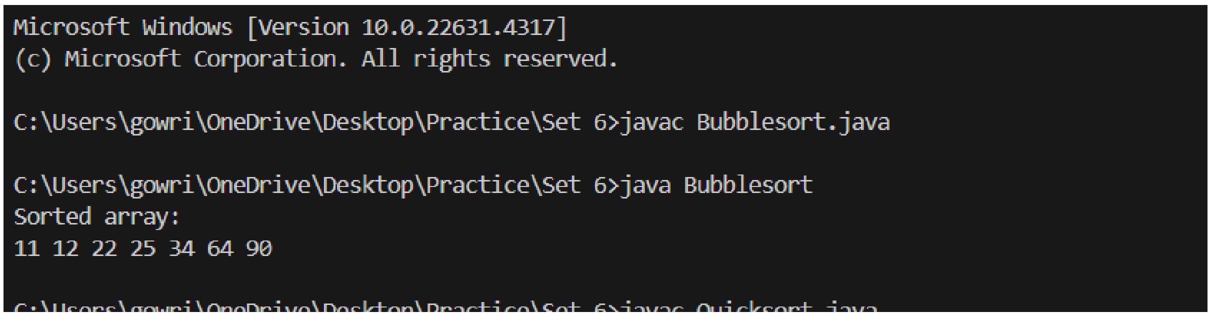
Maha Gowri S **18/11/2024**

1.Bubble Sort

Code:

```
import java.io.*;
class Bubblesort {
    static void bubbleSort(int arr[],int n){
        int i,j,temp;
        boolean swapped;
        for  (i = 0;i<n-1;i++){
            swapped = false;
            for (j=0;j<n-i-1;j++){
              if (arr[j] >arr[j+1]){
                temp = arr[j];
                arr[j] = arr[j + 1];
                arr[j + 1] = temp;
                swapped = true;
              }
            }
            if (swapped == false)
                break;
        }
    }
    static void printArray(int arr[], int size){
        int i;
        for (i = 0; i < size; i++)
            System.out.print(arr[i] + " ");
        System.out.println();
    }
    public static void main(String args[]){
        int arr[] = { 64, 34, 25, 12, 22, 11, 90 };
        int n = arr.length;
        bubbleSort(arr, n);
        System.out.println("Sorted array: ");
        printArray(arr, n);
    }
}
```

Output:



```
Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. All rights reserved.

C:\Users\gowri\OneDrive\Desktop\Practice\Set 6>javac Bubblesort.java

C:\Users\gowri\OneDrive\Desktop\Practice\Set 6>java Bubblesort
Sorted array:
11 12 22 25 34 64 90

C:\Users\gowri\OneDrive\Desktop\Practice\Set 6>javac Quicksort.java
```

Time Complexity:O(N^2)

