EXPERIMENT 1.4

Student Name: Aayush Kumar. **UID:** 21BCS6958

Branch: CSE **Section/Group:** IOT-632/B

Semester: 5th **Date of Performance:** 12-09-23

Subject Name: Advance Programming Lab **Subject Code:** 21CSP-314

1. **(a) Aim: -** Implement the concept of Searching and Sorting techniques.

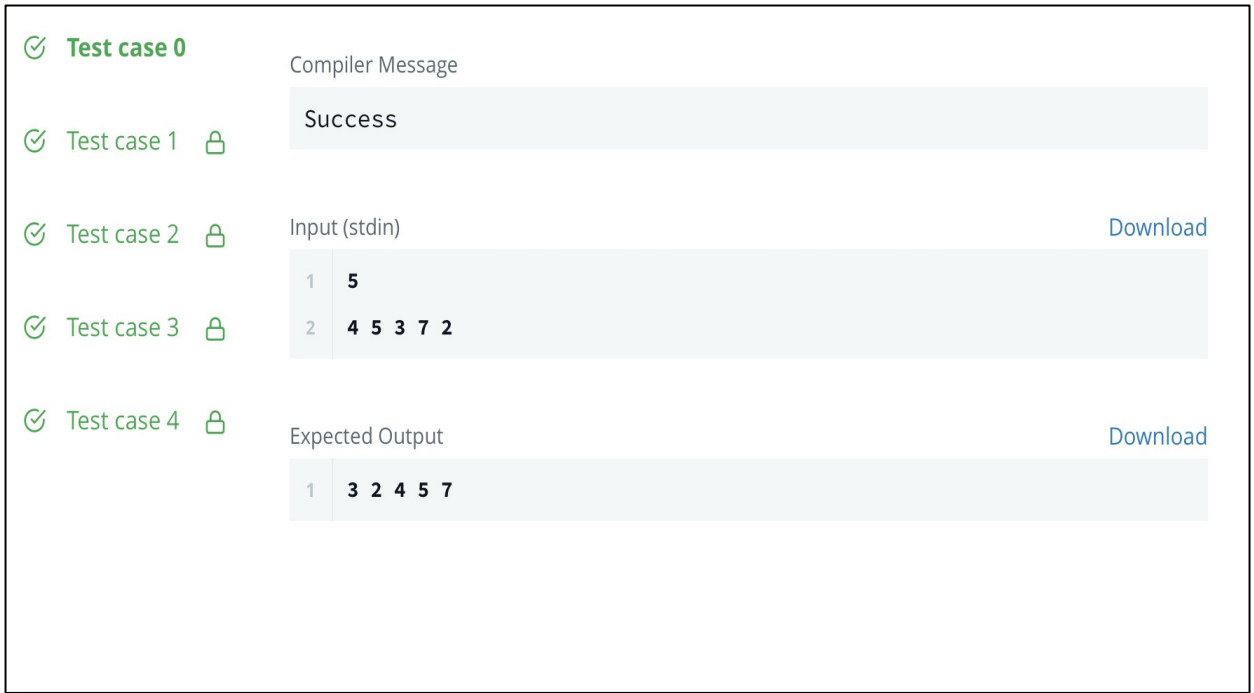
2. **(a) Objective: -** To implement the Quick Sort algorithm to sort an integer array (‘**arr’**) and write the sorted array to the output file specified by the **'OUTPUT_PATH'** environment variable.

3. **(a) Code and Output : -**

```
import math
import os
import random
import re
import sys

def quickSort(arr):
    if len(arr) <= 1:
        return arr

    pivot = arr[0]  # Choose the first element as the pivot
    left = []
    right = []

    # Partition the array into elements less than pivot and elements greater than pivot
    for element in arr[1:]:
        if element <= pivot:
            left.append(element)
        else:
            right.append(element)

    # Recursively sort the left and right partitions
```

```
    left = quickSort(left)
    right = quickSort(right)

    # Concatenate the sorted left partition, pivot, and sorted right partition
    sorted_arr = left + [pivot] + right

    return sorted_arr

if __name__ == '__main__':
    fptr = open(os.environ['OUTPUT_PATH'], 'w')

    n = int(input().strip())

    arr = list(map(int, input().rstrip().split()))

    result = quickSort(arr)

    fptr.write(' '.join(map(str, result)))
    fptr.write('\n')

    fptr.close()
```

Output: -



1. **(b) Aim: -** Implement the concept of Searching and Sorting techniques.

2. **(b) Objective: -** To find and return missing numbers between two integer arrays (‘arr’ and ‘brr’) while ensuring the output is written to the specified output file determined by the 'OUTPUT_PATH' environment variable.

3. **(b) Code and Output : -**

```
import math
import os
import random
import re
import sys

def missingNumbers(arr, brr):
    # Create dictionaries to store the frequency of elements in both arrays
    freq_arr = {}
    freq_brr = {}

    # Populate freq_arr with the frequencies of elements in arr
    for num in arr:
        freq_arr[num] = freq_arr.get(num, 0) + 1

    # Populate freq_brr with the frequencies of elements in brr
    for num in brr:
        freq_brr[num] = freq_brr.get(num, 0) + 1

    # Initialize a list to store missing numbers
    missing = []

    # Compare the frequencies of elements in arr and brr
    for num, count_brr in freq_brr.items():
        count_arr = freq_arr.get(num, 0)
        if count_brr != count_arr:
            missing.append(num)

    # Sort the missing numbers in ascending order
```

```
    missing.sort()

    return missing

if __name__ == '__main__':
    fptr = open(os.environ['OUTPUT_PATH'], 'w')

    n = int(input().strip())

    arr = list(map(int, input().rstrip().split()))

    m = int(input().strip())

    brr = list(map(int, input().rstrip().split()))

    result = missingNumbers(arr, brr)

    fptr.write(' '.join(map(str, result)))
    fptr.write('\n')

    fptr.close()
```

Output: -