Date: 18-11-2024

CODING PRACTICE PROBLEMS – DAY 6

1. Bubble Sort

```
class Solution {
    public static void bubbleSort(int arr[]) {
        int n = arr.length;
        for (int i = 0; i < n - 1; i++) {
            for (int j = 0; j < n - i - 1; j++) {
                if (arr[j] > arr[j + 1]) {
                    int temp = arr[j];
                    arr[j] = arr[j + 1];
                    arr[j + 1] = temp;
                }
            }
        }
    }
}
```

Time Complexity: O(N^2)

OUTPUT:

Output Window

Compilation Results Custom Input

Compilation Completed

For Input:

4 1 3 9 7

Your Output:

1 3 4 7 9

Expected Output:

1 3 4 7 9

2. Quick Sort

```
class Solution {
    static void quickSort(int arr[], int low, int high) {
        if (low < high) {
            int pivotIndex = partition(arr, low, high);
            quickSort(arr, low, pivotIndex - 1);
            quickSort(arr, pivotIndex + 1, high);
        }
    }
    static int partition(int arr[], int low, int high) {
        int pivot = arr[high];
        int i = low - 1;
        for (int j = low; j < high; j++) {
            if (arr[j] <= pivot) {
                i++;
                int temp = arr[i];
                arr[i] = arr[j];
                arr[j] = temp;
            }
        }
        int temp = arr[i + 1];
        arr[i + 1] = arr[high];
        arr[high] = temp;

        return i + 1;
    }
}
```

Time Complexity: O(n log n)

OUTPUT:

Output Window

Compilation Results Custom Input

Compilation Completed

For Input:

```
4 1 3 9 7
```

Your Output:

```
1 3 4 7 9
```

Expected Output:

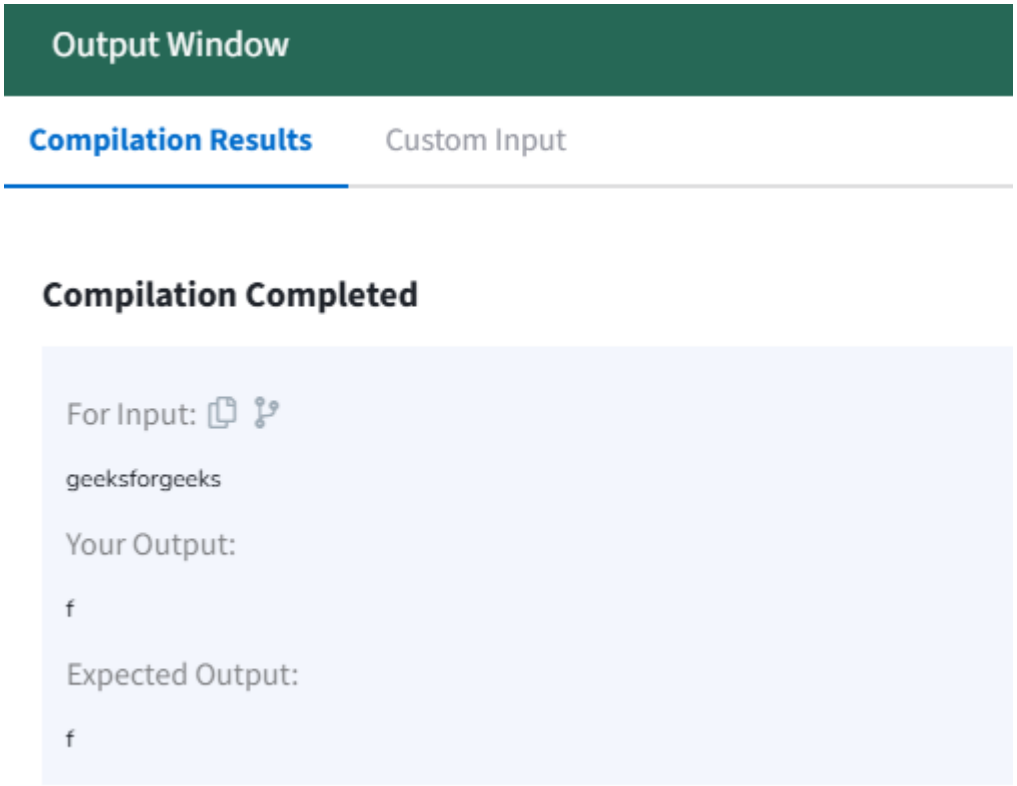
```
1 3 4 7 9
```

3. Non Repeating Character

```java
class Solution {
    static char nonRepeatingChar(String s) {
        HashMap<Character, Integer> frequencyMap = new HashMap<>();
        for (char ch : s.toCharArray()) {
            frequencyMap.put(ch, frequencyMap.getOrDefault(ch, 0) + 1);
        }
        for (char ch : s.toCharArray()) {
            if (frequencyMap.get(ch) == 1) {
                return ch;
            }
        }
        return '$';
    }
}
```

Time Complexity: O(n)

OUTPUT:



4. Edit Distance

```
class Solution {
    public int editDistance(String s1, String s2) {
        int m = s1.length();
        int n = s2.length();

        int[][] dp = new int[m + 1][n + 1];

        for (int i = 0; i <= m; i++) {
            for (int j = 0; j <= n; j++) {
                if (i == 0) {
                    dp[i][j] = j;
                } else if (j == 0) {
                    dp[i][j] = i;
                } else if (s1.charAt(i - 1) == s2.charAt(j - 1)) {
                    dp[i][j] = dp[i - 1][j - 1];
                } else {
                    dp[i][j] = 1 + Math.min(dp[i - 1][j - 1], Math.min(dp[i
- 1][j], dp[i][j - 1]));
                }
            }
        }
        return dp[m][n];
    }
}
```

Time Complexity: O(n)

OUTPUT:

Output Window

Compilation Results Custom Input

Compilation Completed

For Input:

geek

gesek

Your Output:

1

Expected Output:

1

5. k largest elements

```
class Solution {
    static List<Integer> kLargest(int arr[], int k) {
        Arrays.sort(arr);
        List<Integer> result = new ArrayList<>();

        for (int i = arr.length - 1; i >= arr.length - k; i--) {
            result.add(arr[i]);
        }

        return result;
    }
}
```

Time Complexity: O(n log n)

OUTPUT:

Output Window

Compilation Results Custom Input Y.O.G.I. (AI Bot)

Compilation Completed

For Input:

12 5 787 1 23

2

Your Output:

787 23

Expected Output:

787 23

6. Form the Largest Number

```
class Solution {
    String printLargest(int[] arr) {
        String[] strArr = new String[arr.length];
        for (int i = 0; i < arr.length; i++) {
            strArr[i] = String.valueOf(arr[i]);
        }
        Arrays.sort(strArr, new Comparator<String>() {
            public int compare(String x, String y) {
```

```
                String xy = x + y;
                String yx = y + x;
                return yx.compareTo(xy);
            }
        });
        if (strArr[0].equals("0")) {
            return "0";
        }
        StringBuilder result = new StringBuilder();
        for (String str : strArr) {
            result.append(str);
        }
        return result.toString();
    }
}
```

Time Complexity: O(n log n)

OUTPUT:

Output Window

Compilation Results Custom Input Y.O.G.I. (AI Bot)

Compilation Completed

For Input:

4 5 7 15 20 11

Your Output:

754201511

Expected Output:

754201511