

VIJAY M

22IT125

DAY 5

CODING PRACTICES AND PROBLEM

1.Bubble Sort

Time Complexity : $O(n^2)$

Solution :

```
import java.io.*;
```

```
class GFG {
```

```
    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:

The screenshot shows an IDE window with a project named 'TRIPILET'. The source file 'FindMaximum.java' is open, displaying the following Java code:

```

1
2 import java.io.*;
3
4 class GFG {
5
6     static void bubbleSort(int arr[], int n){
7         int i, j, temp;
8         boolean swapped;
9         for (i = 0; i < n - 1; i++) {
10             swapped = false;
11             for (j = 0; j < n - i - 1; j++) {
12                 if (arr[j] > arr[j + 1]) {
13                     temp = arr[j];
14                     arr[j] = arr[j + 1];
15                     arr[j + 1] = temp;
16                     swapped = true;
17                 }
18             }
19         }
20     }

```

The Run console at the bottom shows the execution output:

```

C:\Users\sanja\.jdk\openjdk-23.0.1\bin\java.exe "-javaagent:C:\Users\sanja\AppData\Local\JetBrains\IntelliJ IDEA 2024.2.4\lib\idea_rt.jar=61603:C:\Users\sanja\AppData\Local\
Sorted array:
11 12 22 25 34 64 90
Process finished with exit code 0

```

The status bar at the bottom indicates the current file is 'bubbleSort' and the encoding is 'UTF-8'.

2.Quick Sort

Time Complexity:

Best case : ($\Omega(n \log n)$)

Average Case : ($\theta(n \log n)$)

Worst Case : ($O(n^2)$)

Solution:

```
import java.util.Arrays;
class GfG {
    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:

```

class GfG {
    static int partition(int[] arr, int low, int high) {
        // ...
    }

    static void swap(int[] arr, int i, int j) {
        // ...
    }

    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);
        }
    }
}

```

Run GfG

```

C:\Users\sanja\jdk\openjdk-23.0.1\bin\java.exe --javaagent:C:\Users\sanja\AppData\Local\JetBrains\IntelliJ IDEA 2024.2.4\lib\idea_rt.jar=62037:C:\Users\sanja\AppData\Local\J...
1 5 7 8 9 10
Process finished with exit code 0

