

VIJAY M(IT)

22IT125

CODING PRACTICES AND PROBLEMS

1.Kth Smallest Elements

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

SOLUTION:

```
import java.util.Arrays;

class GFG {

    public static int kthSmallest(Integer[] arr, int K)
    {

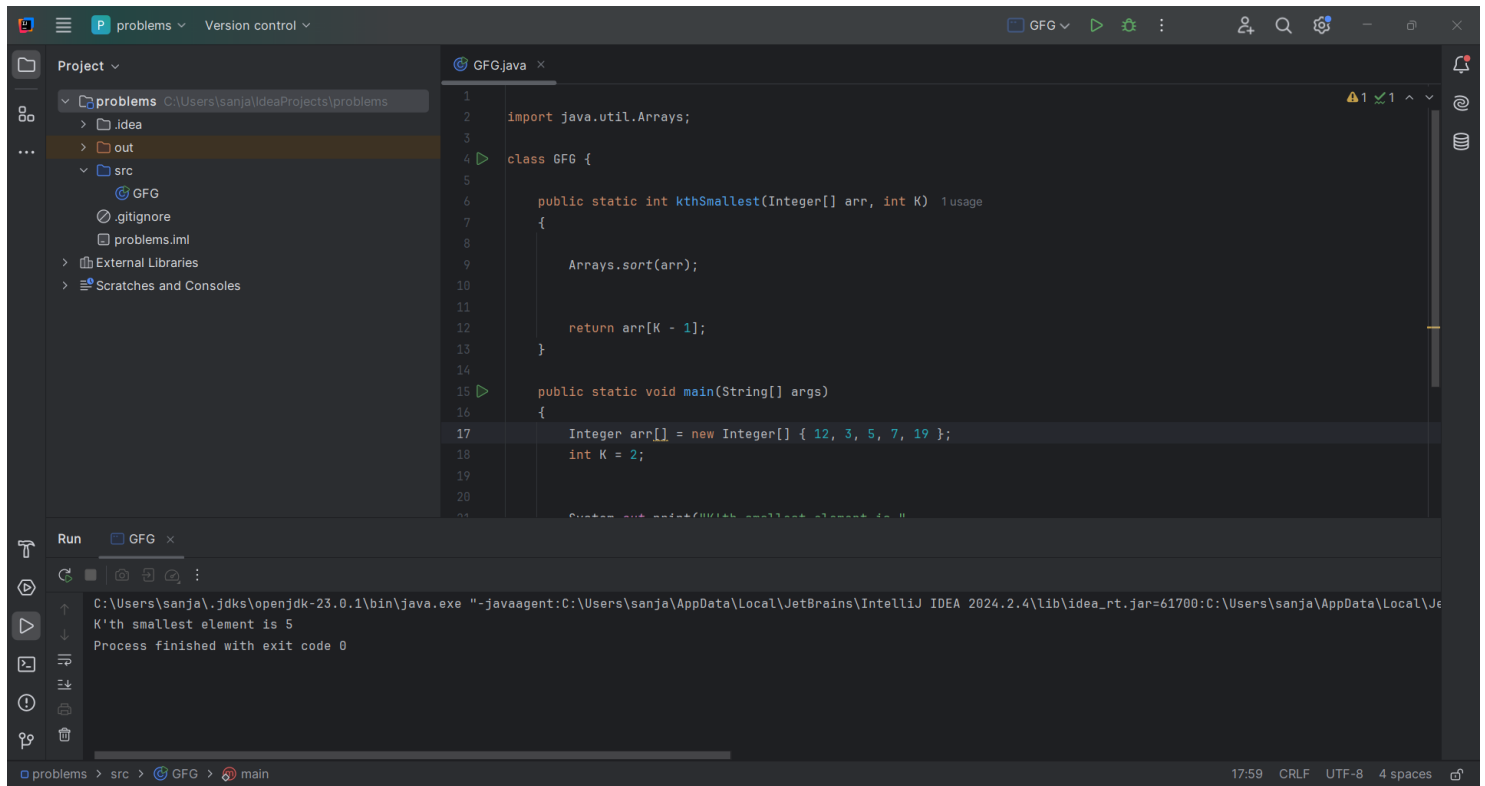
        Arrays.sort(arr);

        return arr[K - 1];
    }

    public static void main(String[] args)
    {
        Integer arr[] = new Integer[] { 12, 3, 5, 7, 19 };
        int K = 2;

        System.out.print("K'th smallest element is "
            + kthSmallest(arr, K));
    }
}
```

