### Question 1: Find Missing Numbers in Array

**Prompt Used:**  
"Implement a C# method to find all missing numbers in an unsorted array containing numbers from 1 to n"

**Response Received:**  
Suggested using HashSet to track present numbers and checking range 1 to n.

**Implementation Details:**

* HashSet for O(1) lookups
* Linear scan for missing numbers
* Returns empty list for invalid inputs

**Adjustments:**

1. Added explicit null/empty array check
2. Maintained original order in result list
3. Preserved input array immutability

### Question 2: Sort Array by Parity

**Prompt Used:**  
"How to move all even numbers to the front of an array in C#?"

**Response Received:**  
Suggested separating evens/odds into lists then combining.

**Implementation Details:**

* Two-pointer in-place swap
* Single pass through array
* Preserves relative order within parity groups

**Adjustments:**

1. Replaced List-based solution with O(1) space in-place swap
2. Added early exit for single-element arrays
3. Optimized swap logic to reduce operations

### Question 3: Two Sum

**Prompt Used:**  
"Implement two sum problem in C# with hash map"

**Response Received:**  
Dictionary-based solution with complement checking.

**Implementation Details:**

* Single pass with dictionary
* Immediate return on solution found
* Handles duplicate values properly

**Adjustments:**

1. Added input size validation
2. Used TryGetValue for safer dictionary access
3. Returns empty array (not null) for no solution

### Question 4: Maximum Product of Three Numbers

**Prompt Used:**  
"Find maximum product of three numbers in C# array"

**Response Received:**  
Suggested sorting and comparing two product cases.

**Implementation Details:**

* Handles negative numbers correctly
* Single sort operation
* Clear two-case comparison

**Adjustments:**

1. Added minimum array length validation
2. Improved variable naming (product1/product2 → logical names)
3. Added explicit exception for invalid inputs

### **Question 5: Decimal to Binary Conversion**

**Prompt Used:**  
"Convert decimal to binary string in C#"

**Response Received:**  
Iterative division-by-2 approach.

**Implementation Details:**

* Handles zero as special case
* Builds string via prepending remainders
* Preserves sign for negatives

**Adjustments:**

1. Added negative number support
2. Optimized string building
3. Removed trailing whitespace

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

**Prompt Used:**  
"Find minimum in rotated sorted array binary search C#"

**Response Received:**  
Modified binary search approach.

**Implementation Details:**

* Checks rotation point
* Handles non-rotated edge case
* Early termination when found

**Adjustments:**

1. Added empty array validation
2. Optimized loop conditions
3. Added explicit check for non-rotated case

### Question 7: Palindrome Number

**Prompt Used:**  
"Check if number is palindrome in C#"

**Response Received:**  
Suggested reversing number and comparing.

**Implementation Details:**

* Half-number reversal optimization
* Early negative number rejection
* Handles trailing zeros properly

**Adjustments:**

1. Added x%10==0 edge case handling
2. Optimized reversal to stop at midpoint
3. Removed unnecessary variable

### Question 8: Fibonacci Number

**Prompt Used:**  
"Implement Fibonacci sequence in C# iteratively"

**Response Received:**  
Iterative solution with three variables.

**Implementation Details:**

* O(1) space complexity
* Handles base cases efficiently
* Protects against stack overflow

**Adjustments:**

1. Added negative input validation
2. Reduced to two variables
3. Improved loop structure clarity