```

TRIPLET > src > GfG > partition 21:6 LF UTF-8 4 spaces

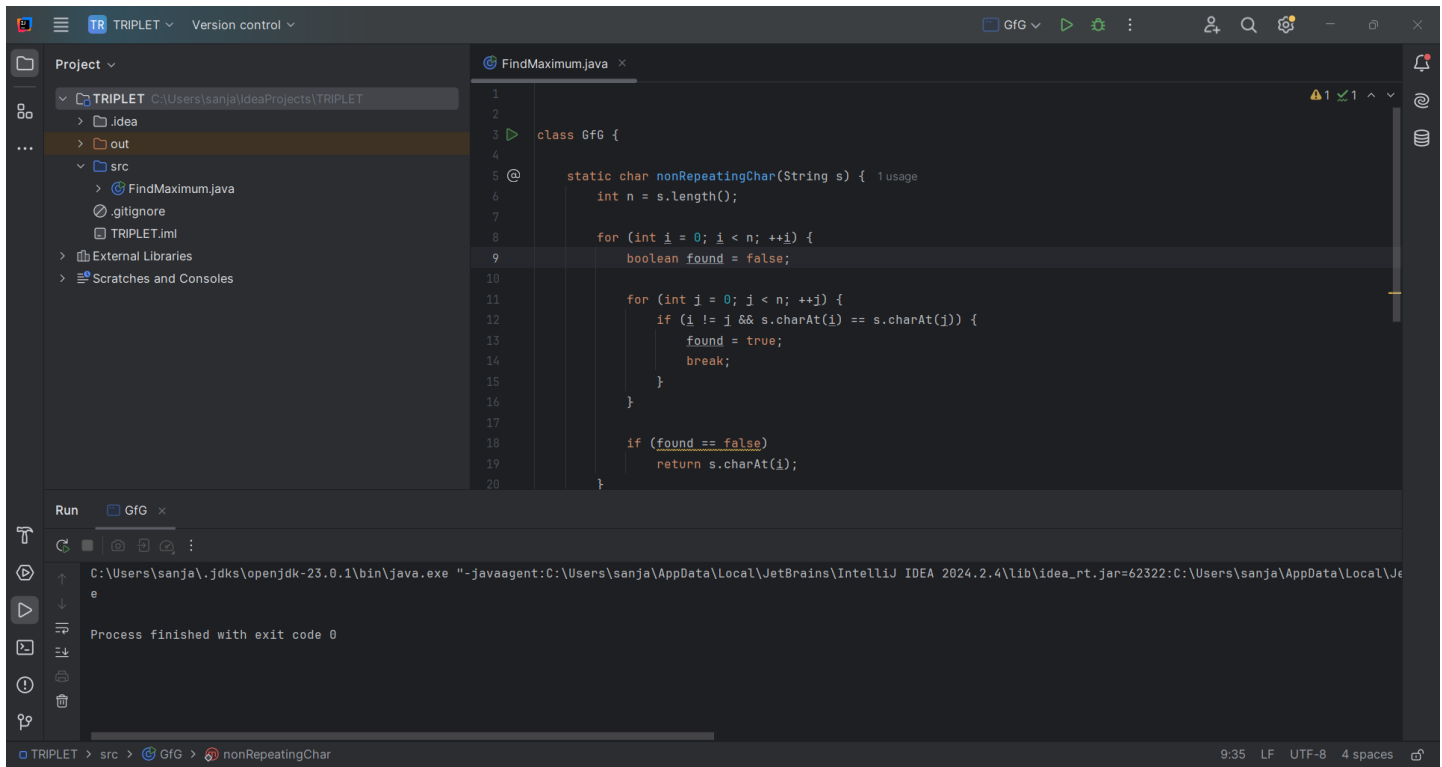
3.Non repeating Character

Time Complexity : $O(n^2)$

Solution:

```
class GfG {  
  
    static char nonRepeatingChar(String s) {  
        int n = s.length();  
  
        for (int i = 0; i < n; ++i) {  
            boolean found = false;  
  
            for (int j = 0; j < n; ++j) {  
                if (i != j && s.charAt(i) == s.charAt(j)) {  
                    found = true;  
                    break;  
                }  
            }  
  
            if (found == false)  
                return s.charAt(i);  
        }  
  
        return '$';  
    }  
  
    public static void main(String[] args) {  
        String s = "racecar";  
  
        System.out.println(nonRepeatingChar(s));  
    }  
}
```

Output:



The screenshot shows an IDE window with a project named 'TRIPLET'. The file explorer on the left shows the project structure with a 'src' folder containing 'FindMaximum.java'. The main editor displays the code for 'FindMaximum.java', which is a class 'GfG' with a static method 'nonRepeatingChar'. The method takes a string 's' and returns the first non-repeating character. The code uses two nested loops: the outer loop iterates over each character 'i' in the string, and the inner loop iterates over all other characters 'j' to check if 's.charAt(i)' equals 's.charAt(j)'. If a duplicate is found, 'found' is set to true and the loop breaks. If no duplicate is found for a character, it is returned. The bottom panel shows the 'Run' tab with the command 'C:\Users\sanja\jdk\openjdk-23.0.1\bin\java.exe' and the output 'Process finished with exit code 0'.

```
1
2
3 class GfG {
4
5     static char nonRepeatingChar(String s) {
6         int n = s.length();
7
8         for (int i = 0; i < n; ++i) {
9             boolean found = false;
10
11             for (int j = 0; j < n; ++j) {
12                 if (i != j && s.charAt(i) == s.charAt(j)) {
13                     found = true;
14                     break;
15                 }
16             }
17
18             if (found == false)
19                 return s.charAt(i);
20         }
21     }
22 }
```

4.Edit Distance

Time Complexity : $O(m \times n)$

Solution:

```
public class GfG {
```

```
    public static int editDistRec(String s1, String s2, int m, int n) {
```

```
        if (m == 0) return n;
```

```
        if (n == 0) return m;
```

```
        if (s1.charAt(m - 1) == s2.charAt(n - 1))
            return editDistRec(s1, s2, m - 1, n - 1);
```

```
        return 1 + Math.min(Math.min(editDistRec(s1, s2, m, n - 1),
```

```

        editDistRec(s1, s2, m - 1, n)),
        editDistRec(s1, s2, m - 1, n - 1));
    }
}

```

```

public static int editDist(String s1, String s2) {
    return editDistRec(s1, s2, s1.length(), s2.length());
}

```

```

public static void main(String[] args) {
    String s1 = "GEEXSFRGEEKKS";
    String s2 = "GEEKSFORGEEKS";
    System.out.println(editDist(s1, s2));
}
}

```

Output:

The screenshot shows an IDE with a project named 'TRIPLET'. The source file 'GfG.java' contains the following code:

```

1 public class GfG {
2
3     public static int editDistRec(String s1, String s2, int m, int n) {
4
5         if (m == 0) return n;
6
7         if (n == 0) return m;
8
9         if (s1.charAt(m - 1) == s2.charAt(n - 1))
10            return editDistRec(s1, s2, m - 1, n - 1);
11
12        return 1 + Math.min(Math.min(editDistRec(s1, s2, m, n - 1),
13            editDistRec(s1, s2, m - 1, n)),
14            editDistRec(s1, s2, m - 1, n - 1));
15    }
16
17    public static int editDist(String s1, String s2) { return editDistRec(s1, s2, s1.length(), s2.length()); }
18
19
20
21    public static void main(String[] args) {
22        String s1 = "GEEXSFRGEEKKS";

```

The Run tab shows the command executed: `C:\Users\sanja\.jdk\openjdk-23.0.1\bin\java.exe "-javaagent:C:\Users\sanja\AppData\Local\JetBrains\IntelliJ IDEA 2024.2.4\lib\idea_rt.jar=62670:C:\Users\sanja\AppData\Local\J...` and the output: `3`. The process finished with exit code 0.