OUTPUT:



The screenshot shows an IDE with a project named 'problems'. The file 'GFG.java' is open, containing the following code:

```
1
2 import java.util.Arrays;
3
4 class GFG {
5
6     public static int kthSmallest(Integer[] arr, int K) 1 usage
7     {
8
9         Arrays.sort(arr);
10
11         return arr[K - 1];
12     }
13
14     public static void main(String[] args)
15     {
16
17         Integer arr[] = new Integer[] { 12, 3, 5, 7, 19 };
18         int K = 2;
19
20         System.out.printf("K'th smallest element is %d\n", kthSmallest(arr, K));
21     }
22 }
```

The 'Run' tab 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=61700:C:\Users\sanja\AppData\Local\Je
K'th smallest element is 5
Process finished with exit code 0
```

2.Minimize Height II

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

SOLUTION:

```
import java.util.Arrays;
```

```
class GfG {
```

```
static int getMinDiff(int[] arr, int k) {
```

```
    int n = arr.length;
```

```
    Arrays.sort(arr);
```

```
    int res = arr[n - 1] - arr[0];
```

```
    for (int i = 1; i < arr.length; i++) {
```

```
if (arr[i] - k < 0)
```

```
    continue;
```

```
int minH = Math.min(arr[0] + k, arr[i] - k);
```

```
int maxH = Math.max(arr[i - 1] + k, arr[n - 1] - k);
```

```
res = Math.min(res, maxH - minH);
```

```
}
```

```
return res;
```

```
}
```

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

```
    int k = 6;
```

```
    int[] arr = {12, 6, 4, 15, 17, 10};
```

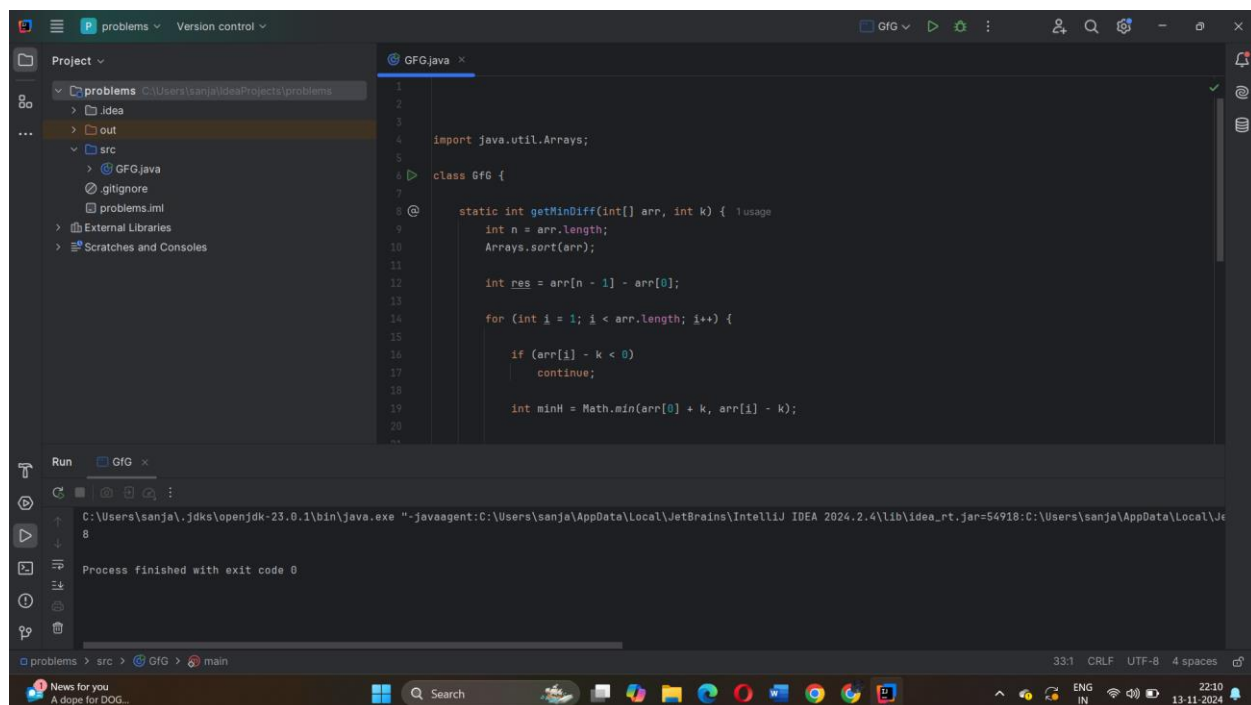
```
    int ans = getMinDiff(arr, k);
```

