VIJAY M

22IT125 DAY-4

CODING PRACTICES AND PROBLEM

1.Stock Buy and Sell

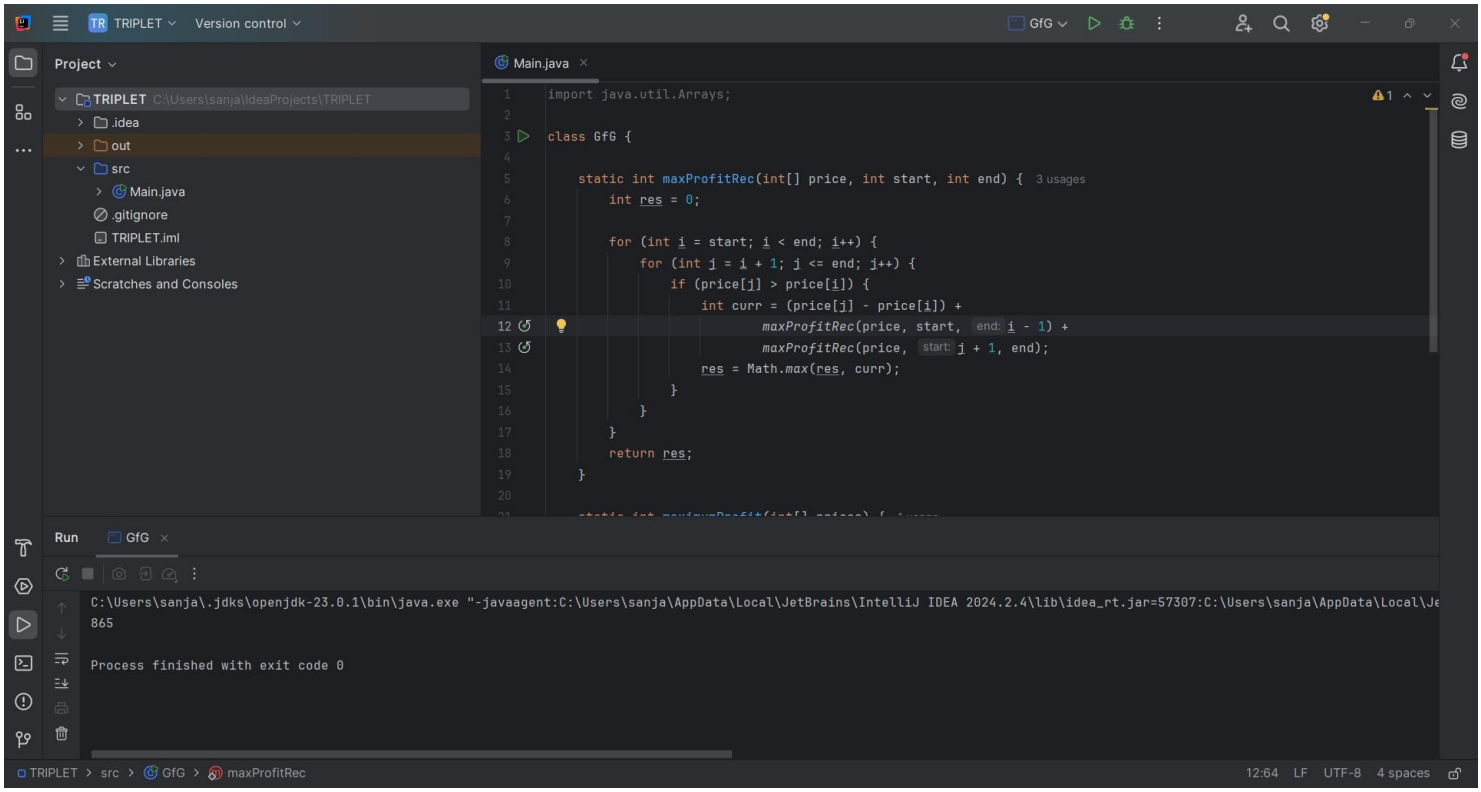
Time Complexity : O(n)

Solution:

```
import java.util.Arrays;

class GfG {

    static int maxProfitRec(int[] price, int start, int end) {
        int res = 0;

        for (int i = start; i < end; i++) {
            for (int j = i + 1; j <= end; j++) {
                if (price[j] > price[i]) {
                    int curr = (price[j] - price[i]) +
                            maxProfitRec(price, start, i - 1) +
                            maxProfitRec(price, j + 1, end);
                    res = Math.max(res, curr);
                }
            }
        }
        return res;
    }

    static int maximumProfit(int[] prices) {
        return maxProfitRec(prices, 0, prices.length - 1);
    }
```

```
    public static void main(String[] args) {
        int[] prices = {100, 180, 260, 310, 40, 535, 695};
        System.out.println(maximumProfit(prices));
    }
}
```