5.K Largest elements

Time Complexity : $O(n * \log(n))$

Solution:

```
import java.util.*;

class GfG {
    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());

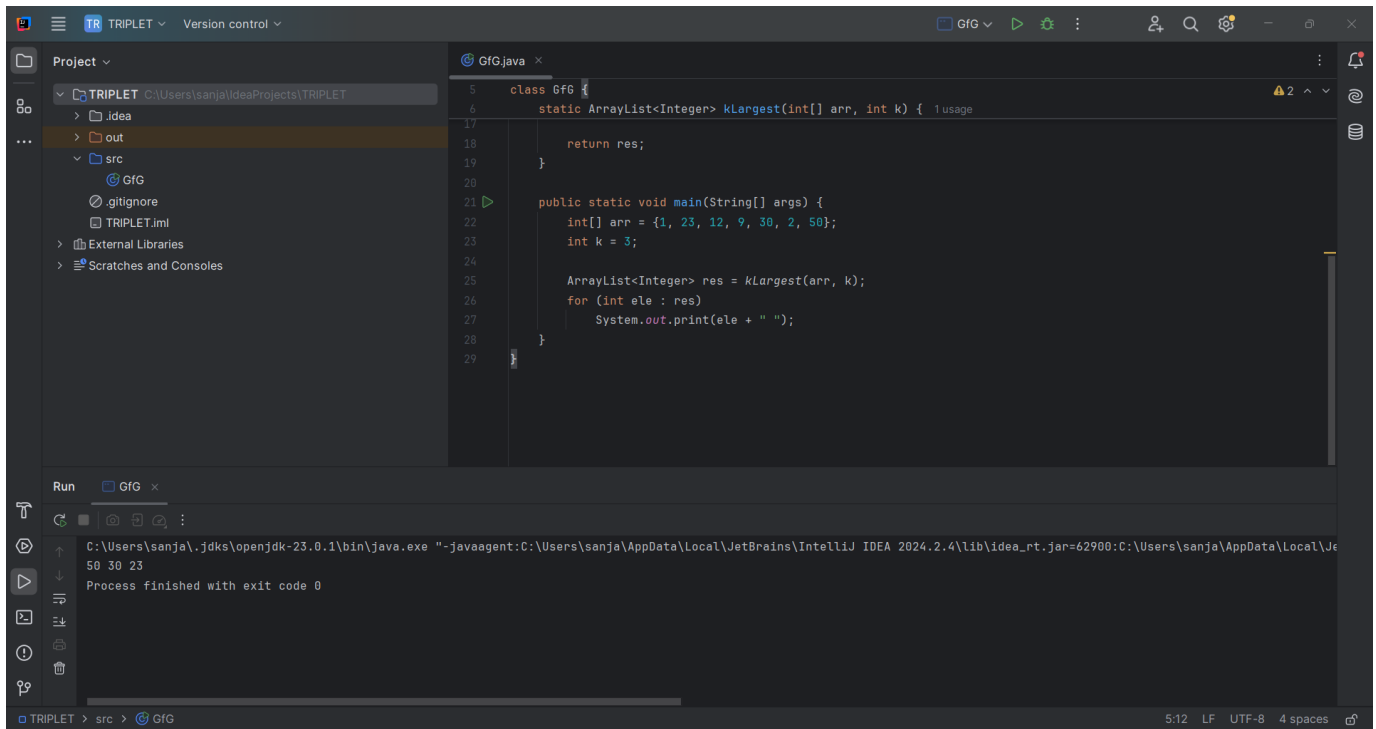
        ArrayList<Integer> res = new ArrayList<>();
        for (int i = 0; i < k; i++)
            res.add(arrInteger[i]);

        return res;
    }

    public static void main(String[] args) {
        int[] arr = {1, 23, 12, 9, 30, 2, 50};
        int k = 3;

        ArrayList<Integer> res = kLargest(arr, k);
        for (int ele : res)
            System.out.print(ele + " ");
    }
}
```


Output:



```
5 class GfG {
6     static ArrayList<Integer> kLargest(int[] arr, int k) {
17
18         return res;
19     }
20
21     public static void main(String[] args) {
22         int[] arr = {1, 23, 12, 9, 30, 2, 50};
23         int k = 3;
24
25         ArrayList<Integer> res = kLargest(arr, k);
26         for (int ele : res)
27             System.out.print(ele + " ");
28     }
29 }
```

Run GfG

```
C:\Users\sanja\.jdk\openjdk-23.0.1\bin\java.exe "-javaagent:C:\Users\sanja\AppData\Local\JetBrains\IntelliJ IDEA 2024.2.4\lib\idea_rt.jar=62900:C:\Users\sanja\AppData\Local\Jc
50 30 23
Process finished with exit code 0
```

6. Form The Largest Number

Time Complexity : $O(n \cdot \log(n))$

Solution:

```
import java.util.*;
```

```
class GfG {
    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) {  
    int[] arr = { 3, 30, 34, 5, 9 };  
    System.out.println(findLargest(arr));  
}
```

Output:

