VIJAY M

22IT125 DAY-7

CODING PRACTICES AND PROBLEM

1.3Sum closest

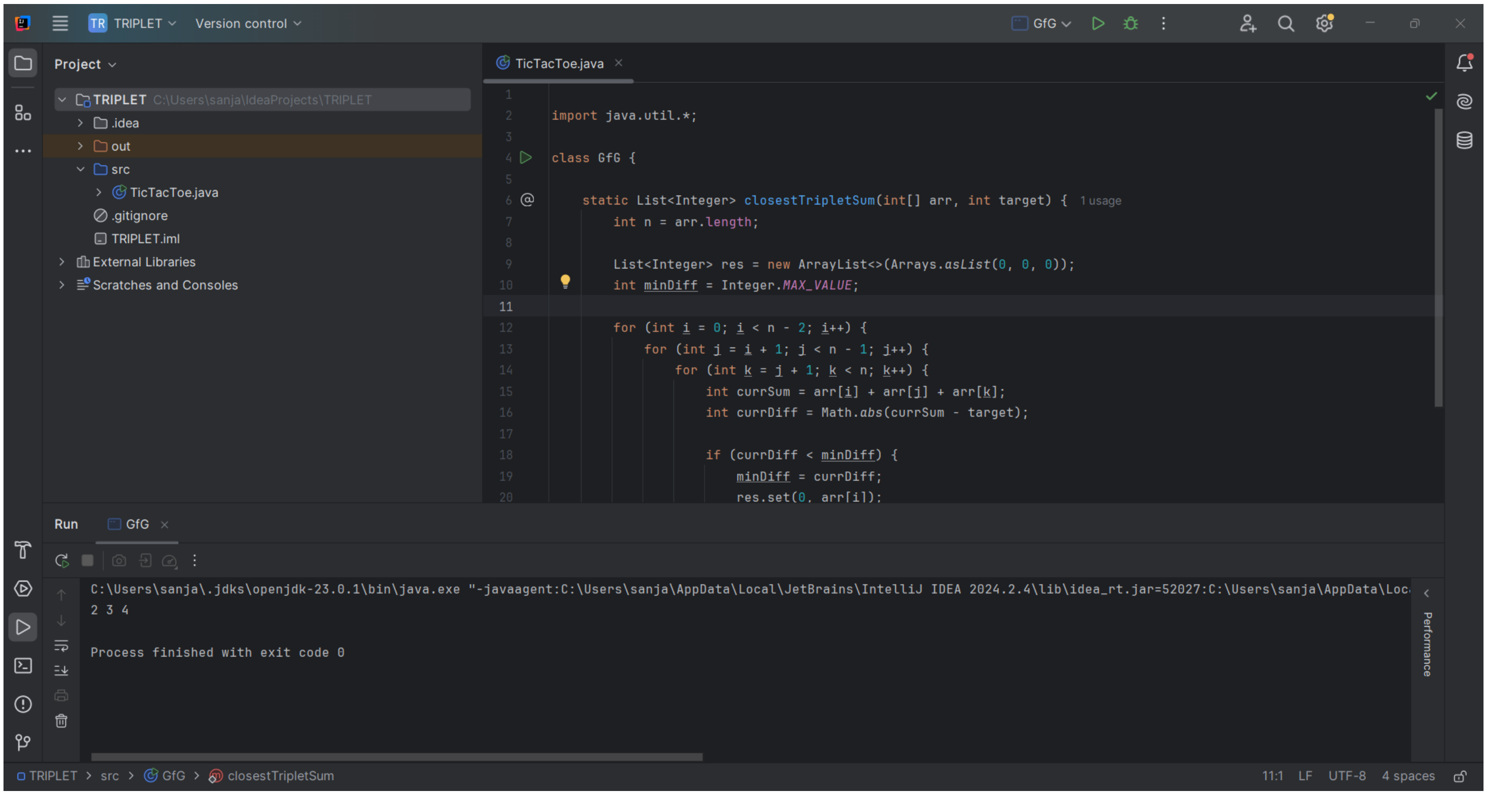
Time complexity : O(n^2)

Solution:

```
import java.util.*;

class GfG {

   static List<Integer> closestTripletSum(int[] arr, int target) {
      int n = arr.length;

      List<Integer> res = new ArrayList<>(Arrays.asList(0, 0, 0));
      int minDiff = Integer.MAX_VALUE;

      for (int i = 0; i < n - 2; i++) {
         for (int j = i + 1; j < n - 1; j++) {
            for (int k = j + 1; k < n; k++) {
               int currSum = arr[i] + arr[j] + arr[k];
               int currDiff = Math.abs(currSum - target);

               if (currDiff < minDiff) {
                  minDiff = currDiff;
                  res.set(0, arr[i]);
                  res.set(1, arr[j]);
                  res.set(2, arr[k]);
               }
            }
         }
      }
```

```
        return res;
    }

    public static void main(String[] args) {
        int[] arr = {1, 2, 3, 4, -5};
        int target = 10;

        List<Integer> res = closestTripletSum(arr, target);
        System.out.println(res.get(0) + " " + res.get(1) + " " + res.get(2));
    }
}
```

Output:



```
import java.util.*;

class GfG {

    static List<Integer> closestTripletSum(int[] arr, int target) {
        int n = arr.length;

        List<Integer> res = new ArrayList<>(Arrays.asList(0, 0, 0));
        int minDiff = Integer.MAX_VALUE;

        for (int i = 0; i < n - 2; i++) {
            for (int j = i + 1; j < n - 1; j++) {
                for (int k = j + 1; k < n; k++) {
                    int currSum = arr[i] + arr[j] + arr[k];
                    int currDiff = Math.abs(currSum - target);

                    if (currDiff < minDiff) {
                        minDiff = currDiff;
                        res.set(0, arr[i]);
```

```
C:\Users\sanja\.jdks\openjdk-23.0.1\bin\java.exe "-javaagent:C:\Users\sanja\AppData\Local\JetBrains\IntelliJ IDEA 2024.2.4\lib\idea_rt.jar=52027:C:\Users\sanja\AppData\Loc
2 3 4

Process finished with exit code 0
```

