Activity – Discussion Points

1. Real-world use of searching algorithms

- Phone contacts – Linear search because the list may not be sorted.
- Dictionary or database search – Binary search because the data is sorted.
- File systems – Tree search because files are stored hierarchically.
- Navigation apps – Graph search to find the best path or route.

2. Searching in other data structures

- Linked lists use linear search since there is no direct access to elements.
- Trees use tree searches like BST search, DFS, or BFS.
- Changes needed include using pointers instead of indexes and keeping the structure organized.

3. Stable sorting algorithm

- A sorting algorithm is **stable** if it keeps the order of equal elements.
- Stability is important when sorting data with multiple attributes.

4. Other factors when choosing a sorting algorithm

- Stability
- Memory usage
- Simplicity
- Type and size of data
- Performance on nearly sorted data

Questions

Given array

{1, 4, 6, 7, 9, 10, 14}

1a. Binary search (finding 9)

- Comparisons needed: **3**

1b. Sequential search (finding 9)

- Comparisons needed: **5**

1c. Sequential search (checking if 11 is not in the array)

- Comparisons needed: **7**

2. Sequential Search Code

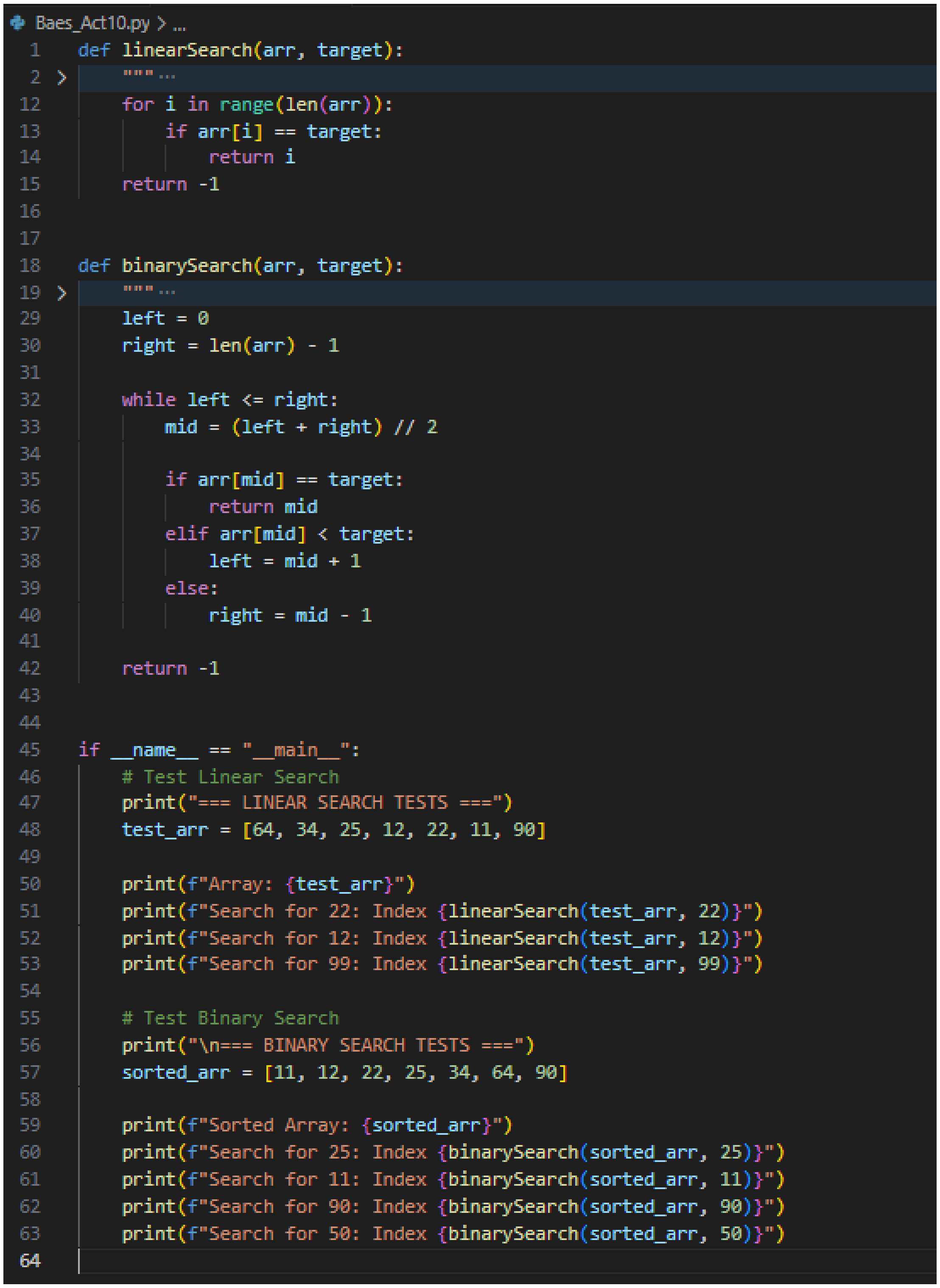
```
public static boolean seqSearch(int s) {
    int[] a = {9, 12, 14, 3, 25};

    for (int i = 0; i < a.length; i++) {
        if (a[i] == s) {
            return true;
        }
    }
    return false;
}
```

