# AI Assistance Documentation – Assignment 2

**Question 1: Find Missing Numbers in Array**

* **Prompt**:  
  How do I find all numbers missing from an array of integers from 1 to n in C#?
* **Response**:  
  Use in-place negation to track visited indices. Check for values that remain positive to identify missing ones.
* **Implementation**:  
  Followed Copilot’s suggestion by using the index-based negation approach for marking visited values. This ensured O(n) time and O(1) space.
* **Adjustment**:  
  Added an edge case check for an empty array and included exception handling.

**Question 2: Sort Array by Parity**

* **Prompt**:  
  Write a C# function to sort an array by parity, placing even numbers first.
* **Response**:  
  Separate even and odd numbers into two lists, then concatenate them.
* **Implementation**:  
  Used two lists to maintain order and combined them at the end.
* **Adjustment**:  
  Preserved the order of appearance and added exception handling.

**Question 3: Two Sum**

* **Prompt**:  
  What’s the most efficient way to implement Two Sum in C# using a dictionary?
* **Response**:  
  Use a dictionary to store indices of numbers and look up the complement.
* **Implementation**:  
  Used dictionary for constant time lookup and looped through only once.
* **Adjustment**:  
  Checked for existing keys to avoid duplicate index issues.

**Question 4: Maximum Product of Three Numbers**

* **Prompt**:  
  How do I find the maximum product of any three numbers in a C# array?
* **Response**:  
  Either multiply the three largest or two smallest and the largest numbers.
* **Implementation**:  
  Sorted the array, then used max of both options to handle negatives.
* **Adjustment**:  
  Added check for arrays with fewer than 3 elements.

**Question 5: Decimal to Binary Conversion**

* **Prompt**:  
  How can I convert an integer to binary string in C#?
* **Response**:  
  Use Convert.ToString(decimal, 2) for conversion.
* **Implementation**:  
  Directly applied the suggested built-in method.
* **Adjustment**:  
  Included condition for 0 and exception handling.

**Question 6: Find Minimum in Rotated Sorted Array**

* **Prompt**:  
  What is the binary search approach to find minimum in rotated sorted array in C#?
* **Response**:  
  Use binary search comparing mid and right elements.
* **Implementation**:  
  Applied binary search logic to locate pivot (minimum element).
* **Adjustment**:  
  Handled single-element arrays and added try-catch block.

**Question 7: Palindrome Number**

* **Prompt**:  
  Write a C# function to check if a number is a palindrome without converting it to a string.
* **Response**:  
  Reverse the number mathematically and compare it with original.
* **Implementation**:  
  Used modulo and division to reverse digits.
* **Adjustment**:  
  Included condition for negative numbers and try-catch.

**Question 8: Fibonacci Number**

* **Prompt**:  
  How do I write an iterative function to get the nth Fibonacci number in C#?
* **Response**:  
  Use a loop instead of recursion for better performance.
* **Implementation**:  
  Used a loop with three variables to generate the nth Fibonacci.
* **Adjustment**:  
  Handled n = 0 and n = 1 cases explicitly, added try-catch.