Output:



2.Coin Change

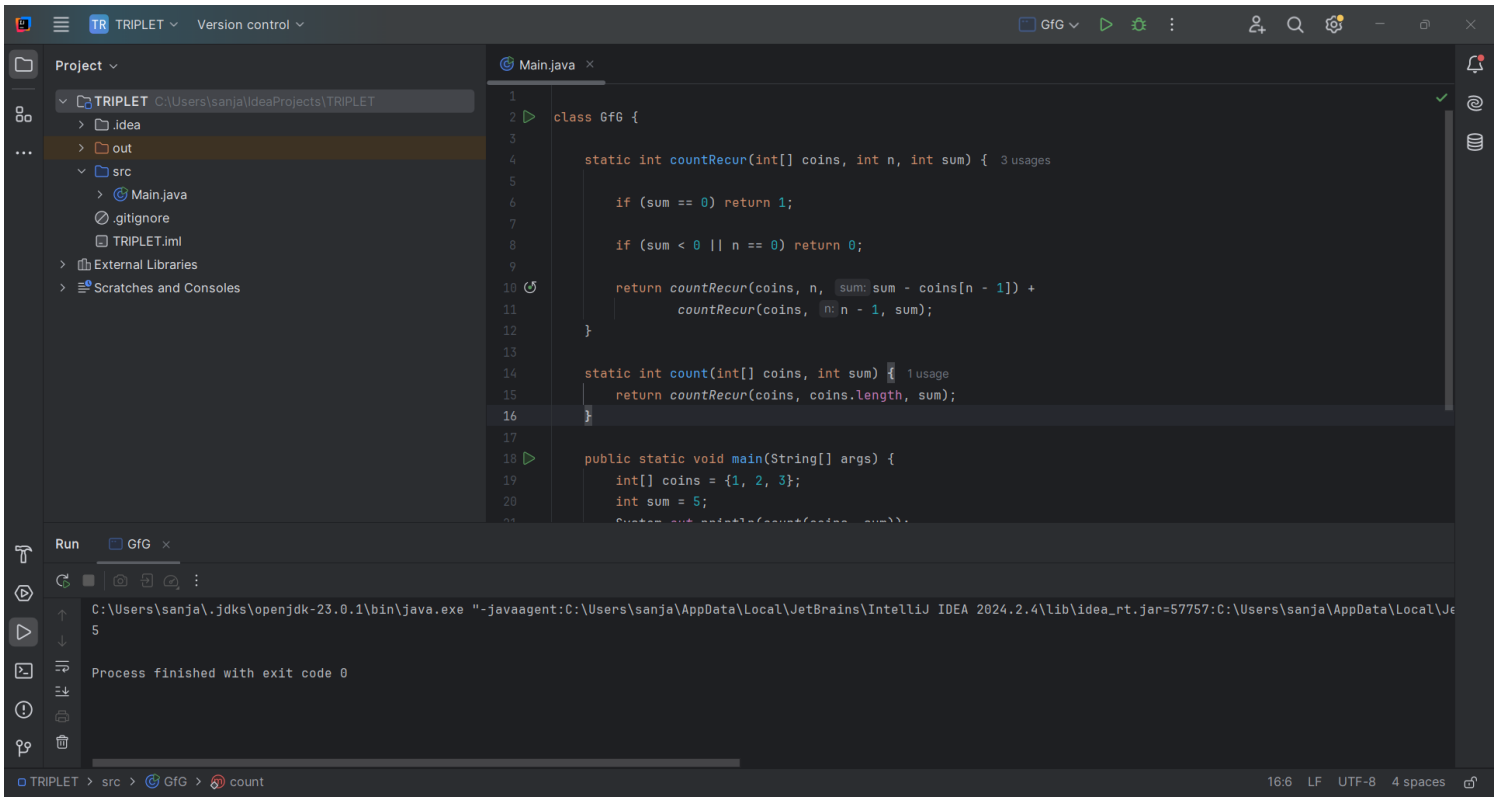
Time Complexity : O(sum*n)

Solution:

```
class GfG {

    static int countRecur(int[] coins, int n, int sum) {

        if (sum == 0) return 1;
```

```
    if (sum < 0 || n == 0) return 0;

    return countRecur(coins, n, sum - coins[n - 1]) +
            countRecur(coins, n - 1, sum);
  }

  static int count(int[] coins, int sum) {
    return countRecur(coins, coins.length, sum);
  }

  public static void main(String[] args) {
    int[] coins = {1, 2, 3};
    int sum = 5;
    System.out.println(count(coins, sum));
  }
}
```