3. Draw the Array after each iteration

5	3	8	1
1	3	8	5
1	3	8	5
1	3	5	8

1	3	5	8

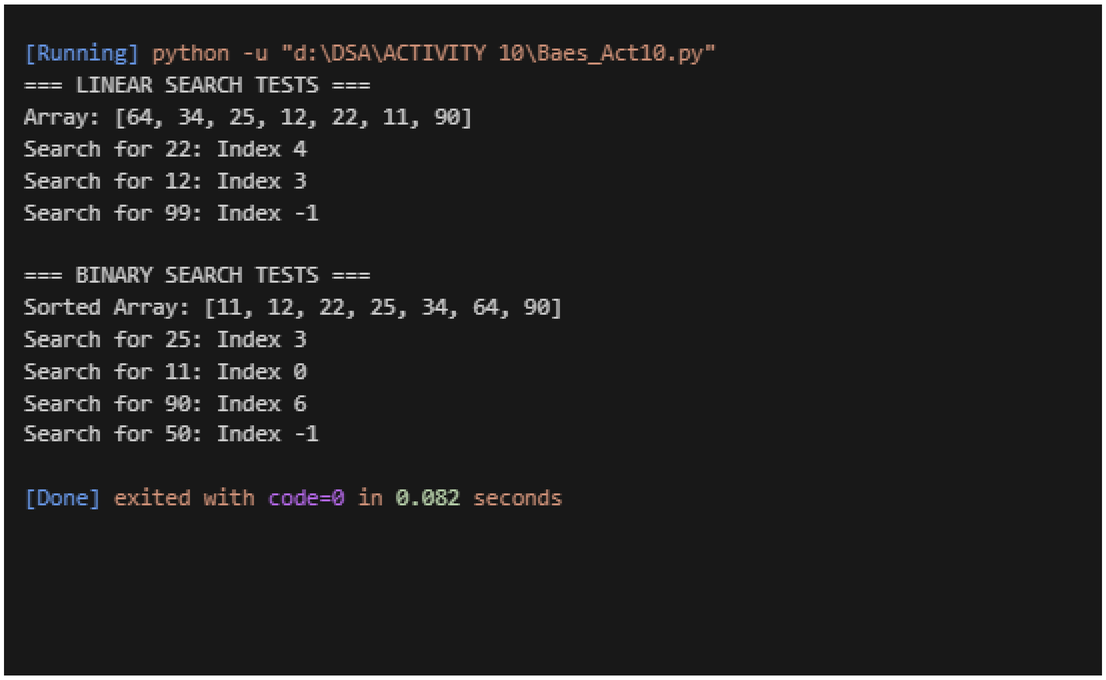
Programming Project



Baes_Act10.py > ...

```python
def linearSearch(arr, target):
    """...
    for i in range(len(arr)):
        if arr[i] == target:
            return i
    return -1


def binarySearch(arr, target):
    """...
    left = 0
    right = len(arr) - 1

    while left <= right:
        mid = (left + right) // 2

        if arr[mid] == target:
            return mid
        elif arr[mid] < target:
            left = mid + 1
        else:
            right = mid - 1

    return -1


if __name__ == "__main__":
    # Test Linear Search
    print("=== LINEAR SEARCH TESTS ===")
    test_arr = [64, 34, 25, 12, 22, 11, 90]

    print(f"Array: {test_arr}")
    print(f"Search for 22: Index {linearSearch(test_arr, 22)}")
    print(f"Search for 12: Index {linearSearch(test_arr, 12)}")
    print(f"Search for 99: Index {linearSearch(test_arr, 99)}")

    # Test Binary Search
    print("\n=== BINARY SEARCH TESTS ===")
    sorted_arr = [11, 12, 22, 25, 34, 64, 90]

    print(f"Sorted Array: {sorted_arr}")
    print(f"Search for 25: Index {binarySearch(sorted_arr, 25)}")
    print(f"Search for 11: Index {binarySearch(sorted_arr, 11)}")
    print(f"Search for 90: Index {binarySearch(sorted_arr, 90)}")
    print(f"Search for 50: Index {binarySearch(sorted_arr, 50)}")
```

OUTPUT



```
[Running] python -u "d:\DSA\ACTIVITY 10\Baes_Act10.py"
=== LINEAR SEARCH TESTS ===
Array: [64, 34, 25, 12, 22, 11, 90]
Search for 22: Index 4
Search for 12: Index 3
Search for 99: Index -1

=== BINARY SEARCH TESTS ===
Sorted Array: [11, 12, 22, 25, 34, 64, 90]
Search for 25: Index 3
Search for 11: Index 0
Search for 90: Index 6
Search for 50: Index -1

[Done] exited with code=0 in 0.082 seconds
```