```
    System.out.println(ans);
```

```
}
```

```
}
```

OUTPUT:



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

```
1
2
3
4 import java.util.Arrays;
5
6 class GFG {
7
8     static int getMinDiff(int[] arr, int k) {
9         int n = arr.length;
10        Arrays.sort(arr);
11
12        int res = arr[n - 1] - arr[0];
13
14        for (int i = 1; i < arr.length; i++) {
15
16            if (arr[i] - k < 0)
17                continue;
18
19            int minH = Math.min(arr[0] + k, arr[i] - k);
20
21        }
22    }
23 }
```

The Run tab at the bottom shows the command executed: `C:\Users\sanja\jdk-23.0.1\bin\java.exe -javaagent:C:\Users\sanja\AppData\Local\JetBrains\IntelliJ IDEA 2024.2.4\lib\idea_rt.jar=54918:C:\Users\sanja\AppData\Local\J...`. The output is `8`, and the process finished with exit code 0.

3.Paranthesis Checker

Time Complexity:O(n)

SOLUTION:

```
import java.util.Stack;
public class Main {
    public static boolean ispar(String s) {

        Stack<Character> stk = new Stack<>();
        for (int i = 0; i < s.length(); i++) {

            if (s.charAt(i) == '(' || s.charAt(i) == '{' || s.charAt(i) == '[') {
                stk.push(s.charAt(i));
            }
            else {

                if (!stk.empty() &&
                    ((stk.peek() == '(' && s.charAt(i) == ')') ||
                     (stk.peek() == '{' && s.charAt(i) == '}') ||
                     (stk.peek() == '[' && s.charAt(i) == ']'))) {
                    stk.pop();
                }
                else {
                    return false;
                }
            }
        }

        return stk.empty();
    }

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

```

String s = "{()}[]";
if (ispar(s))
    System.out.println("true");
else
    System.out.println("false");
}
}

```

OUTPUT:

The screenshot shows an IDE with a project named 'main'. The source code in 'Main.java' implements a function 'ispar' using a stack to check if a string of parentheses is valid. The code is as follows:

```

1 import java.util.Stack;
2
3 public class Main {
4     @
5     public static boolean ispar(String s) {
6         Stack<Character> stk = new Stack<>();
7         for (int i = 0; i < s.length(); i++) {
8
9             if (s.charAt(i) == '(' || s.charAt(i) == '{' || s.charAt(i) == '[') {
10                 stk.push(s.charAt(i));
11             }
12             else {
13
14                 if (!stk.empty() &&
15                     ((stk.peek() == '(' && s.charAt(i) == ')') ||
16                      (stk.peek() == '{' && s.charAt(i) == '}') ||
17                      (stk.peek() == '[' && s.charAt(i) == ']'))) {
18                     stk.pop();
19                 }
20                 else {
21                     return false;
22                 }
23             }
24         }
25         return true;
26     }
27 }

```

The Run window 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=56139:C:\Users\sanja\AppData\Local\J...` and the output: `true`. The process finished with exit code 0.