Output:



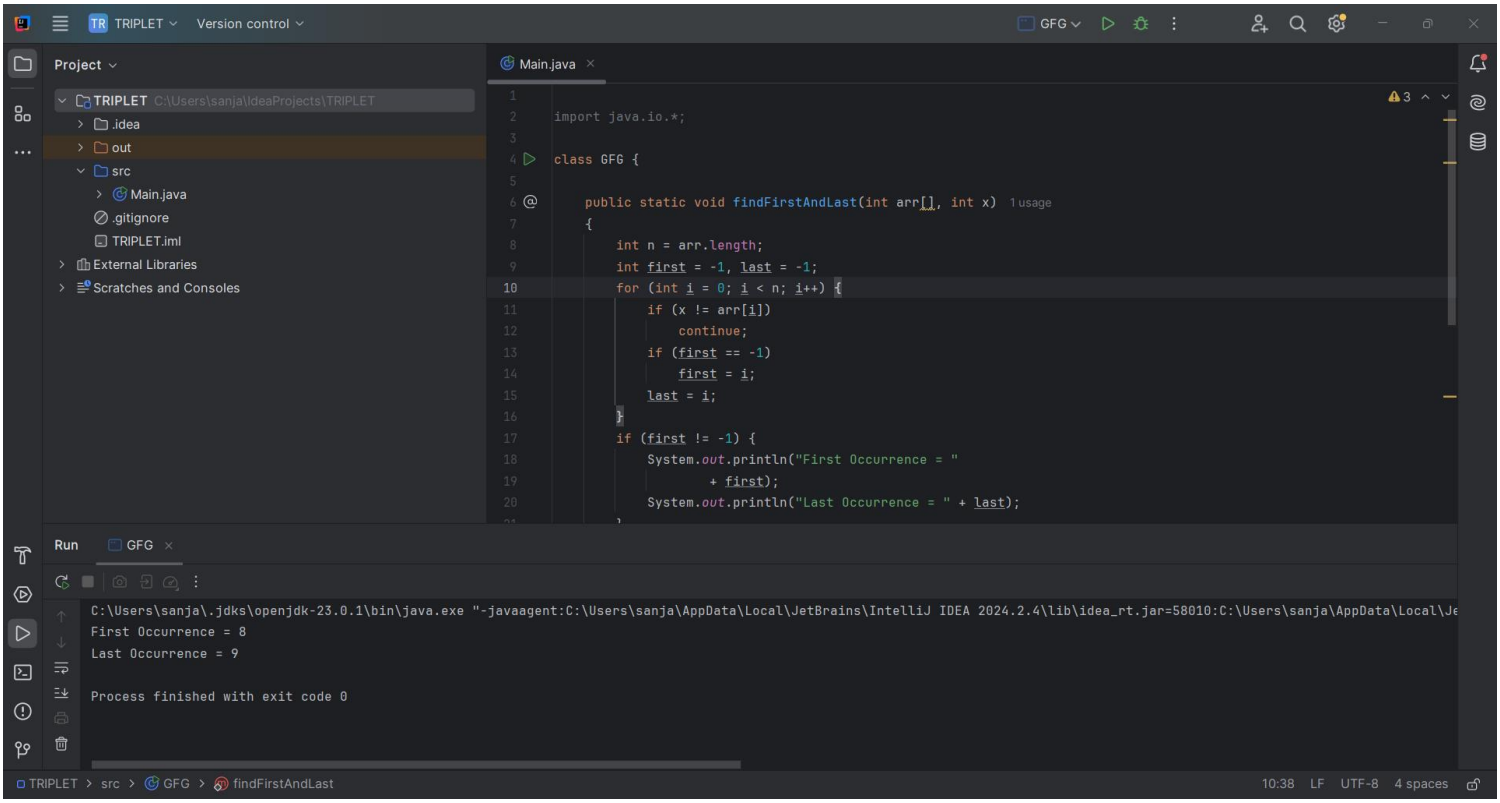
3.First and Last Occurance

Time Complexity : O(n)

Solution:

```
import java.io.*;
class GFG {
    public static void findFirstAndLast(int arr[], int x)
    {
        int n = arr.length;
        int first = -1, last = -1;
        for (int i = 0; i < n; i++) {
            if (x != arr[i])
                continue;
            if (first == -1)
                first = i;
            last = i;
        }
        if (first != -1) {
            System.out.println("First Occurrence = "
                    + first);
            System.out.println("Last Occurrence = " + last);
        }
        else
            System.out.println("Not Found");
    }

    public static void main(String[] args)
    {
        int arr[] = { 1, 2, 2, 2, 2, 3, 4, 7, 8, 8 };
        int x = 8;
        findFirstAndLast(arr, x);
    }
}
```