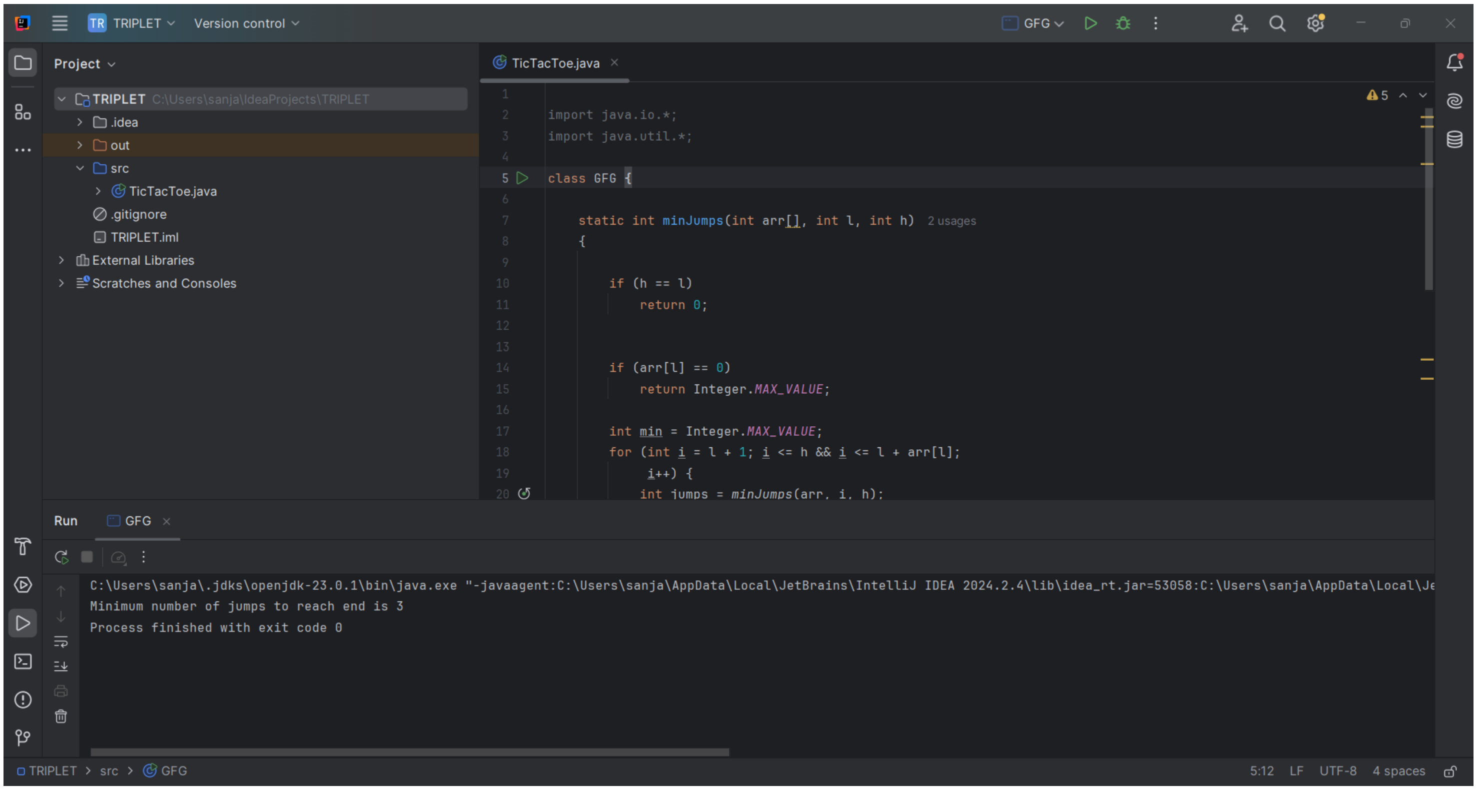
2.Jump Game II

Time Complexity : O(n^n)

Solution:

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

class GFG {

    static int minJumps(int arr[], int l, int h)
    {

        if (h == l)
            return 0;


        if (arr[l] == 0)
            return Integer.MAX_VALUE;

        int min = Integer.MAX_VALUE;
        for (int i = l + 1; i <= h && i <= l + arr[l];
             i++) {
            int jumps = minJumps(arr, i, h);
            if (jumps != Integer.MAX_VALUE
                    && jumps + 1 < min)
                min = jumps + 1;
        }
        return min;
    }

    public static void main(String args[])
    {
        int arr[] = { 1, 3, 5, 8, 9, 2, 6, 7, 6, 8, 9 };
        int n = arr.length;
        System.out.print(
                "Minimum number of jumps to reach end is "
                        + minJumps(arr, 0, n - 1));
    }
}
```

Output:



```

import java.io.*;
import java.util.*;

class GFG {

    static int minJumps(int arr[], int l, int h)  2 usages
    {

        if (h == l)
            return 0;


        if (arr[l] == 0)
            return Integer.MAX_VALUE;

        int min = Integer.MAX_VALUE;
        for (int i = l + 1; i <= h && i <= l + arr[l];
             i++) {
            int jumps = minJumps(arr, i, h);
```

```
C:\Users\sanja\.jdks\openjdk-23.0.1\bin\java.exe "-javaagent:C:\Users\sanja\AppData\Local\JetBrains\IntelliJ IDEA 2024.2.4\lib\idea_rt.jar=53058:C:\Users\sanja\AppData\Local\Je
Minimum number of jumps to reach end is 3
Process finished with exit code 0
```

3.Group Anagrams

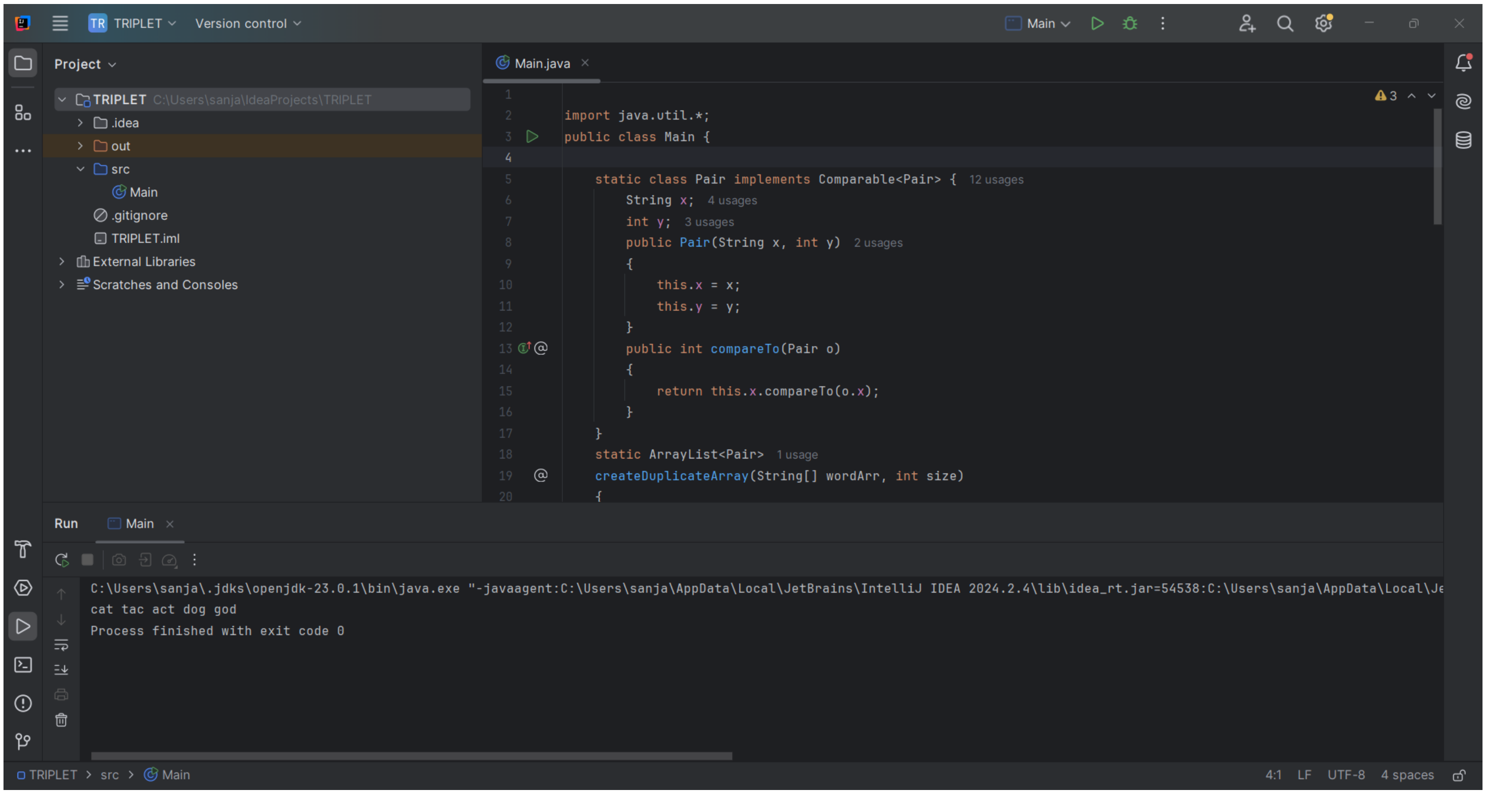
Time Complexity : O(NMLogM + MNLogN)

Solution:

```
import java.util.*;
public class Main {

    static class Pair implements Comparable<Pair> {
        String x;
        int y;
        public Pair(String x, int y)
        {
            this.x = x;
            this.y = y;
        }
        public int compareTo(Pair o)
        {
            return this.x.compareTo(o.x);
        }
    }
    static ArrayList<Pair>
    createDuplicateArray(String[] wordArr, int size)
    {
        ArrayList<Pair> dupArray = new ArrayList<Pair>();
        for (int i = 0; i < size; i++) {
            Pair p = new Pair(wordArr[i], i);
            dupArray.add(p);

        }
        return dupArray;
    }
    static void printAnagramsTogether(String[] wordArr,
                                      int size)
    {

        ArrayList<Pair> dupArray = new ArrayList<Pair>();

        dupArray = createDuplicateArray(
                wordArr, size);

        for (int i = 0; i < size; ++i) {
            Pair e = dupArray.get(i);
            char[] arr = e.x.toCharArray();
            Arrays.sort(arr);
            String x = String.valueOf(arr);
            Pair p = new Pair(x, e.y);
```

```
            dupArray.set(i, p);
        }

        Collections.sort(dupArray);

        for (int i = 0; i < size; ++i)
            System.out.print(wordArr[dupArray.get(i).y]
                    + " ");
    }

    public static void main(String[] args)
    {
        String[] wordArr
                = { "cat", "dog", "tac", "god", "act" };
        printAnagramsTogether(wordArr, wordArr.length);
    }
}
```

Output:



```

import java.util.*;
public class Main {

    static class Pair implements Comparable<Pair> {  12 usages
        String x;  4 usages
        int y;  3 usages
        public Pair(String x, int y)  2 usages
        {
            this.x = x;
            this.y = y;
        }
        public int compareTo(Pair o)
        {
            return this.x.compareTo(o.x);
        }
    }
    static ArrayList<Pair>  1 usage
    createDuplicateArray(String[] wordArr, int size)
    {
```

```
C:\Users\sanja\.jdks\openjdk-23.0.1\bin\java.exe "-javaagent:C:\Users\sanja\AppData\Local\JetBrains\IntelliJ IDEA 2024.2.4\lib\idea_rt.jar=54538:C:\Users\sanja\AppData\Local\Je
cat tac act dog god
Process finished with exit code 0
```

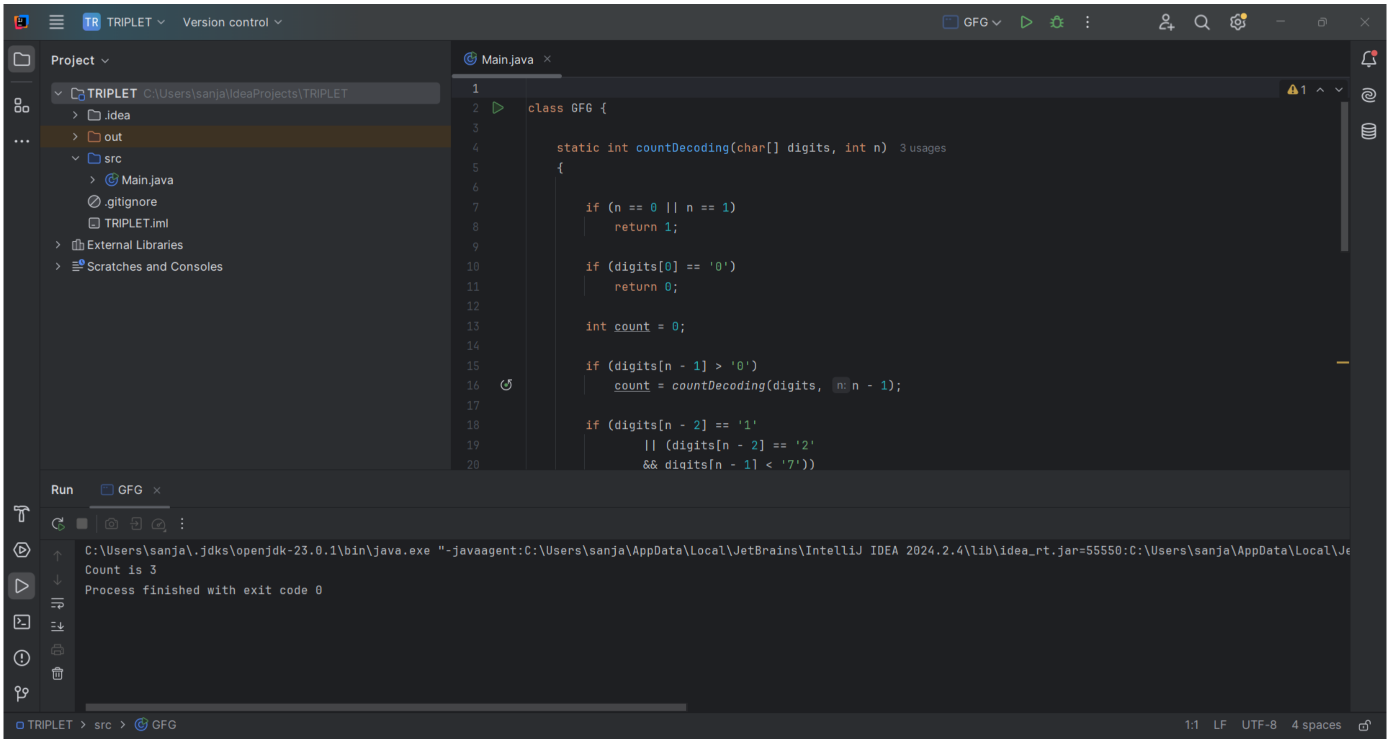
4.Decoding Ways

Time Complexity : O(n)

Solution:

```
class GFG {

  static int countDecoding(char[] digits, int n)
  {

    if (n == 0 || n == 1)
      return 1;

    if (digits[0] == '0')
      return 0;

    int count = 0;

    if (digits[n - 1] > '0')
      count = countDecoding(digits, n - 1);

    if (digits[n - 2] == '1'
        || (digits[n - 2] == '2'
        && digits[n - 1] < '7'))
      count += countDecoding(digits, n - 2);

    return count;
  }

  static int countWays(char[] digits, int n)
  {
    if (n == 0 || (n == 1 && digits[0] == '0'))
      return 0;
    return countDecoding(digits, n);
  }
  public static void main(String[] args)
  {
      char digits[] = { '1', '2', '3', '4' };
      int n = digits.length;
      System.out.printf("Count is %d",
              countWays(digits, n));
  }
}
```

Output:



```

class GFG {

    static int countDecoding(char[] digits, int n)  3 usages
    {

        if (n == 0 || n == 1)
            return 1;

        if (digits[0] == '0')
            return 0;

        int count = 0;

        if (digits[n - 1] > '0')
            count = countDecoding(digits, n: n - 1);

        if (digits[n - 2] == '1'
                || (digits[n - 2] == '2'
                && digits[n - 1] < '7'))
```

```
C:\Users\sanja\.jdks\openjdk-23.0.1\bin\java.exe "-javaagent:C:\Users\sanja\AppData\Local\JetBrains\IntelliJ IDEA 2024.2.4\lib\idea_rt.jar=55550:C:\Users\sanja\AppData\Local\Je
Count is 3
Process finished with exit code 0
```

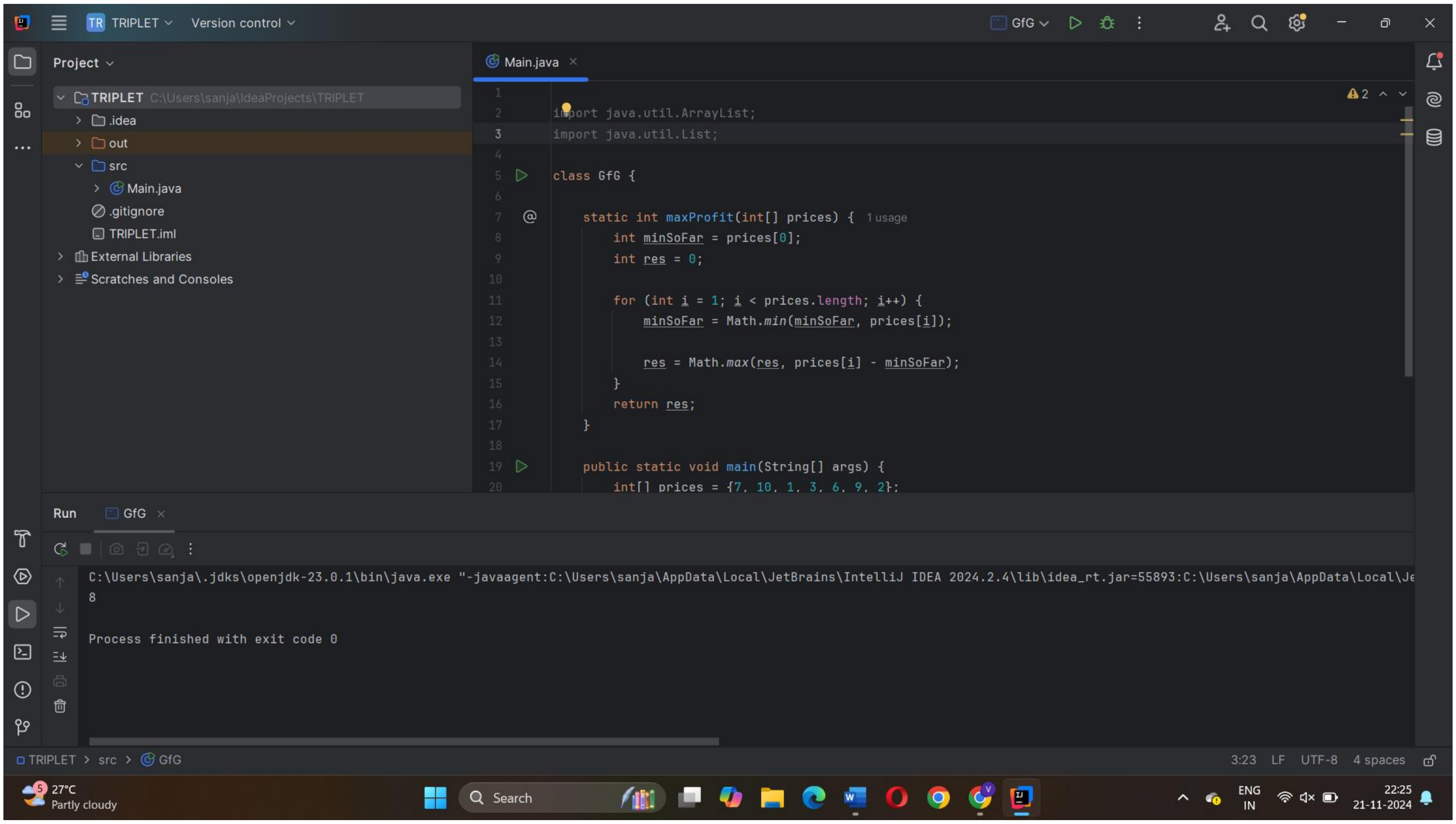
5.Best Time to Buy and Sell

Time Complexity : O(n)

Solution:

```
import java.util.ArrayList;
import java.util.List;
class GfG {
    static int maxProfit(int[] prices) {
        int minSoFar = prices[0];
        int res = 0;

        for (int i = 1; i < prices.length; i++) {
            minSoFar = Math.min(minSoFar, prices[i]);

            res = Math.max(res, prices[i] - minSoFar);
        }
        return res;
    }

    public static void main(String[] args) {
        int[] prices = {7, 10, 1, 3, 6, 9, 2};
        System.out.println(maxProfit(prices));
    }
}
```

Output:



```
import java.util.ArrayList;
import java.util.List;

class GfG {

    static int maxProfit(int[] prices) {
        int minSoFar = prices[0];
        int res = 0;

        for (int i = 1; i < prices.length; i++) {
            minSoFar = Math.min(minSoFar, prices[i]);

            res = Math.max(res, prices[i] - minSoFar);
        }
        return res;
    }

    public static void main(String[] args) {
        int[] prices = {7, 10, 1, 3, 6, 9, 2};
```

```
C:\Users\sanja\.jdks\openjdk-23.0.1\bin\java.exe "-javaagent:C:\Users\sanja\AppData\Local\JetBrains\IntelliJ IDEA 2024.2.4\lib\idea_rt.jar=55893:C:\Users\sanja\AppData\Local\Je
8

