Lab 05: Advanced Search Design

Step 1: By Hand, Number In list: 28, number NOT in list: 50

My List [3, 7, 15, 19, 23, 28, 34, 42, 49, 56, 62, 69, 75, 82, 88, 94, 101, 110, 118, 125]

Searching for “28” Manually:

Get middle number: 56
Is it less or greater? Greater
Then slit 0-56 in half and get number:

Get middle number: 23
Is it less or greater? Less
Then slit 23-56 in half and get number:

Get middle number: 34
Is it less or greater? Greater
Then slit 23-34 in half and get number:

Get middle number: 28
Is it less or greater? Found it!

Finding Number not in List: 50

Get middle number: 56
Is it less or greater? Less
Then split 0-56 in half and get number:

Get middle number: 23
Is it less or greater? Greater
Then split 23-56 in half and get number:

Get middle number: 42
Is it less or greater? Greater
Then split 42-56 in half and get number:

Get middle number: 49
Is it less or greater? Greater
Then split 49-56 in half and get number:

Get middle number: 49
Is it less or greater? Greater
But there’s nothing left to search—50 is not found.

Step 2: Approach

In order to solve this problem, I will use a binary search algorithm. This method works by first finding the middle element of a sorted list. If the target number is less than the middle element, I will eliminate all the numbers greater than the middle and focus only on the lower half of the list. If the target number is greater than the middle, I will eliminate all numbers smaller than the middle and focus on the upper half. I will repeat this process, each time narrowing the search range by finding the new middle, until the middle element matches the target number. If no match is found after the list has been split multiple times, it means the number is not in the list.

Step 3: Pseudocode

```
SET Low TO 0
SET High TO  length of list – 1
WHILE Low <= High DO
      SET MiddleNumberIndex =(Low + high)/2
      IF number == List[MiddleNumberIndex] THEN:
            PUT number // number has been found
      IF  number > List[middleNumberIndex] THEN:
            Set Low TO  middleNumberIndex + 1
      ELSE IF  number < List[middleNumberIndex] THEN:
            Set High TO middleNumberIndex – 1
      PUT number not found
```

Step 4: Microsoft Copilot (This is after multiple times asking, it kept just giving the code and saying its pseudocode)

```
BinarySearch(array, target)
Set left to 0
Set right to length of array - 1
     While left is less than or equal to right
          Set middle to the floor division of (left + right) by 2
          If array[middle] equals target
               Return middle
          Else if array[middle] is less than target
               Set left to middle + 1
          Else
               Set right to middle - 1
     Return -1  // target not found
```

Step 5: Compare and Contrast

1. **Pros and Cons of the Two Solutions:**

- **My Pseudocode (Step 3):**
  - **Pros:** Follows pseudocode standards, good for learning.
  - **Cons:** Doesn’t return index, lacks clear failure handling, no floor division.
- **Copilot’s Pseudocode (Step 4):**
  - **Pros:** Concise, returns index, uses floor division, handles failure.
  - **Cons:** Not pseudocode-like, lacks user-friendly output.

2. **Improvements for My Solution Based on Copilot's:**

- I could return the index instead of just printing the number.
- I should use floor division for middle index calculation.
- I can add a clear failure return (Return -1).

3. **Improvements for Copilot's Solution Based on Mine:**

- Use pseudocode keywords like SET, PUT, WHILE.

- Add output for user-friendliness.

4. **Does the Pseudocode Match the Algorithm in Step 1?**

- Yes, both versions follow the binary search algorithm described, but mine is more instructional, while Copilot's is more practical.

Step 6: Update

```
SET Low TO 0
SET High TO length of List – 1
WHILE Low <= High DO:
    SET MiddleNumberIndex TO (Low + High) // 2 // Use floor division to avoid decimals
    IF number == List[MiddleNumberIndex] THEN:
        RETURN MiddleNumberIndex // Return the index where the number is found
    IF number > List[MiddleNumberIndex] THEN:
        SET Low TO MiddleNumberIndex + 1 // Search in the right half
    ELSE:
        SET High TO MiddleNumberIndex - 1 // Search in the left half
WRITE ERROR “NOT FOUND” // If the number is not found
RETURN -1 // If the number is not found
```

Step 7:

Step 1 By Hand:25 minutes

Step 2 Approach: 30 minutes

Step 3 Pseudocode: 100 minutes

Step 4 Copilot: 20 minutes

Step 5 Compare and Contrast: 40 minutes

Step 6 Update: 25 minutes