Output:



4.First and Transition Point

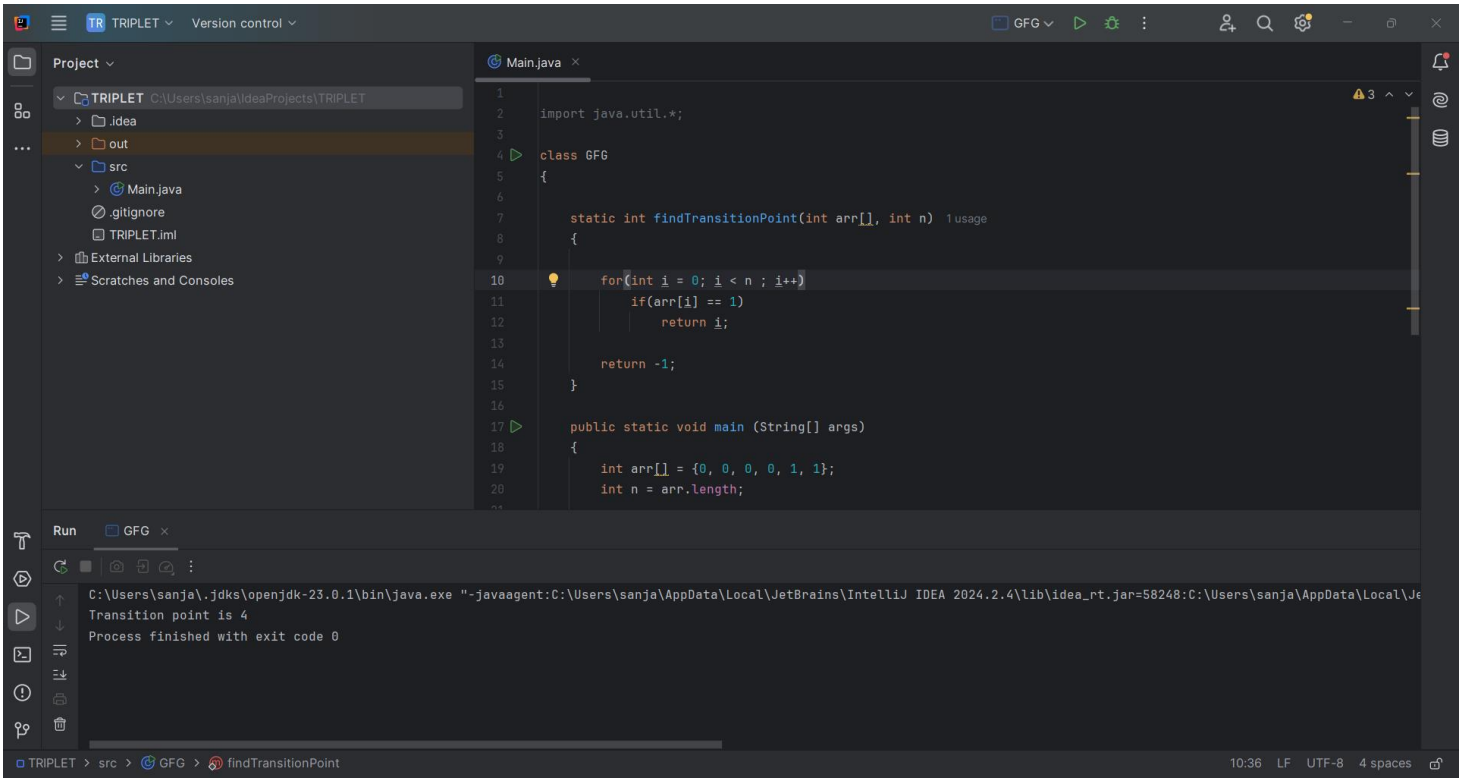
Time Complexity : O(n)

Solution:

```
import java.util.*;
class GFG
{

  static int findTransitionPoint(int arr[], int n)
  {

    for(int i = 0; i < n ; i++)
      if(arr[i] == 1)
        return i;

    return -1;
```

```
    }

    public static void main (String[] args)
    {
        int arr[] = {0, 0, 0, 0, 1, 1};
        int n = arr.length;

        int point = findTransitionPoint(arr, n);

        if (point >= 0)
            System.out.print("Transition point is " + point);
        else
            System.out.print("There is no transition point");
    }
}
```

Output:



5.First Repeating Element

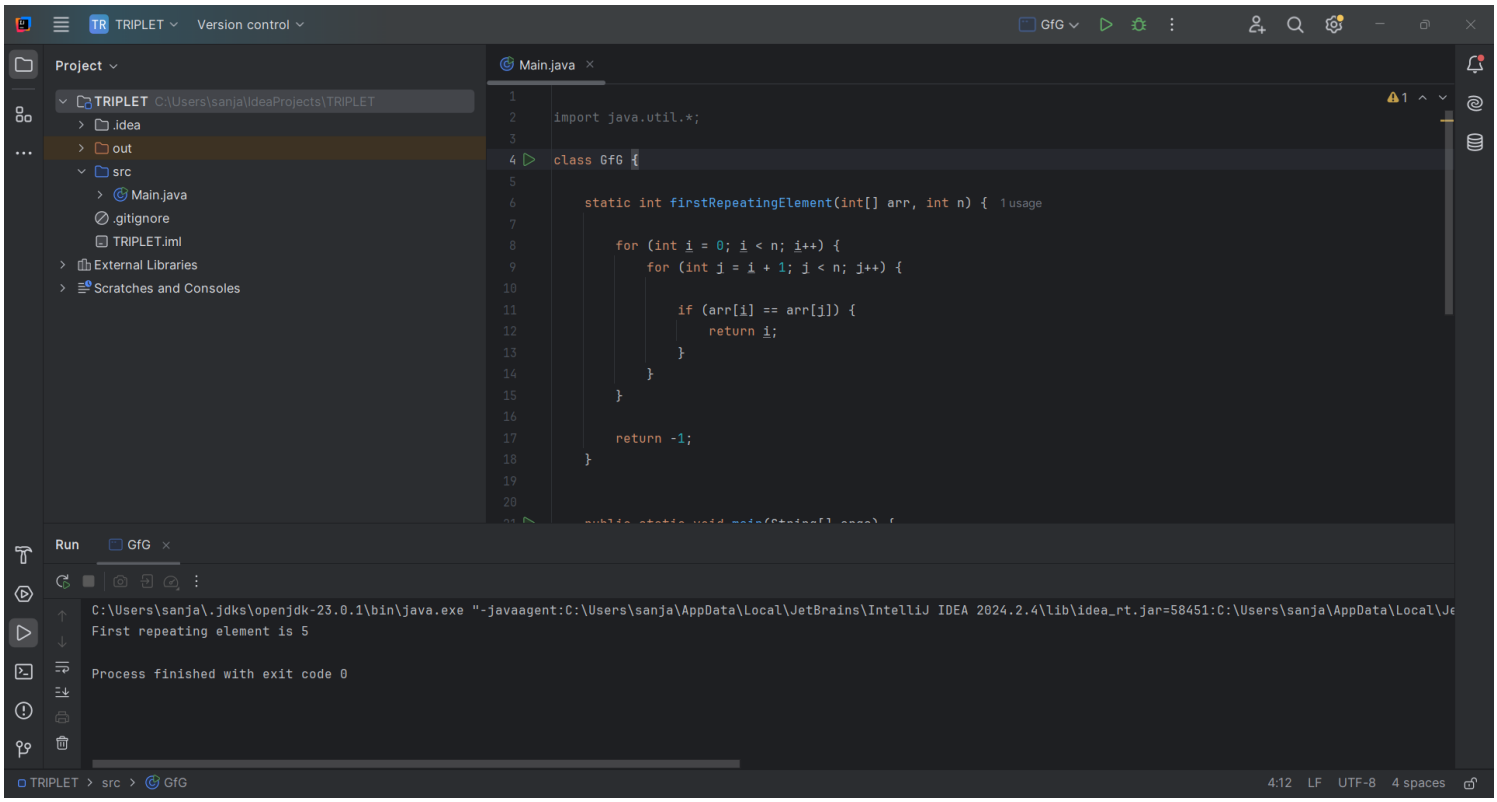
Time Complexity: O(n)

Solution:

```
import java.util.*;

class GfG {

    static int firstRepeatingElement(int[] arr, int n) {

        for (int i = 0; i < n; i++) {
            for (int j = i + 1; j < n; j++) {

                if (arr[i] == arr[j]) {
                    return i;
                }
            }
        }

        return -1;
    }


    public static void main(String[] args) {

        int[] arr = { 10, 5, 3, 4, 3, 5, 6 };
        int n = arr.length;

        int index = firstRepeatingElement(arr, n);


        if (index == -1) {
            System.out.println("No repeating element found!");
        }
```

```
        else {
            System.out.println("First repeating element is " + arr[index]);
        }
    }
}
```

Output:



6.Remove Duplicates sorted array

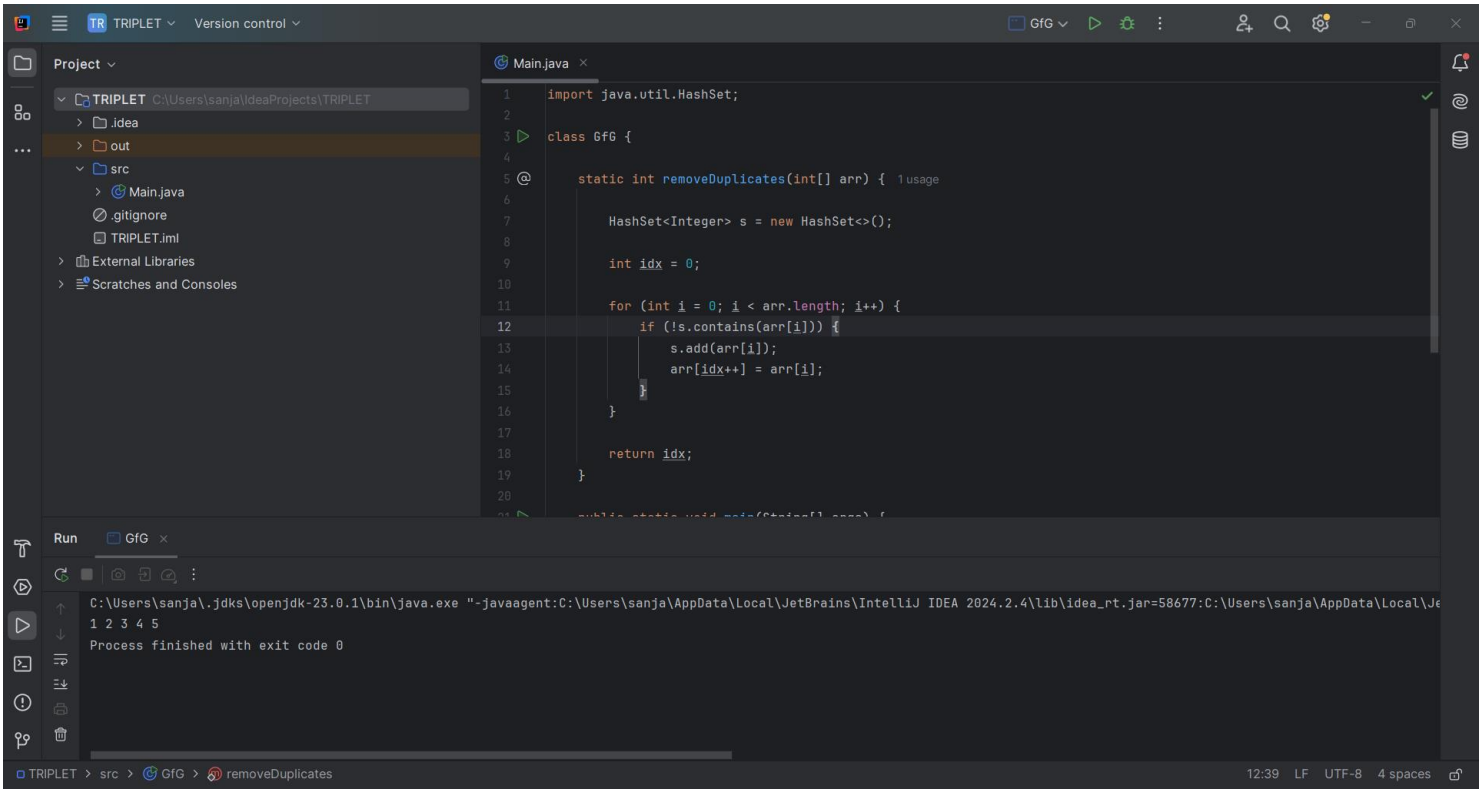
Time Complexity : O(n)

Solution:

```
import java.util.HashSet;

class GfG {

    static int removeDuplicates(int[] arr) {
```

```
    HashSet<Integer> s = new HashSet<>();

    int idx = 0;

    for (int i = 0; i < arr.length; i++) {
        if (!s.contains(arr[i])) {
            s.add(arr[i]);
            arr[idx++] = arr[i];
        }
    }

    return idx;
  }

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

    for (int i = 0; i < newSize; i++) {
        System.out.print(arr[i] + " ");
    }
  }
}
```

Output:



6.Maximum Index

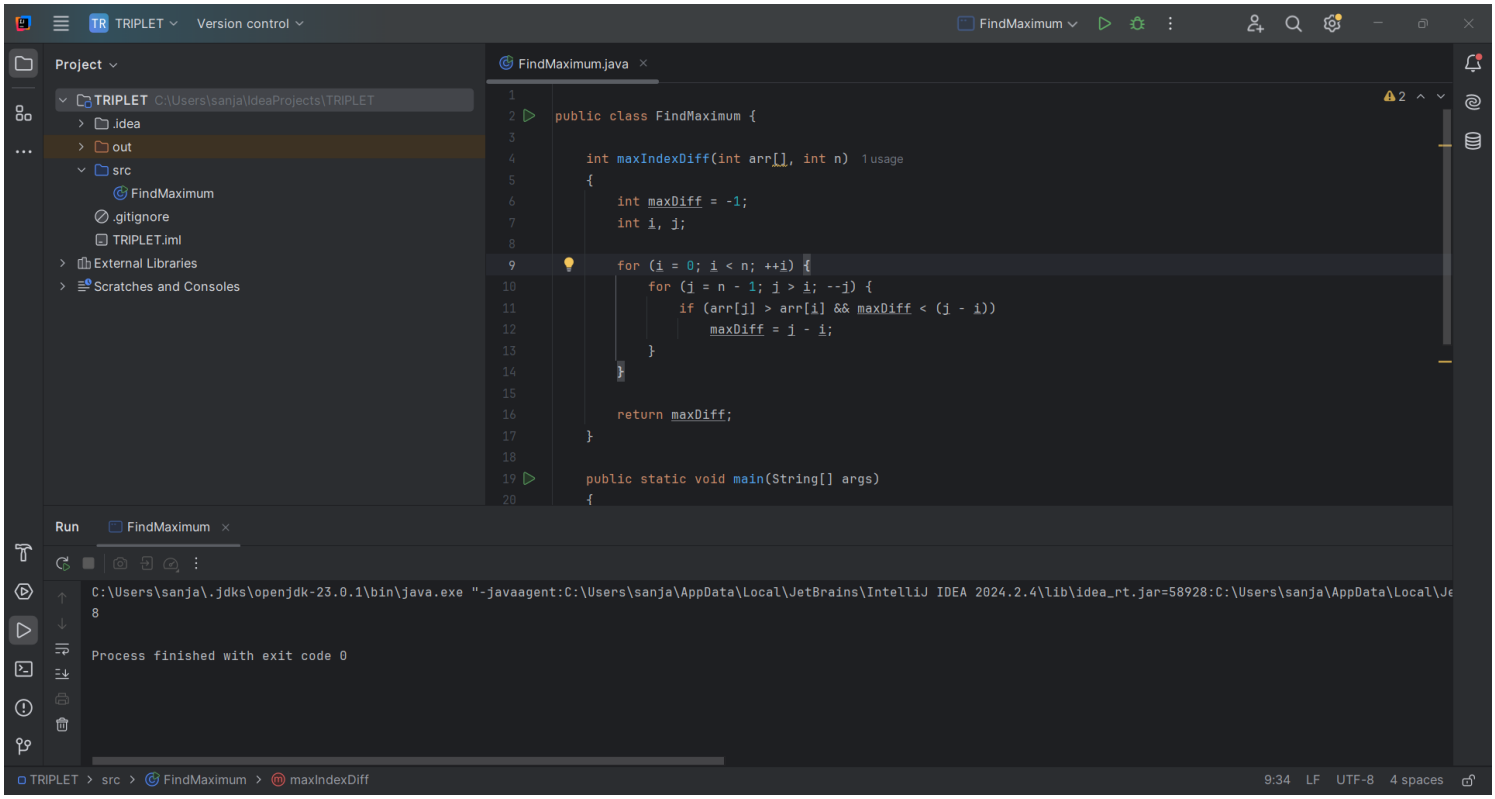
Time Complexity : $O(n^2)$

Solution:

```
public class FindMaximum {

  int maxIndexDiff(int arr[], int n)
  {
     int maxDiff = -1;
     int i, j;

     for (i = 0; i < n; ++i) {
        for (j = n - 1; j > i; --j) {
           if (arr[j] > arr[i] && maxDiff < (j - i))
              maxDiff = j - i;
```

```
        }
    }

    return maxDiff;
  }

  public static void main(String[] args)
  {
    FindMaximum max = new FindMaximum();
    int arr[] = { 9, 2, 3, 4, 5, 6, 7, 8, 18, 0 };
    int n = arr.length;
    int maxDiff = max.maxIndexDiff(arr, n);
    System.out.println(maxDiff);
  }
}
```

Output:



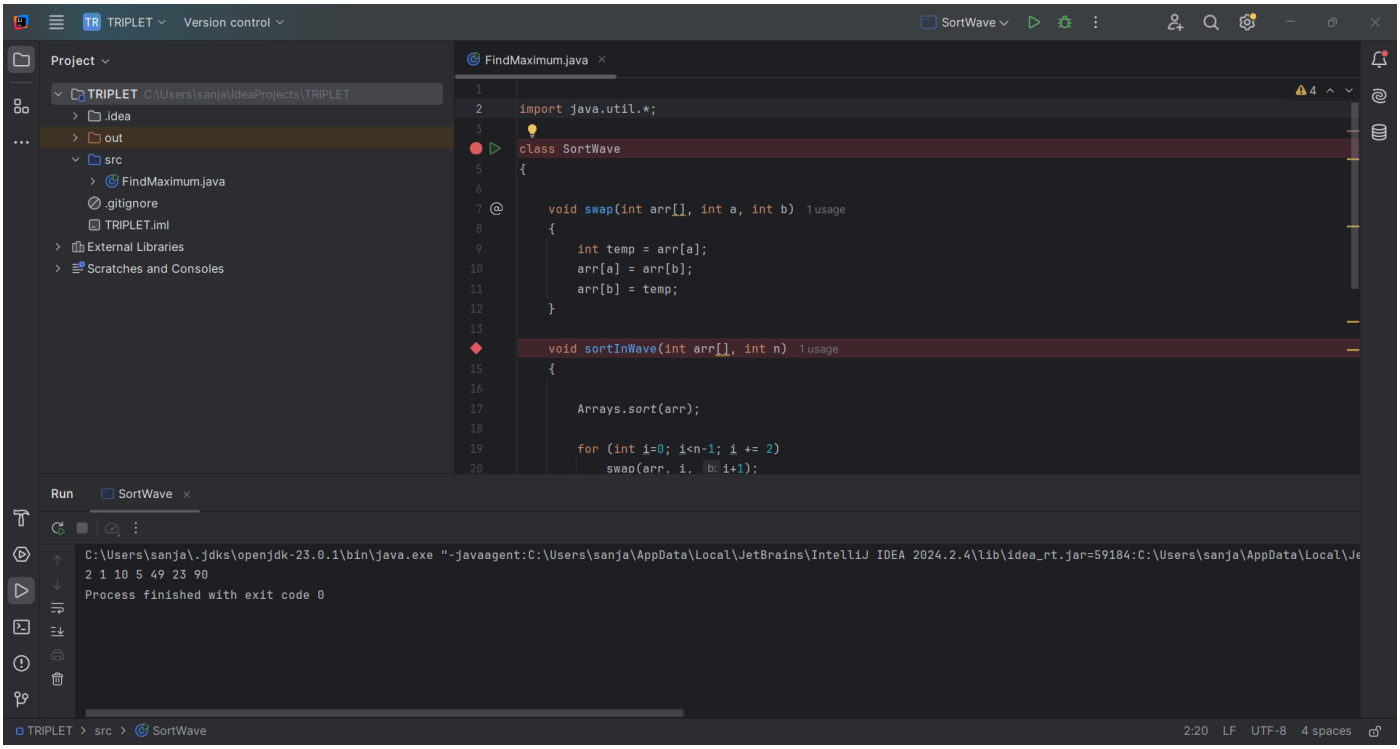
7.Wave Array

Time Complexity : O(n Log n)

Solution:

```
import java.util.*;

class SortWave
{

    void swap(int arr[], int a, int b)
    {
        int temp = arr[a];
        arr[a] = arr[b];
        arr[b] = temp;
    }

    void sortInWave(int arr[], int n)
    {

        Arrays.sort(arr);

        for (int i=0; i<n-1; i += 2)
            swap(arr, i, i+1);
    }
    public static void main(String args[])
    {
        SortWave ob = new SortWave();
        int arr[] = {10, 90, 49, 2, 1, 5, 23};
        int n = arr.length;
        ob.sortInWave(arr, n);
        for (int i : arr)
            System.out.print(i + " ");
    }
}
```

Output:


TRIPLET
Version control
SortWave
Project
TRIPLET C:\Users\sanja\IdeaProjects\TRIPLET
.idea
out
src
FindMaximum.java
.gitignore
TRIPLET.iml
External Libraries
Scratches and Consoles
FindMaximum.java
import java.util.*;
class SortWave
{
void swap(int arr[], int a, int b) 1 usage
{
int temp = arr[a];
arr[a] = arr[b];
arr[b] = temp;
}
void sortInWave(int arr[], int n) 1 usage
{
Arrays.sort(arr);
for (int i=0; i<n-1; i += 2)
swap(arr, i, i+1);
Run
SortWave
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=59184:C:\Users\sanja\AppData\Local\Je
2 1 10 5 49 23 90
Process finished with exit code 0
TRIPLET > src > SortWave
2:20 LF UTF-8 4 spaces