4. Equilibrium Point:

Time complexity: $O(N^2)$

SOLUTION:

```
public class Main {

    public static int equilibriumPoint(long[] arr)
    {
        int n = arr.length;
        long leftsum, rightsum;

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

            leftsum = 0;
            for (int j = 0; j < i; j++)
                leftsum += arr[j];

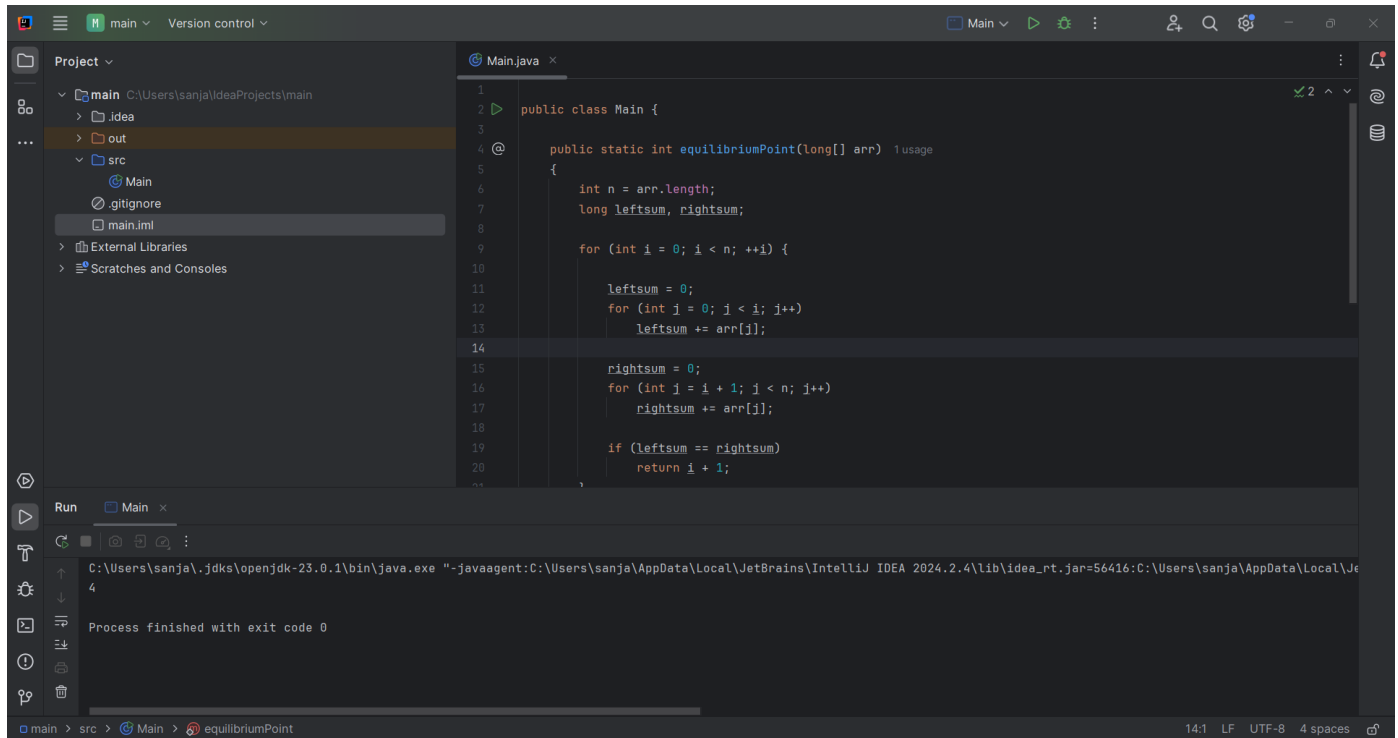
            rightsum = 0;
            for (int j = i + 1; j < n; j++)
                rightsum += arr[j];

            if (leftsum == rightsum)
                return i + 1;
        }
        return -1;
    }

    public static void main(String[] args)
    {
        long[] arr = { -7, 1, 5, 2, -4, 3, 0 };

        System.out.println(equilibriumPoint(arr));
    }
}
```

OUTPUT:



The screenshot shows an IDE with a project named 'main'. The source code in 'Main.java' defines a method `equilibriumPoint` that finds the index of an element in an array such that the sum of elements to its left equals the sum of elements to its right. The code is as follows:

```
1 public class Main {
2
3
4     public static int equilibriumPoint(long[] arr) 1 usage
5     {
6         int n = arr.length;
7         long leftsum, rightsum;
8
9         for (int i = 0; i < n; ++i) {
10
11             leftsum = 0;
12             for (int j = 0; j < i; j++)
13                 leftsum += arr[j];
14
15             rightsum = 0;
16             for (int j = i + 1; j < n; j++)
17                 rightsum += arr[j];
18
19             if (leftsum == rightsum)
20                 return i + 1;
21         }
22     }
23 }
```

The 'Run' window 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=56416:C:\Users\sanja\AppData\Local\J...` and the output: `4`. A status bar at the bottom indicates the file is at `main > src > Main > equilibriumPoint` with a cursor at line 14, column 1.