Process finished with exit code 0
```

6.Number of Islands

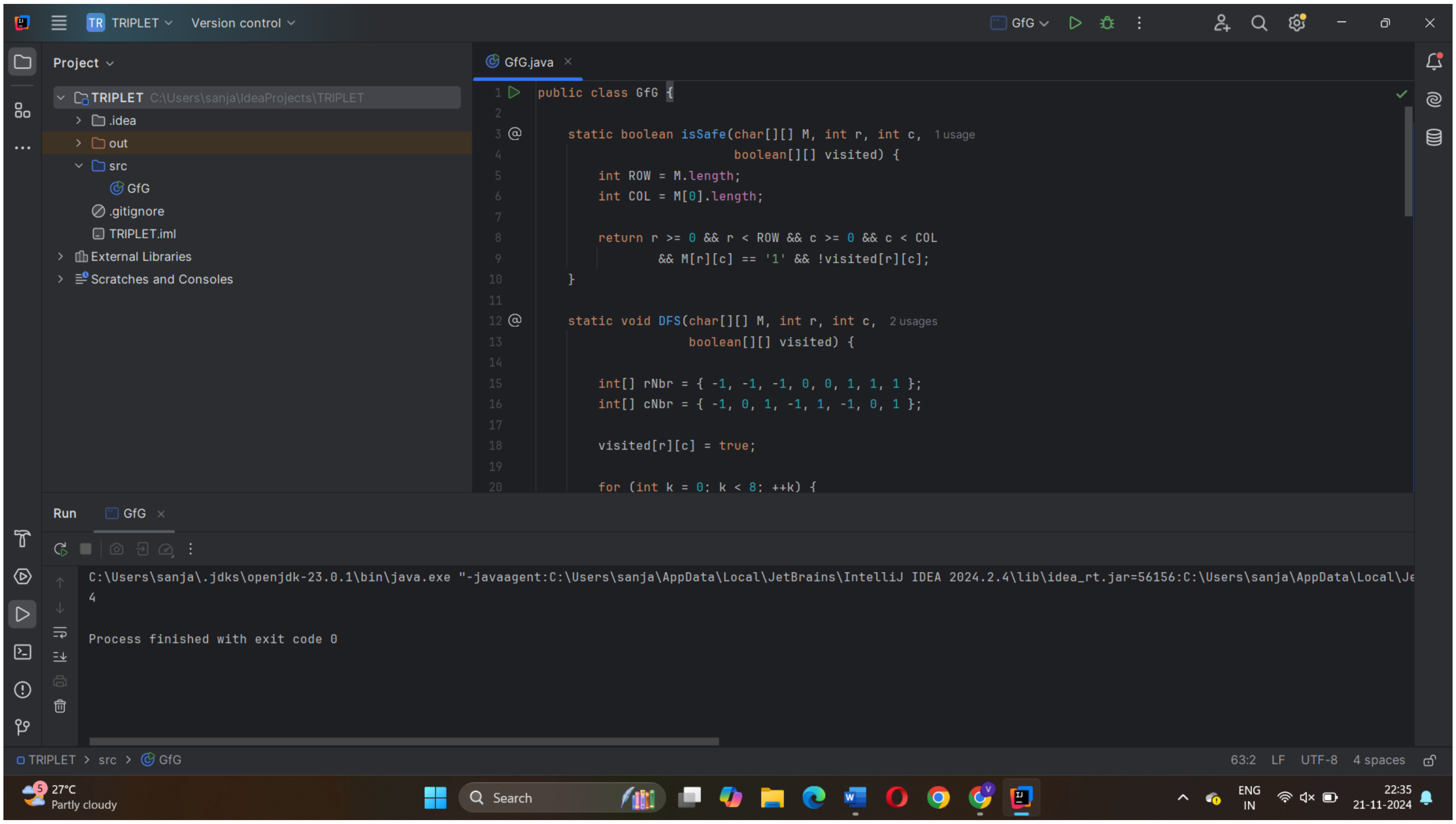
Time Complexity : O(mn)

Solution:

```
public class GfG {

  static boolean isSafe(char[][] M, int r, int c,
                        boolean[][] visited) {
    int ROW = M.length;
    int COL = M[0].length;

    return r >= 0 && r < ROW && c >= 0 && c < COL
        && M[r][c] == '1' && !visited[r][c];
  }

  static void DFS(char[][] M, int r, int c,
                  boolean[][] visited) {

    int[] rNbr = { -1, -1, -1, 0, 0, 1, 1, 1 };
    int[] cNbr = { -1, 0, 1, -1, 1, -1, 0, 1 };

    visited[r][c] = true;

    for (int k = 0; k < 8; ++k) {
      int newR = r + rNbr[k];
      int newC = c + cNbr[k];
      if (isSafe(M, newR, newC, visited)) {
        DFS(M, newR, newC, visited);
      }
    }
  }

  static int countIslands(char[][] M) {
    int ROW = M.length;
```

```
        int COL = M[0].length;

        boolean[][] visited = new boolean[ROW][COL];

        int count = 0;
        for (int r = 0; r < ROW; ++r) {
            for (int c = 0; c < COL; ++c) {

                if (M[r][c] == '1' && !visited[r][c]) {


                    DFS(M, r, c, visited);

                    ++count;
                }
            }
        }
        return count;
    }

    public static void main(String[] args) {
        char[][] M = {
                { '1', '1', '0', '0', '0' },
                { '0', '1', '0', '0', '1' },
                { '1', '0', '0', '1', '1' },
                { '0', '0', '0', '0', '0' },
                { '1', '0', '1', '1', '0' }
        };

        System.out.println(countIslands(M));
    }
}
```

Output:



```
public class GfG {

    static boolean isSafe(char[][] M, int r, int c,
                          boolean[][] visited) {
        int ROW = M.length;
        int COL = M[0].length;

        return r >= 0 && r < ROW && c >= 0 && c < COL
                && M[r][c] == '1' && !visited[r][c];
    }

    static void DFS(char[][] M, int r, int c,
                    boolean[][] visited) {

        int[] rNbr = { -1, -1, -1, 0, 0, 1, 1, 1 };
        int[] cNbr = { -1, 0, 1, -1, 1, -1, 0, 1 };

        visited[r][c] = true;

        for (int k = 0; k < 8; ++k) {
```

```
C:\Users\sanja\.jdks\openjdk-23.0.1\bin\java.exe "-javaagent:C:\Users\sanja\AppData\Local\JetBrains\IntelliJ IDEA 2024.2.4\lib\idea_rt.jar=56156:C:\Users\sanja\AppData\Local\Je
4

