manipulation

Description:

You're tasked with developing a JavaScript function that manipulates the Document Object Model (DOM) of any webpage to create a simplified and structured view. The function should perform the following actions:
- Remove all multimedia content (images, videos, iframes) and CSS styles on the page.
- Retrieve all elements from the page and put their textual content in a simple div element.
- Set the width of each div element to match the width of the screen.
- Insert a Sample banner element with size (728x90) between each DIV with content of your choice

Requirements:

- Use vanilla JavaScript to achieve the specified functionality. Do not rely on any libraries or frameworks.
- Ensure that only div elements are left on the page after the manipulation.
- The code should handle scenarios where different elements may contain nested multimedia content and CSS styles gracefully.
- Ensure that after applying the function, the webpage contains only the div elements with each div spanning the width of the screen + the added banners.
- Document your code to explain its purpose and any assumptions made.

Example:

Given an HTML page with various content, including elements containing textual content, images, videos, and iframes, applying the function should result in the following:
- Multimedia content (images, videos, iframes) and CSS styles are removed.
- The textual content is placed in different div elements.
- All other content on the webpage is cleared, leaving only the div elements with each div spanning the width of the screen.
- A banner with size of 728x90 is placed between each individual DIV element

Additional Information:

This task evaluates the candidate's proficiency in:
- DOM manipulation using vanilla JavaScript.
- Handling and traversing DOM elements effectively.
- Understanding and implementing client-side responsiveness.
- Ensuring graceful handling of various scenarios and edge cases.