# GitHub Copilot Interaction Documentation

ISM 6225 – Assignment 2: Computational Problem Solving

This file documents the use of GitHub Copilot as a coding assistant during the development of solutions to algorithmic problems in C#.

## Question 1: Find Missing Numbers in Array

• **Prompts Used:** How to find missing numbers from 1 to n in an array with duplicates

• **Responses Received:** Copilot suggested using in-place modification by marking visited indices as negative values to identify the missing numbers.

• **Implementation Details:** Implemented the negation trick to track seen numbers and then scanned the array to identify missing indices.

• **Adjustments Made:** Added a check for `nums.Length == 0` to gracefully handle the empty array edge case.

## Question 2: Sort Array by Parity

• **Prompts Used:** Sort array by parity in-place using two pointers

• **Responses Received**: Copilot recommended a two-pointer in-place swapping approach, swapping even and odd elements directly within the array.

• **Implementation Details**: Instead of the in-place method suggested by Copilot, I implemented a version using two separate lists—one for even numbers and one for odd numbers—and then concatenated them.

• **Adjustments Made:** Chose the two-list approach for clarity and order preservation, even though it's less space-efficient.

## Question 3: Two Sum

• **Prompts Used:** Two sum problem using dictionary

• **Responses Received:** Copilot provided a single-pass hash map solution to find the complement.

• **Implementation Details:** Applied the suggested approach directly with O(n) time complexity.

• **Adjustments Made:** Added a check to avoid overwriting existing keys using `!map.ContainsKey(nums[i])`.

## Question 4: Find Maximum Product of Three Numbers

• **Prompts Used:** Maximum product of three numbers in array

• **Responses Received:** Copilot suggested sorting the array and comparing the product of two smallest and largest vs three largest.

• **Implementation Details:** Used `Array.Sort()` and implemented both product comparisons.

• **Adjustments Made:** Checked for length < 3 and returned 0 to avoid invalid operations.

## Question 5: Decimal to Binary Conversion

• **Prompts Used:** Convert decimal to binary string

• **Responses Received:** Copilot used `Convert.ToString(decimal, 2)` for conversion.

• **Implementation Details:** Followed the suggestion and wrapped with an edge case for `0`.

• **Adjustments Made:** Explicitly returned `"0"` for input 0 instead of relying on built-in behavior.

## Question 6: Find Minimum in Rotated Sorted Array

• **Prompts Used:** Binary search for minimum in rotated sorted array

• **Responses Received:** Suggested binary search checking `nums[mid] > nums[right]` to narrow the search.

• **Implementation Details:** Implemented the exact logic using a binary search approach.

• **Adjustments Made:** No changes required; used suggested approach as-is.

## Question 7: Palindrome Number

• **Prompts Used**: Check if an integer is a palindrome

• **Responses Received:** Suggested reversing the number and comparing to the original.

• **Implementation Details:** Implemented number reversal and compared with the original value.

• **Adjustments Made:** Handled negative numbers explicitly by returning false.

## Question 8: Fibonacci Number

• **Prompts Used:** Iterative fibonacci function

• **Responses Received:** Suggested initializing two variables and looping to compute the nth number.

• **Implementation Details:** Used variables `a`, `b`, and `c` to iteratively build Fibonacci series.

• **Adjustments Made:** Checked for `n == 0` and `n == 1` separately to return early values.