Process finished with exit code 0
```

7.Quick Sort

Time cpmplexity:

Best Case : (Ω(n log n))

Worst Case : (θ(n log n)

Average Case : (O(n^2))

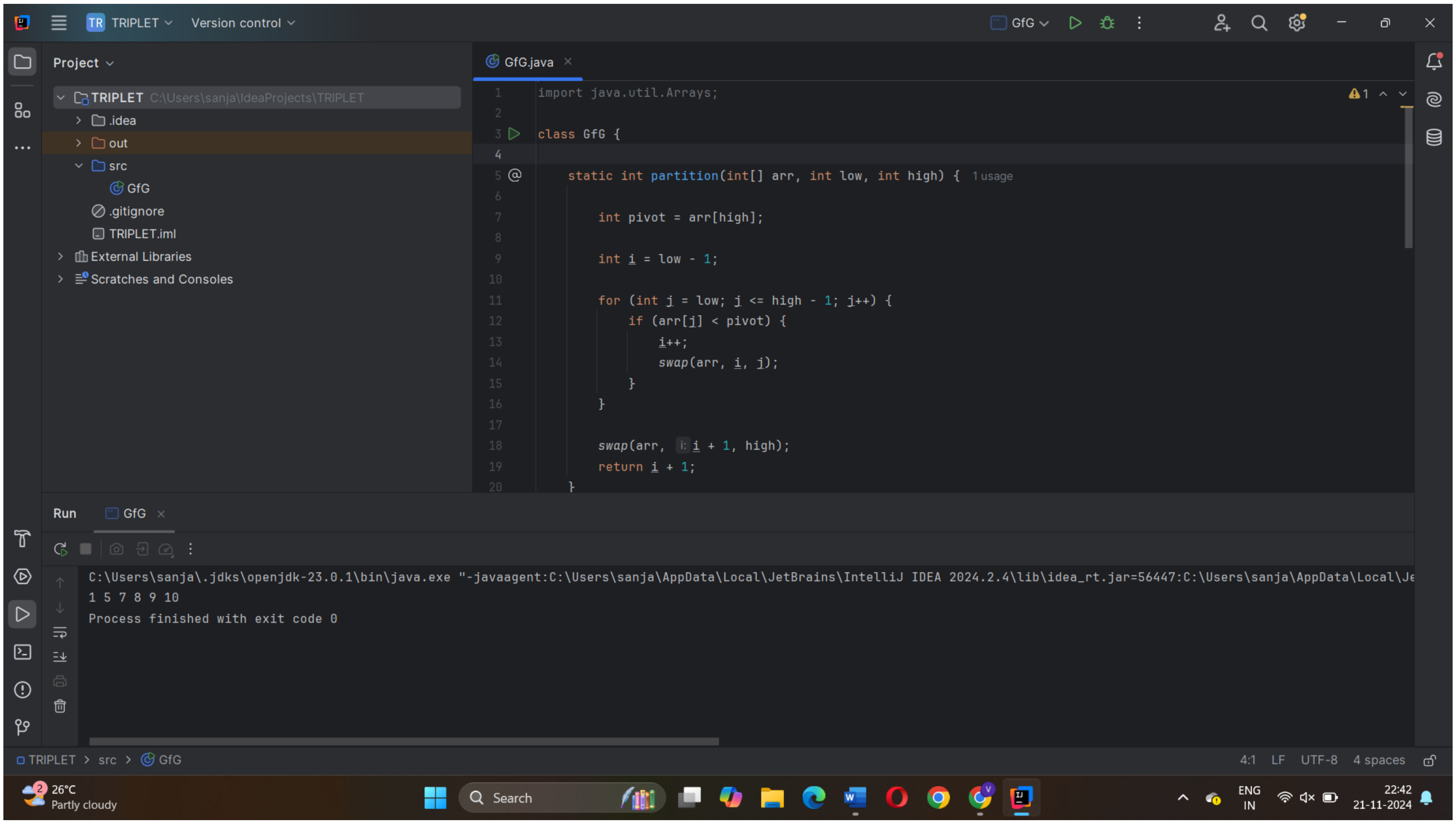
Solution:

```
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:



```
import java.util.Arrays;

class GfG {

    static int partition(int[] arr, int low, int high) {  1 usage

        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;
    }
```

```
C:\Users\sanja\.jdks\openjdk-23.0.1\bin\java.exe "-javaagent:C:\Users\sanja\AppData\Local\JetBrains\IntelliJ IDEA 2024.2.4\lib\idea_rt.jar=56447:C:\Users\sanja\AppData\Local\Je
1 5 7 8 9 10
Process finished with exit code 0
```

8.Merge Sort

Time Complexity:

Best Case : O(n log n)

Worst Case : O(n log n)

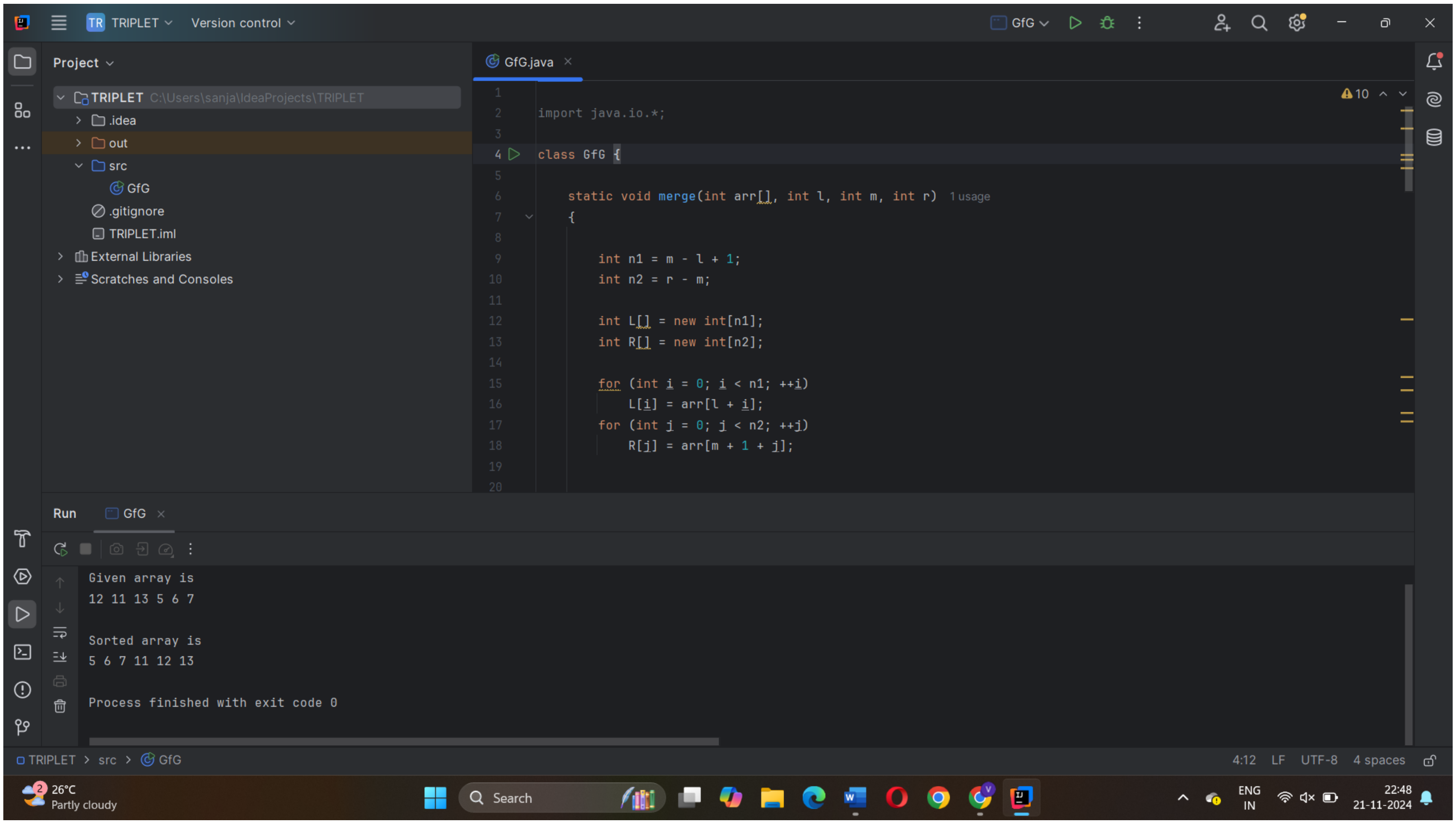
Average Case : O(n log n)