5.Binary Search:

Time Complexity: $O(\log N)$

SOLUTION:

```
class BinarySearch {
```

```
    int binarySearch(int arr[], int x)
```

```
    {
```

```
        int low = 0, high = arr.length - 1;
```

```
        while (low <= high) {
```

```
            int mid = low + (high - low) / 2;
```

```
            if (arr[mid] == x)
```

```
                return mid;
```

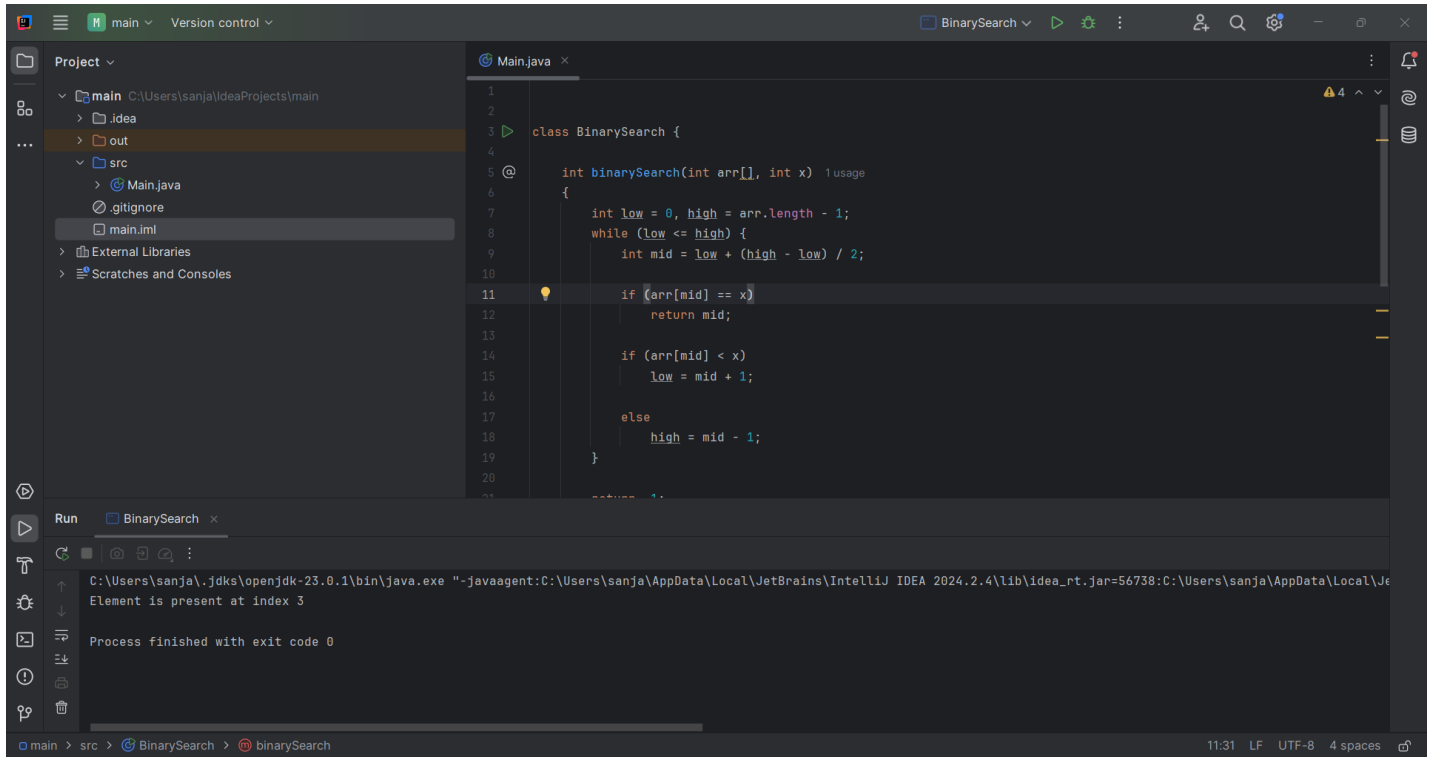
```
    if (arr[mid] < x)
        low = mid + 1;
```

```
    else
        high = mid - 1;
}
```

```
return -1;
}
```

```
public static void main(String args[])
{
    BinarySearch ob = new BinarySearch();
    int arr[] = { 2, 3, 4, 10, 40 };
    int n = arr.length;
    int x = 10;
    int result = ob.binarySearch(arr, x);
    if (result == -1)
        System.out.println(
            "Element is not present in array");
    else
        System.out.println("Element is present at "
            + "index " + result);
}
}
```


