JavaScript Array Partition

Challenge

Given an integer array `nums` of `2n` integers, group these integers into `n` pairs $(a_1, b_1), (a_2, b_2), ..., (a_n, b_n)$ such that the sum of $\min(a_i, b_i)$ for all `i` is maximized. Return the maximized sum.

1st Example

```
Input: nums = [1,4,3,2]
Output: 4
Explanation: All possible pairings
(ignoring the ordering of elements) are:
1. (1, 4), (2, 3) -> min(1, 4) + min(2, 3) = 1 + 2 = 3
2. (1, 3), (2, 4) -> min(1, 3) + min(2, 4) = 1 + 2 = 3
3. (1, 2), (3, 4) -> min(1, 2) + min(3, 4) = 1 + 3 = 4
So the maximum possible sum is 4.
```

2nd Example

```
Input: nums = [6,2,6,5,1,2]
Output: 9
Explanation:
The optimal pairing is (2, 1), (2, 5), (6, 6).
min(2, 1) + min(2, 5) + min(6, 6) = 1 + 2 + 6 = 9
```

Constraints

- $1 <= n <= 10^4$
- `nums.length == 2 * n`
- $-10^4 <= nums[i] <= 10^4$

Solution

```
const arrayPairSum = (nums) => {
    let result = 0;

    nums.sort(function (a, b) {
        return a - b;
    });

    for (let i = 0; i < nums.length; i = i + 2) {
        result += nums[i];
    }

    return result;
};
```

Explanation

I've defined a function called `arrayPairSum` that takes an array of numbers, `nums`, as input. The purpose of this function is to calculate the sum of the minimum of each pair of numbers in the array and return the total sum.

Inside the function, a variable called `result` is initialized to `0`. This variable will store the sum of the minimum of each pair.

The input array `nums` is sorted in ascending order using the `sort` method and a compare function. This ensures that the smaller numbers come before the larger numbers in the array.

A `for` loop is used to iterate over the sorted `nums` array. The loop runs from `i = 0` to `i < nums.length` with an increment of `2` in each iteration. This ensures that we only consider every second number in the array, which represents the minimum of each pair.

Within the loop, the current number `nums[i]` (which represents the minimum of the pair) is added to the `result` variable.

After the loop finishes, the final value of `result`, which represents the sum of the minimum of each pair in the array, is returned as the output of the function.

In summary, this function calculates the sum of the minimum of each pair of numbers in the input array. It achieves this by sorting the array and then iterating over it, adding the minimum of each pair to a running total. The final total is returned as the output of the function.

Author: Trevor Morin