Solution:

```
import java.io.*;

class GfG {

    static void merge(int arr[], int l, int m, int r)
    {

        int n1 = m - l + 1;
        int n2 = r - m;

        int L[] = new int[n1];
        int R[] = new int[n2];

        for (int i = 0; i < n1; ++i)
            L[i] = arr[l + i];
        for (int j = 0; j < n2; ++j)
            R[j] = arr[m + 1 + j];


        int i = 0, j = 0;

        int k = l;
        while (i < n1 && j < n2) {
            if (L[i] <= R[j]) {
                arr[k] = L[i];
                i++;
            }
            else {
                arr[k] = R[j];
                j++;
            }
            k++;
        }
    while (i < n1) {
        arr[k] = L[i];
        i++;
        k++;
    }

    while (j < n2) {
        arr[k] = R[j];
        j++;
        k++;
    }
}

static void sort(int arr[], int l, int r)
{
    if (l < r) {

        int m = l + (r - l) / 2;

        sort(arr, l, m);
        sort(arr, m + 1, r);

        merge(arr, l, m, r);
    }
}

static void printArray(int arr[])
{
    int n = arr.length;
    for (int i = 0; i < n; ++i)
        System.out.print(arr[i] + " ");
    System.out.println();
}

public static void main(String args[])
```

```
{
    int arr[] = { 12, 11, 13, 5, 6, 7 };

    System.out.println("Given array is");
    printArray(arr);

    sort(arr, 0, arr.length - 1);

    System.out.println("\nSorted array is");
    printArray(arr);
  }
}
```

Output:



```
import java.io.*;

class GfG {

    static void merge(int arr[], int l, int m, int r)
    {

        int n1 = m - l + 1;
        int n2 = r - m;

        int L[] = new int[n1];
        int R[] = new int[n2];

        for (int i = 0; i < n1; ++i)
            L[i] = arr[l + i];
        for (int j = 0; j < n2; ++j)
            R[j] = arr[m + 1 + j];
```

```
Given array is
12 11 13 5 6 7

Sorted array is
5 6 7 11 12 13

Process finished with exit code 0
```

9.Ternary Search

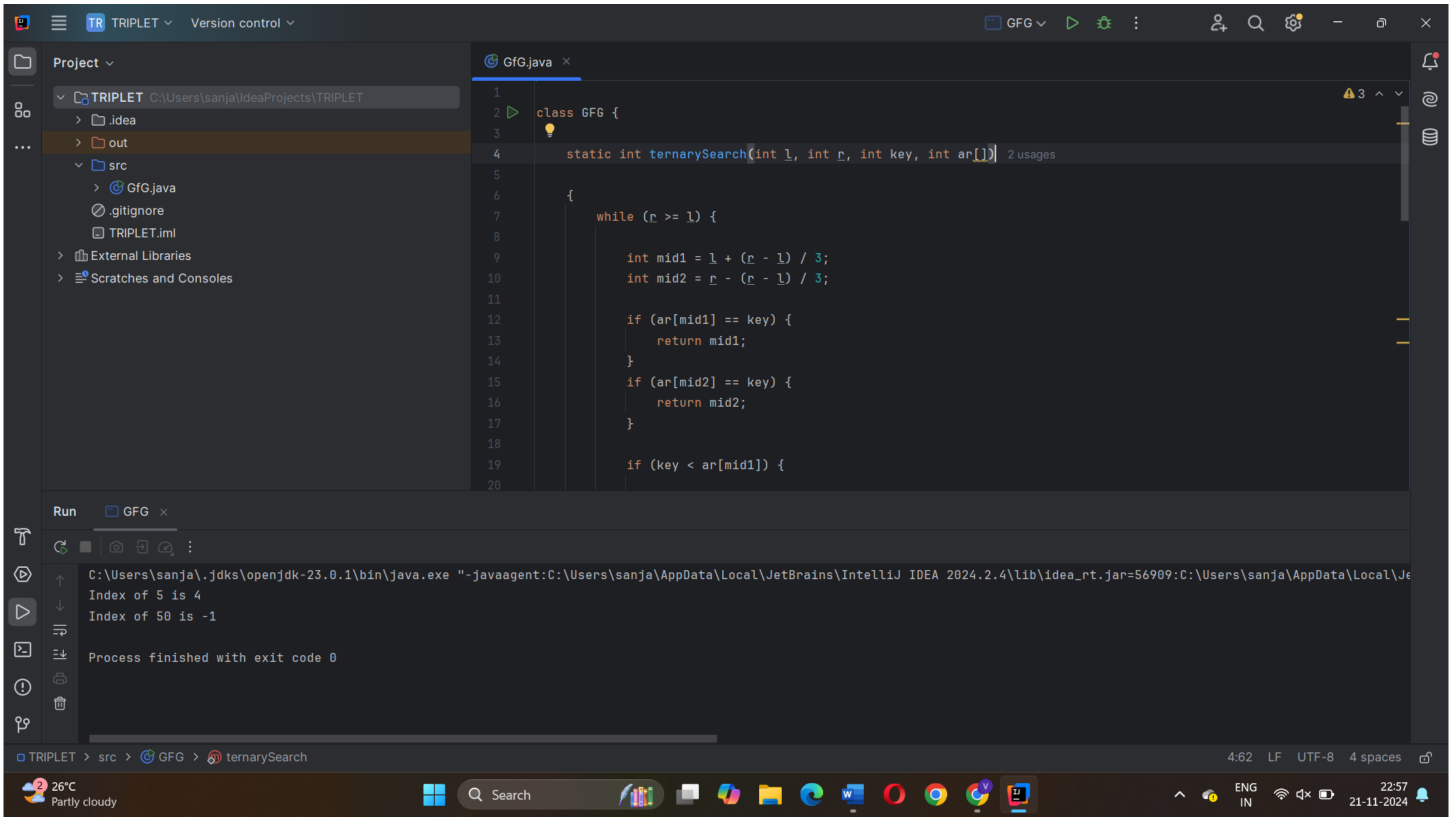
Time Complexity : O(2 * log3n)