OUTPUT:



The screenshot shows the IntelliJ IDEA IDE. The 'Project' view on the left shows a project named 'main' with a source folder 'src' containing 'Main.java' and 'main.iml'. The 'Main.java' file is open in the editor, showing a 'BinarySearch' class with a 'binarySearch' method. The method uses a while loop to find the index of an element 'x' in an array 'arr'. The 'Run' view at the bottom shows the execution of 'BinarySearch' with the command '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=56738:C:\Users\sanja\AppData\Local\J...'. The output is 'Element is present at index 3' and 'Process finished with exit code 0'.

```
1
2
3 class BinarySearch {
4
5     int binarySearch(int arr[], int x) {
6         int low = 0, high = arr.length - 1;
7         while (low <= high) {
8             int mid = low + (high - low) / 2;
9
10            if (arr[mid] == x)
11                return mid;
12
13            if (arr[mid] < x)
14                low = mid + 1;
15
16            else
17                high = mid - 1;
18        }
19
20        return -1;
21    }
22 }
```

Run BinarySearch

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=56738:C:\Users\sanja\AppData\Local\J...
Element is present at index 3

Process finished with exit code 0

6.Next Greater Element: Time complexity:O(n2)

SOLUTION:

```
class Main {
```

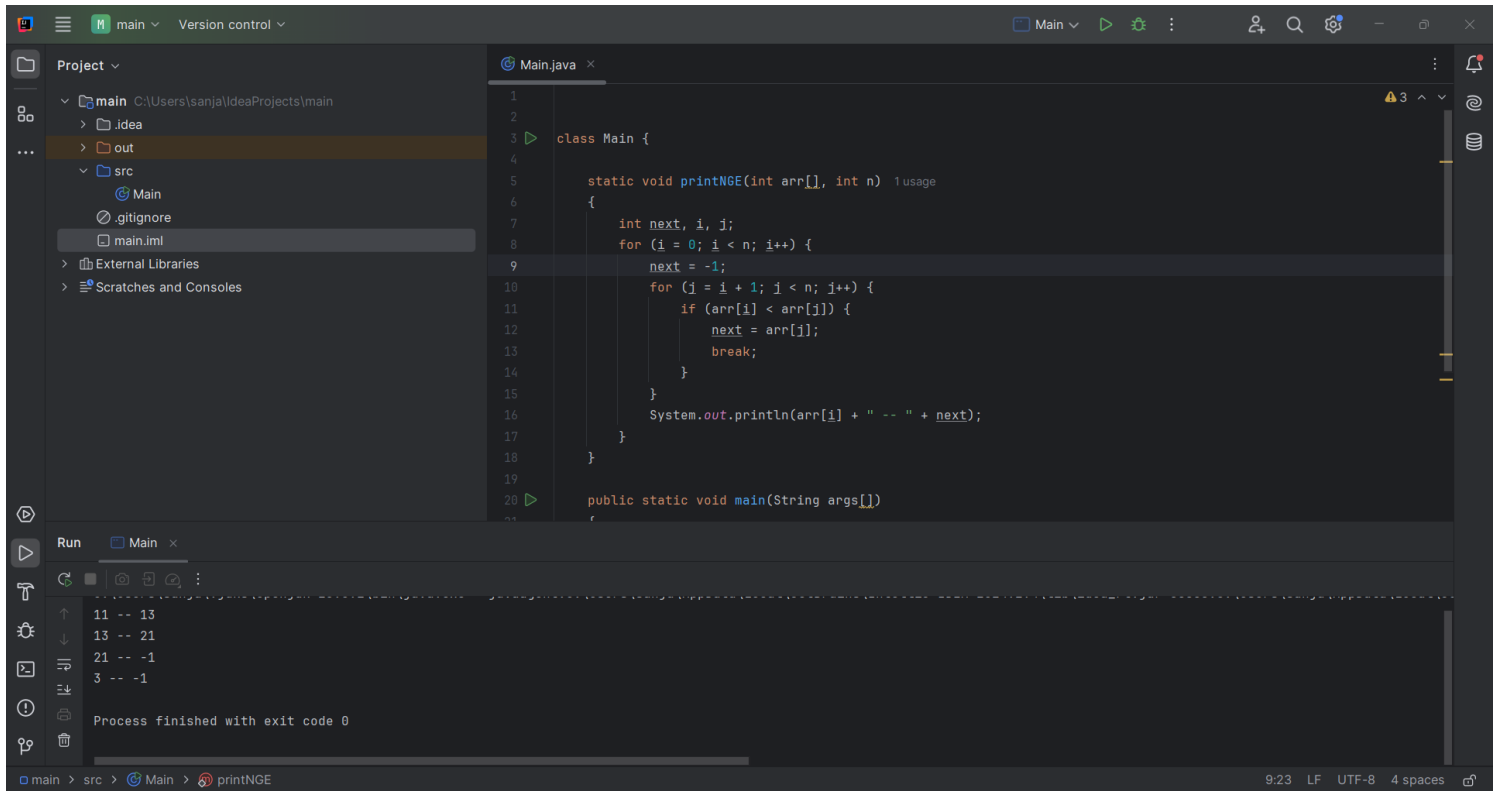
```
static void printNGE(int arr[], int n)
{
    int next, i, j;
    for (i = 0; i < n; i++) {
        next = -1;
```

```
        for (j = i + 1; j < n; j++) {  
            if (arr[i] < arr[j]) {  
                next = arr[j];  
                break;  
            }  
            System.out.println(arr[i] + " -- " + next);  
        }  
    }
```

```
public static void main(String args[])  
{  
    int arr[] = { 11, 13, 21, 3 };  
    int n = arr.length;  
    printNGE(arr, n);  
}
```

```
}
```