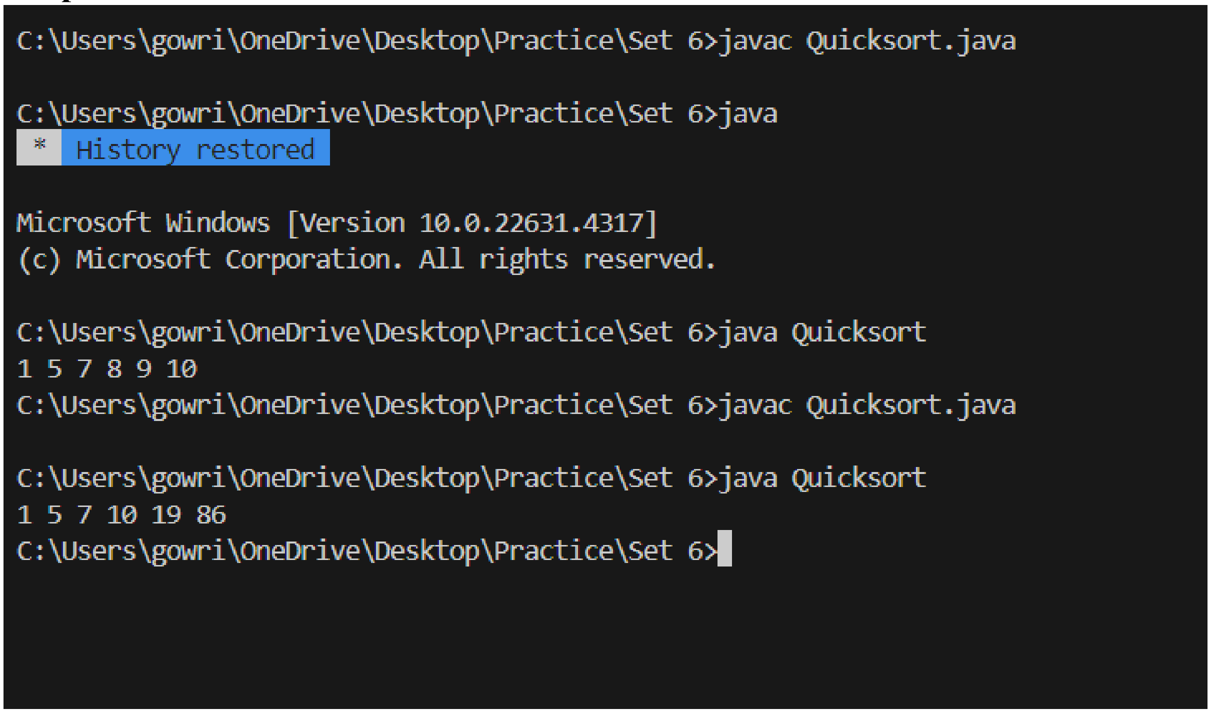
2.Quick Sort

Code:

```
import java.util.Arrays;


class Quicksort {


    static int partition(int[] arr, int low, int high) {


        int pivot = arr[high];


        int i = low - 1;


        for (int j = low; j <= high - 1; j++) {
            if (arr[j] < pivot) {
                i++;
                swap(arr, i, j);
            }
        }
        swap(arr, i + 1, high);
        return i + 1;
    }


    static void swap(int[] arr, int i, int j) {
        int temp = arr[i];
        arr[i] = arr[j];
        arr[j] = temp;
    }

    static void quickSort(int[] arr, int low, int high) {
        if (low < high) {

            int pi = partition(arr, low, high);

            quickSort(arr, low, pi - 1);
            quickSort(arr, pi + 1, high);
        }
    }

    public static void main(String[] args) {
        int[] arr = {10, 7, 8, 9, 1, 5};
        int n = arr.length;

        quickSort(arr, 0, n - 1);
```

```
        for (int val : arr) {
            System.out.print(val + " ");
        }
    }
}
```

Output:



```
C:\Users\gowri\OneDrive\Desktop\Practice\Set 6>javac Quicksort.java

C:\Users\gowri\OneDrive\Desktop\Practice\Set 6>java
 *  History restored

Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. All rights reserved.

C:\Users\gowri\OneDrive\Desktop\Practice\Set 6>java Quicksort
1 5 7 8 9 10
C:\Users\gowri\OneDrive\Desktop\Practice\Set 6>javac Quicksort.java

C:\Users\gowri\OneDrive\Desktop\Practice\Set 6>java Quicksort
1 5 7 10 19 86
C:\Users\gowri\OneDrive\Desktop\Practice\Set 6>
```

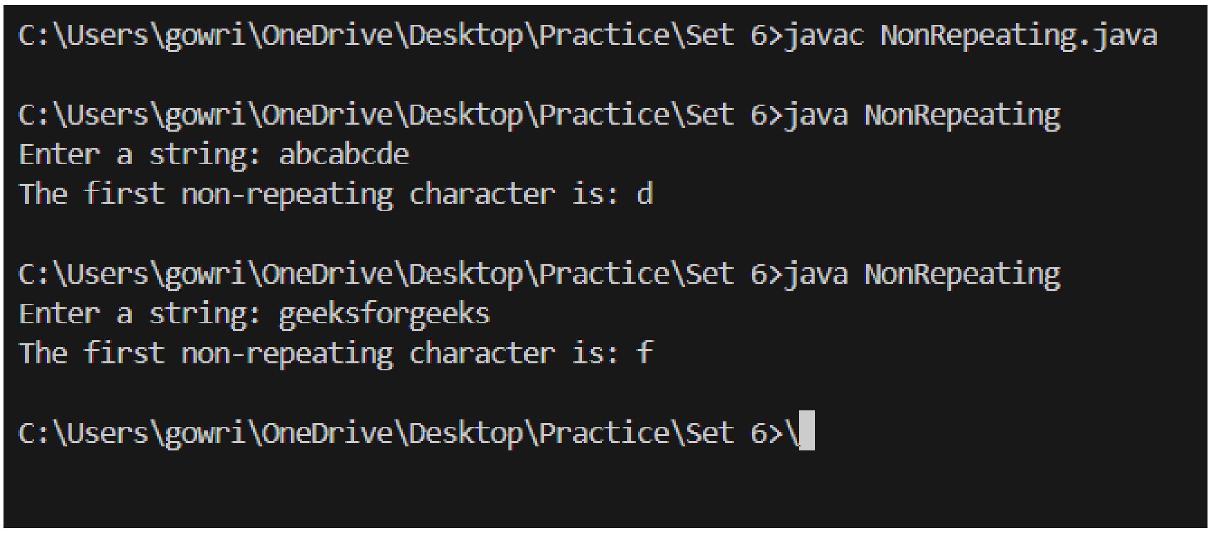
Time Complexity:O(n log n)

3.Non-Repeating Character

Code:

```
import java.util.*;


class NonRepeating {


    static final int MAX_CHAR = 26;


    static char nonRepeatingChar(String s) {
        int[] vis = new int[MAX_CHAR];
        Arrays.fill(vis, -1);

        for (int i = 0; i < s.length(); i++) {
            if (vis[s.charAt(i) - 'a'] == -1)
                vis[s.charAt(i) - 'a'] = i;
            else
                vis[s.charAt(i) - 'a'] = -2;
        }

        int idx = Integer.MAX_VALUE;

        for (int i = 0; i < MAX_CHAR; i++) {
            if (vis[i] >= 0)
                idx = Math.min(idx, vis[i]);
        }

        return (idx == Integer.MAX_VALUE ? '$' : s.charAt(idx));
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter a string: ");
        String s = sc.nextLine();
        char result = nonRepeatingChar(s);

        if (result == '$') {
            System.out.println("No non-repeating character found.");
        } else {
            System.out.println("The first non-repeating character is: " + result);
        }


        sc.close();
    }
}
```

Output:



```
C:\Users\gowri\OneDrive\Desktop\Practice\Set 6>javac NonRepeating.java

C:\Users\gowri\OneDrive\Desktop\Practice\Set 6>java NonRepeating
Enter a string: abcabcde
The first non-repeating character is: d

C:\Users\gowri\OneDrive\Desktop\Practice\Set 6>java NonRepeating
Enter a string: geeksforgeeks
The first non-repeating character is: f

C:\Users\gowri\OneDrive\Desktop\Practice\Set 6>\
```

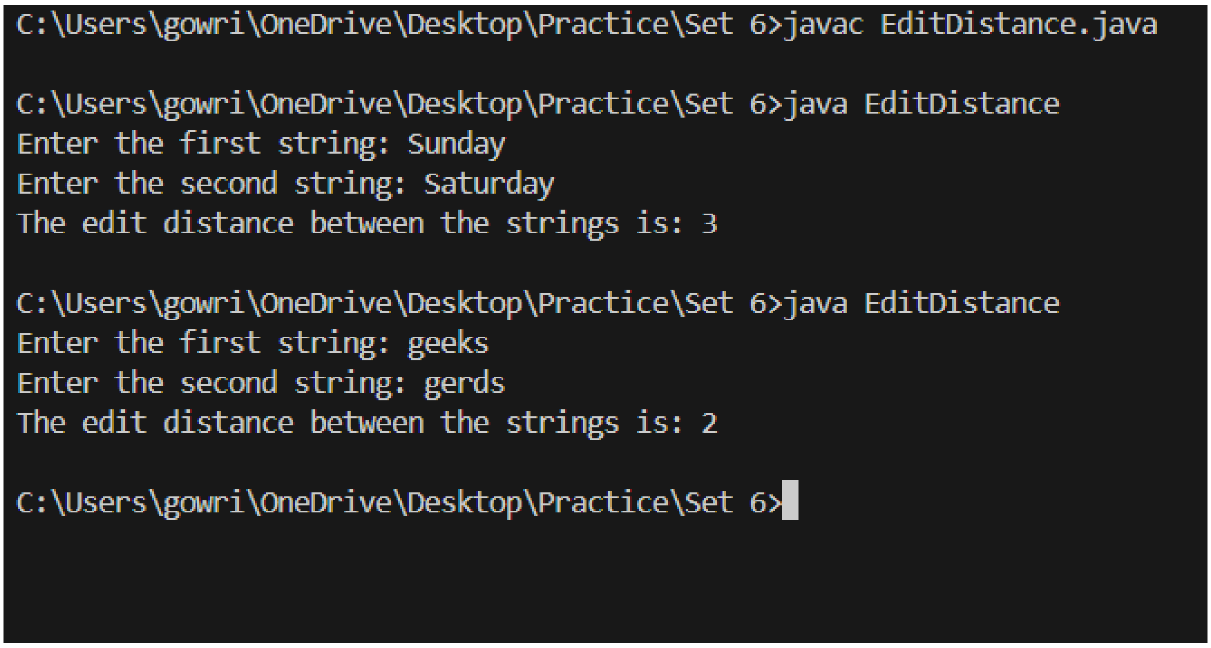
Time Complexity:O(N)

3.Edit Distance

Code:

```
import java.util.Scanner;


public class EditDistance {


    public static int editDistDP(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++)
            dp[i][0] = i;
        for (int j = 0; j <= n; j++)
            dp[0][j] = j;

        for (int i = 1; i <= m; i++) {
            for (int j = 1; j <= n; j++) {
                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][j - 1], Math.min(dp[i - 1][j], dp[i - 1][j -
1]));
            }
        }

        return dp[m][n];
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter the first string: ");
        String s1 = sc.nextLine();

        System.out.print("Enter the second string: ");
        String s2 = sc.nextLine();

        System.out.println("The edit distance between the strings is: " + editDistDP(s1,
s2));

        sc.close();
    }
}
```

Output:



```
C:\Users\gowri\OneDrive\Desktop\Practice\Set 6>javac EditDistance.java

C:\Users\gowri\OneDrive\Desktop\Practice\Set 6>java EditDistance
Enter the first string: Sunday
Enter the second string: Saturday
The edit distance between the strings is: 3

C:\Users\gowri\OneDrive\Desktop\Practice\Set 6>java EditDistance
Enter the first string: geeks
Enter the second string: gerds
The edit distance between the strings is: 2

C:\Users\gowri\OneDrive\Desktop\Practice\Set 6>
```

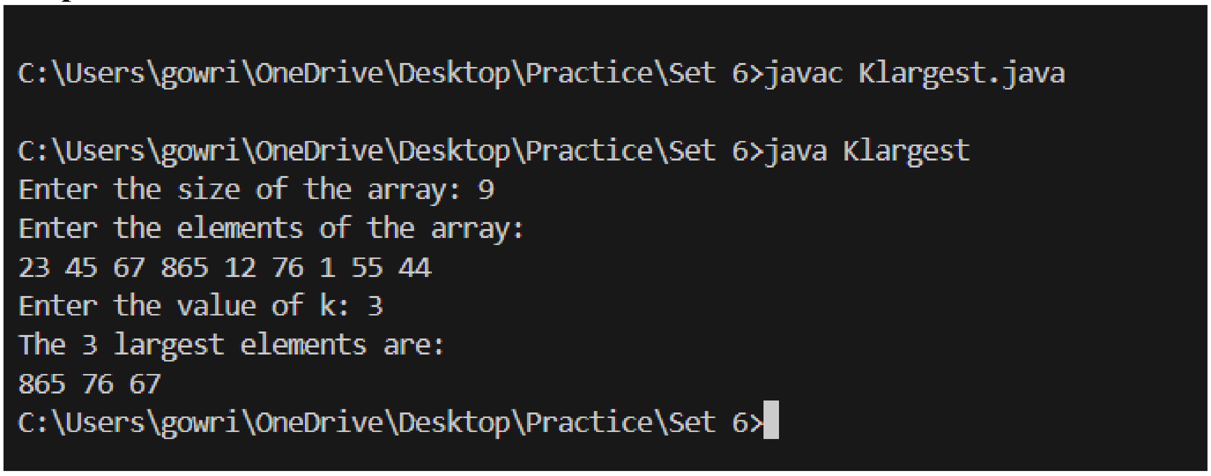
Time Complexity : O(M*N)

4.K largest

Code:

```
import java.util.*;


class Klargest {
    static ArrayList<Integer> kLargest(int[] arr, int k) {
        int n = arr.length;
        Integer[] arrInteger = Arrays.stream(arr).boxed().toArray(Integer[]::new);
        Arrays.sort(arrInteger, Collections.reverseOrder());
        ArrayList<Integer> res = new ArrayList<>();
        for (int i = 0; i < k; i++)
            res.add(arrInteger[i]);
        return res;
    }
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter the size of the array: ");
        int n = sc.nextInt();

        int[] arr = new int[n];
        System.out.println("Enter the elements of the array:");
        for (int i = 0; i < n; i++)
            arr[i] = sc.nextInt();

        System.out.print("Enter the value of k: ");
        int k = sc.nextInt();

        ArrayList<Integer> res = kLargest(arr, k);
        System.out.println("The " + k + " largest elements are:");
        for (int ele : res)
            System.out.print(ele + " ");

        sc.close();
    }
}
```

Output:



```
C:\Users\gowri\OneDrive\Desktop\Practice\Set 6>javac Klargest.java

C:\Users\gowri\OneDrive\Desktop\Practice\Set 6>java Klargest
Enter the size of the array: 9
Enter the elements of the array:
23 45 67 865 12 76 1 55 44
Enter the value of k: 3
The 3 largest elements are:
865 76 67
C:\Users\gowri\OneDrive\Desktop\Practice\Set 6>
```

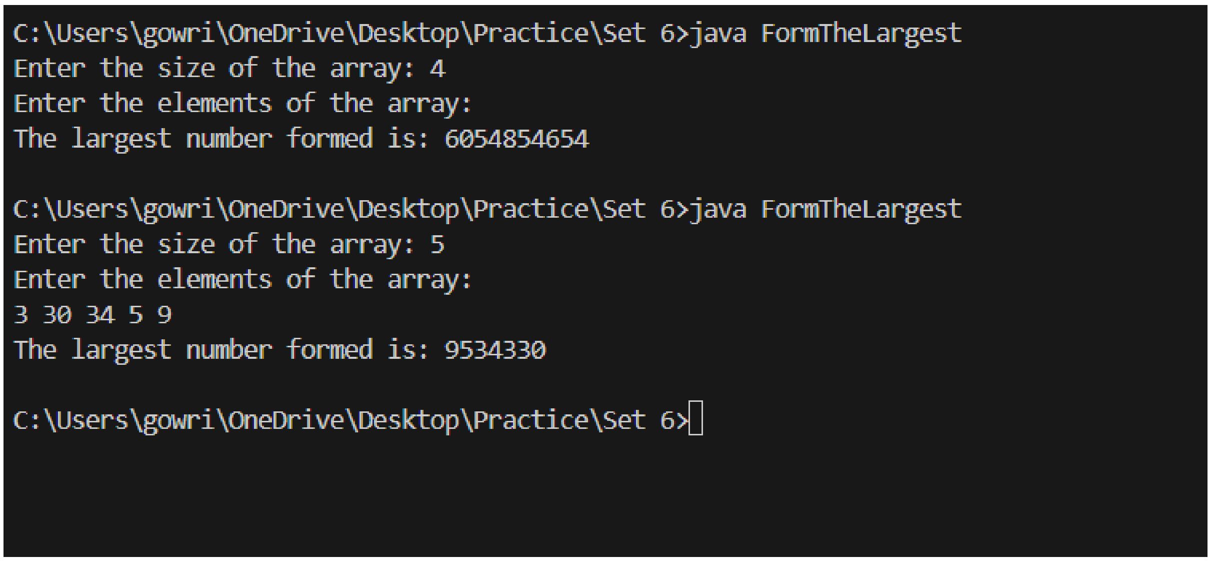
Time Complexity:O(n log n)

6.From the largest Number

Code:

```
import java.util.*;


class FormTheLargest {


    static boolean myCompare(String s1, String s2) {
        return (s1 + s2).compareTo(s2 + s1) > 0;
    }


    static String findLargest(int[] arr) {
        ArrayList<String> numbers = new ArrayList<>();
        for (int ele : arr) {
            numbers.add(Integer.toString(ele));
        }


        Collections.sort(numbers, (s1, s2) -> myCompare(s1, s2) ? -1 : 1);


        if (numbers.get(0).equals("0")) {
            return "0";
        }

        StringBuilder res = new StringBuilder();
        for (String num : numbers) {
            res.append(num);
        }

        return res.toString();
    }

    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);

        System.out.print("Enter the size of the array: ");
        int n = sc.nextInt();

        int[] arr = new int[n];
        System.out.println("Enter the elements of the array:");
        for (int i = 0; i < n; i++) {
            arr[i] = sc.nextInt();
        }

        System.out.println("The largest number formed is: " + findLargest(arr));

        sc.close();
    }
}
```

Output:



```
C:\Users\gowri\OneDrive\Desktop\Practice\Set 6>java FormTheLargest
Enter the size of the array: 4
Enter the elements of the array:
The largest number formed is: 6054854654

C:\Users\gowri\OneDrive\Desktop\Practice\Set 6>java FormTheLargest
Enter the size of the array: 5
Enter the elements of the array:
3 30 34 5 9
The largest number formed is: 9534330

C:\Users\gowri\OneDrive\Desktop\Practice\Set 6>
```

Time Complexity:O(N*log N)