Solution:

```
class GFG {
    static int ternarySearch(int l, int r, int key, int ar[])
    {
        while (r >= l) {
            int mid1 = l + (r - l) / 3;
            int mid2 = r - (r - l) / 3;

            if (ar[mid1] == key) {
                return mid1;
            }
            if (ar[mid2] == key) {
                return mid2;
            }

            if (key < ar[mid1]) {

                r = mid1 - 1;
            }
            else if (key > ar[mid2]) {

                l = mid2 + 1;
            }
            else {

                l = mid1 + 1;
                r = mid2 - 1;
            }
        }

        return -1;
```

```
  }
  public static void main(String args[])
  {
      int l, r, p, key;
      int ar[] = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 };
      l = 0;
      r = 9;
      key = 5;
      p = ternarySearch(l, r, key, ar);
      System.out.println("Index of " + key + " is " + p);
      key = 50;
      p = ternarySearch(l, r, key, ar);
      System.out.println("Index of " + key + " is " + p);
  }
}
```

Output:



```

class GFG {

    static int ternarySearch(int l, int r, int key, int ar[])

    {
        while (r >= l) {

            int mid1 = l + (r - l) / 3;
            int mid2 = r - (r - l) / 3;

            if (ar[mid1] == key) {
                return mid1;
            }
            if (ar[mid2] == key) {
                return mid2;
            }

            if (key < ar[mid1]) {
```

```
C:\Users\sanja\.jdks\openjdk-23.0.1\bin\java.exe "-javaagent:C:\Users\sanja\AppData\Local\JetBrains\IntelliJ IDEA 2024.2.4\lib\idea_rt.jar=56909:C:\Users\sanja\AppData\Local\Je
Index of 5 is 4
Index of 50 is -1

Process finished with exit code 0
```

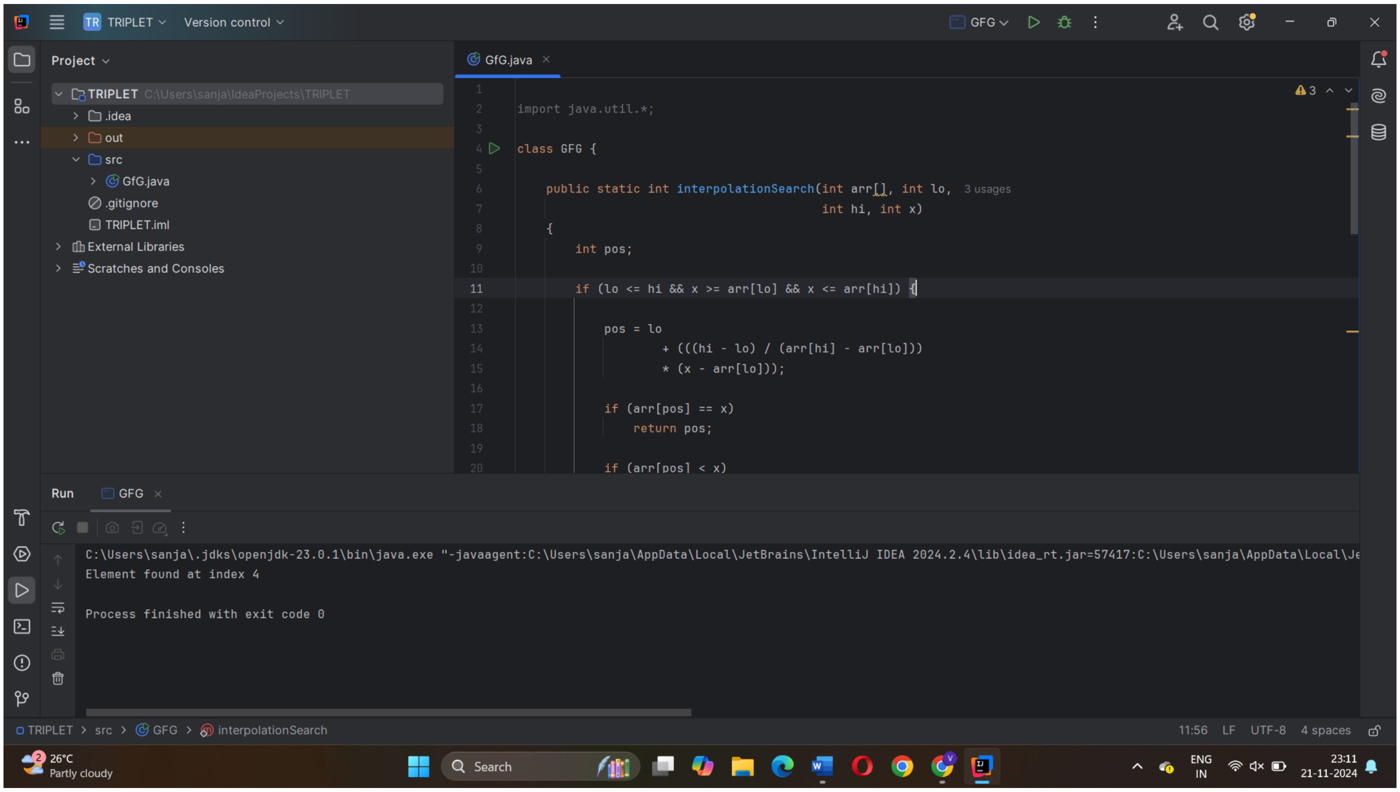
10.Interpolation Search

Time Complexity : $O(\log_2(\log_2 n))$

Solution:

```
import java.util.*;
class GFG {
   public static int interpolationSearch(int arr[], int lo,
                                          int hi, int x)
   {
      int pos;

      if (lo <= hi && x >= arr[lo] && x <= arr[hi]) {

         pos = lo
                + (((hi - lo) / (arr[hi] - arr[lo]))
                * (x - arr[lo]));

         if (arr[pos] == x)
            return pos;

         if (arr[pos] < x)
            return interpolationSearch(arr, pos + 1, hi,
                  x);

         if (arr[pos] > x)
            return interpolationSearch(arr, lo, pos - 1,
                  x);
      }
      return -1;
   }

   public static void main(String[] args)
   {
    int arr[] = { 10, 12, 13, 16, 18, 19, 20, 21,
            22, 23, 24, 33, 35, 42, 47 };

    int n = arr.length;

    int x = 18;
    int index = interpolationSearch(arr, 0, n - 1, x);

    if (index != -1)
        System.out.println("Element found at index "
                + index);
    else
        System.out.println("Element not found.");
  }
}
```

Output:



```
import java.util.*;

class GFG {

    public static int interpolationSearch(int arr[], int lo,
                                          int hi, int x)
    {
        int pos;

        if (lo <= hi && x >= arr[lo] && x <= arr[hi]) {

            pos = lo
                    + (((hi - lo) / (arr[hi] - arr[lo]))
                    * (x - arr[lo]));

            if (arr[pos] == x)
                return pos;

            if (arr[pos] < x)
```

```
C:\Users\sanja\.jdks\openjdk-23.0.1\bin\java.exe "-javaagent:C:\Users\sanja\AppData\Local\JetBrains\IntelliJ IDEA 2024.2.4\lib\idea_rt.jar=57417:C:\Users\sanja\AppData\Local\Je
Element found at index 4

Process finished with exit code 0
```