OUTPUT:



The screenshot shows an IDE with a project named 'main'. The 'Project' view on the left shows the file structure: 'main' (C:\Users\sanja\IdeaProjects\main) containing '.idea', 'out', 'src', 'Main', '.gitignore', and 'main.iml'. The 'src' folder contains 'Main'. The 'Main.java' file is open in the editor, showing the following code:

```
1
2
3 class Main {
4
5     static void printNGE(int arr[], int n) {usage
6     {
7         int next, i, j;
8         for (i = 0; i < n; i++) {
9             next = -1;
10            for (j = i + 1; j < n; j++) {
11                if (arr[i] < arr[j]) {
12                    next = arr[j];
13                    break;
14                }
15            }
16            System.out.println(arr[i] + " -- " + next);
17        }
18    }
19
20    public static void main(String args[])
21    {
```

The 'Run' view at the bottom shows the output of the program:

```
11 -- 13
13 -- 21
21 -- -1
3 -- -1
```

Below the output, it says 'Process finished with exit code 0'.

7. Union Of Two Arrays:

Time Complexity: $O(n*m)$

SOLUTION:

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

```
class GfG {
```

```
static ArrayList<Integer> findUnion(int[] a, int[] b) {
    int n = a.length, m = b.length;
    ArrayList<Integer> res = new ArrayList<>();
```

```
for (int i = 0; i < n; i++) {  
    res.add(a[i]);  
}
```

```
for (int i = 0; i < m; i++) {
```

```
    int j;  
    for (j = 0; j < n; j++) {  
        if (a[j] == b[i])  
            break;  
    }
```

```
    if (j == n) {  
        res.add(b[i]);  
    }  
}
```

```
res.sort(null);  
return res;  
}
```

```
public static void main(String[] args) {  
    int[] a = {1, 2, 3};  
    int[] b = {2, 5, 7};
```

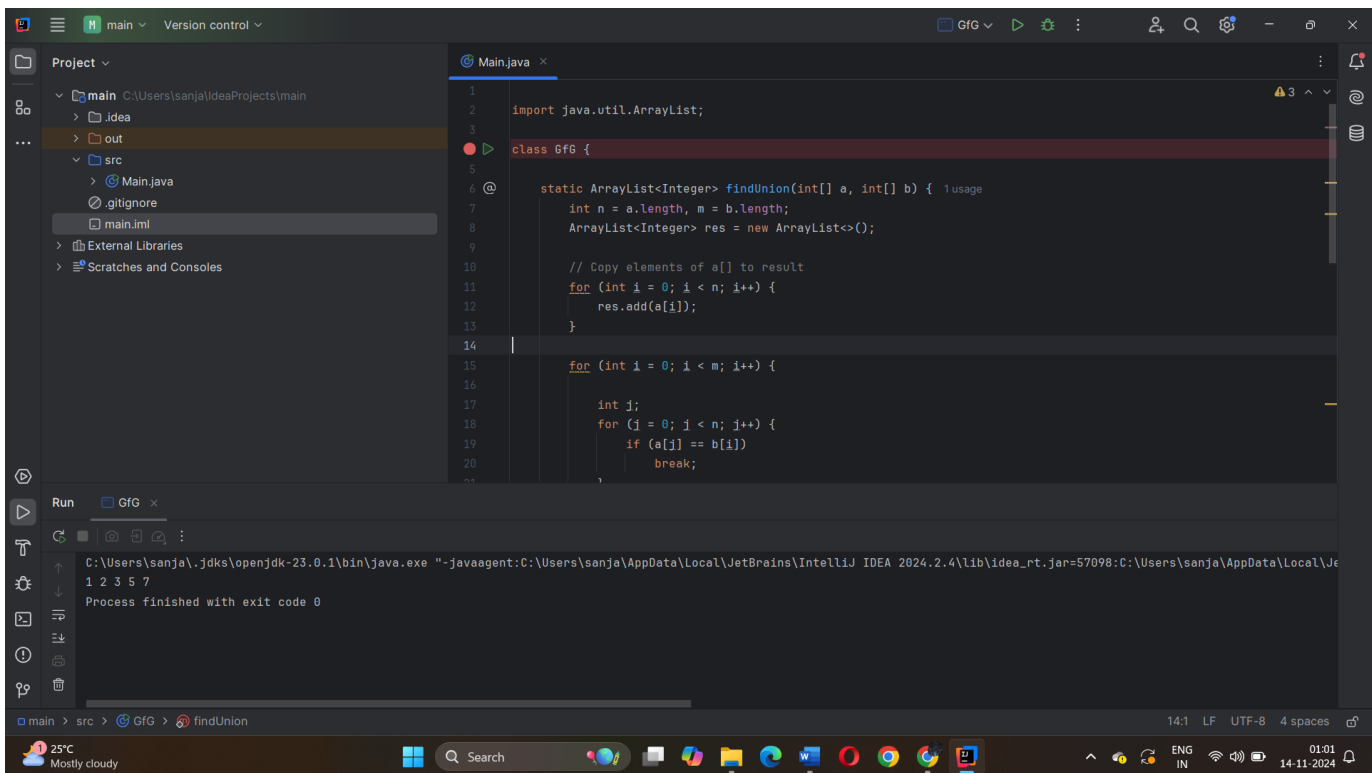
```
ArrayList<Integer> res = findUnion(a, b);
```

```
for (int i = 0; i < res.size(); i++)  
    System.out.print(res.get(i) + " ");
```

```
}
```

```
}
```

OUTPUT:



The screenshot shows an IDE with a project named 'main' and a file 'Main.java'. The code in 'Main.java' defines a class 'GFG' with a static method 'findUnion' that takes two integer arrays 'a' and 'b' and returns an 'ArrayList<Integer>'. The method implementation is as follows:

```
1  
2 import java.util.ArrayList;  
3  
4 class GFG {  
5  
6     static ArrayList<Integer> findUnion(int[] a, int[] b) { //usage  
7         int n = a.length, m = b.length;  
8         ArrayList<Integer> res = new ArrayList<>();  
9  
10        // Copy elements of a[] to result  
11        for (int i = 0; i < n; i++) {  
12            res.add(a[i]);  
13        }  
14  
15        for (int i = 0; i < m; i++) {  
16  
17            int j;  
18            for (j = 0; j < n; j++) {  
19                if (a[j] == b[i])  
20                    break;  
21            }  
22        }  
23    }  
24 }
```

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=57098:C:\Users\sanja\AppData\Local\Jc`. The output is: `1 2 3 5 7`. The status bar at the bottom indicates the file is 'Main.java' in the 'src' directory, with a line number of 14, and the encoding is 'UTF-8'.