

VIJAY M

22IT125

Day-3

CODING PRACTICES AND PROBLEM

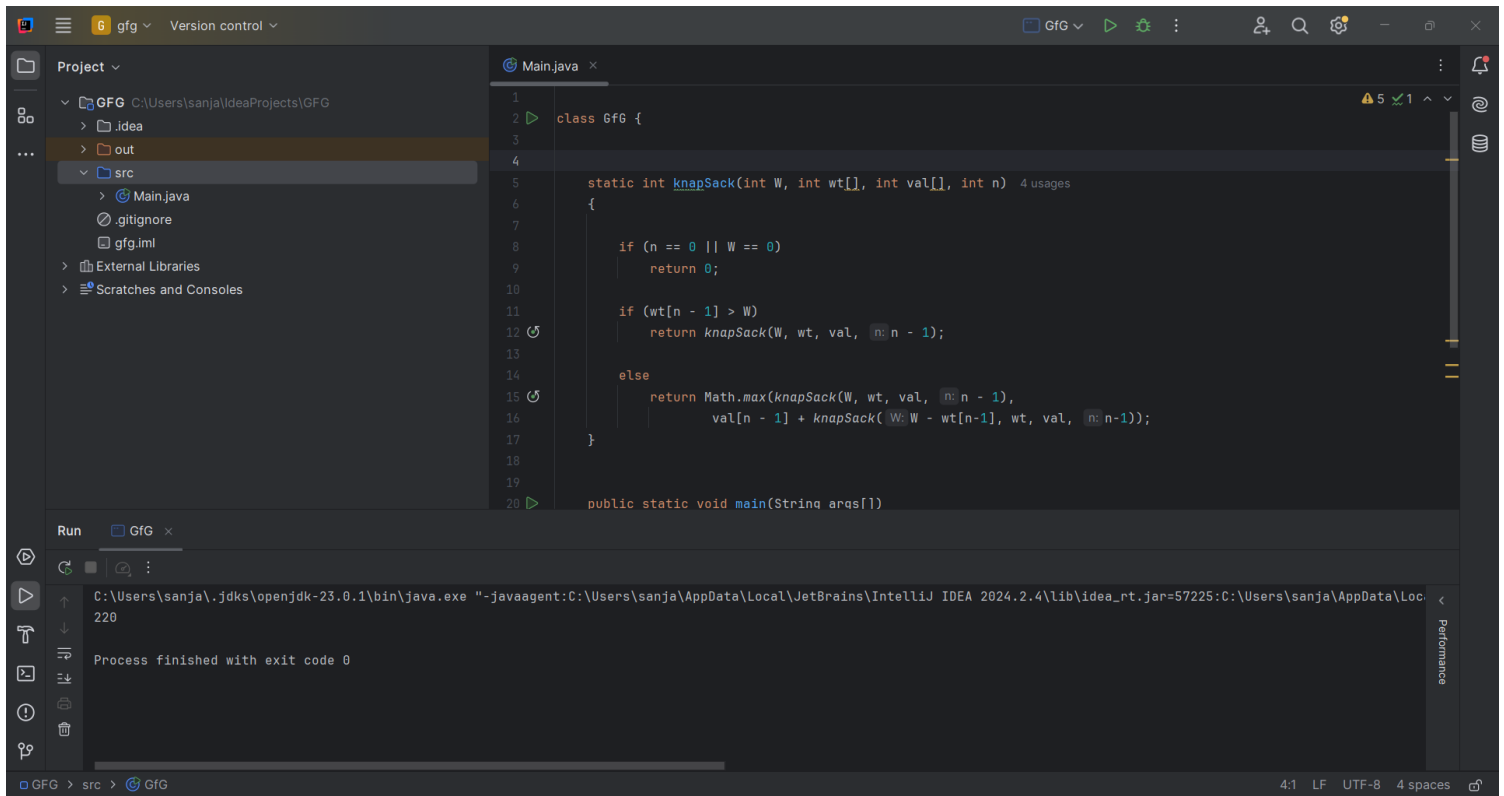
1.0-1 Knapsack Problem

Time Complexity : $O(2N)$

Solution:

```
class GfG {  
  
    static int knapSack(int W, int wt[], int val[], int n)  
    {  
  
        if (n == 0 || W == 0)  
            return 0;  
  
        if (wt[n - 1] > W)  
            return knapSack(W, wt, val, n - 1);  
  
        else  
            return Math.max(knapSack(W, wt, val, n - 1),  
                           val[n - 1] + knapSack(W - wt[n-1], wt, val, n-1));  
    }  
  
    public static void main(String args[])  
    {  
        int profit[] = new int[] { 60, 100, 120 };  
        int weight[] = new int[] { 10, 20, 30 };  
        int W = 50;  
        int n = profit.length;  
        System.out.println(knapSack(W, weight, profit, n));  
    }  
}
```

Output:



The screenshot shows an IDE with a project named 'GFG'. The 'src' folder contains 'Main.java'. The code implements a recursive knapsack solution. The 'Run' tab shows the program executed successfully with exit code 0. The console output is '220'.

```
1 class GfG {
2
3
4
5 static int knapSack(int W, int wt[], int val[], int n) 4 usages
6 {
7     if (n == 0 || W == 0)
8         return 0;
9
10    if (wt[n - 1] > W)
11        return knapSack(W, wt, val, n - 1);
12
13    else
14        return Math.max(knapSack(W, wt, val, n - 1),
15                        val[n - 1] + knapSack(W - wt[n-1], wt, val, n-1));
16
17 }
18
19
20 public static void main(String args[])
```

2.Floor in Sorted Array:

Time Complexity : $O(N)$

Solution:

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

```
class GFG {
```

```
static int floorSearch(int arr[], int n, int x)
{
```

```
    if (x >= arr[n - 1])
        return n - 1;
```

```
if (x < arr[0])  
    return -1;
```

```
for (int i = 1; i < n; i++)  
    if (arr[i] > x)  
        return (i - 1);
```

```
return -1;
```

```
}
```

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

```
{
```

```
    int arr[] = { 1, 2, 4, 6, 10, 12, 14 };
```

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

```
    int x = 7;
```

```
    int index = floorSearch(arr, n - 1, x);
```

```
    if (index == -1)
```

```
        System.out.print("Floor of " + x  
            + " doesn't exist in array ");
```

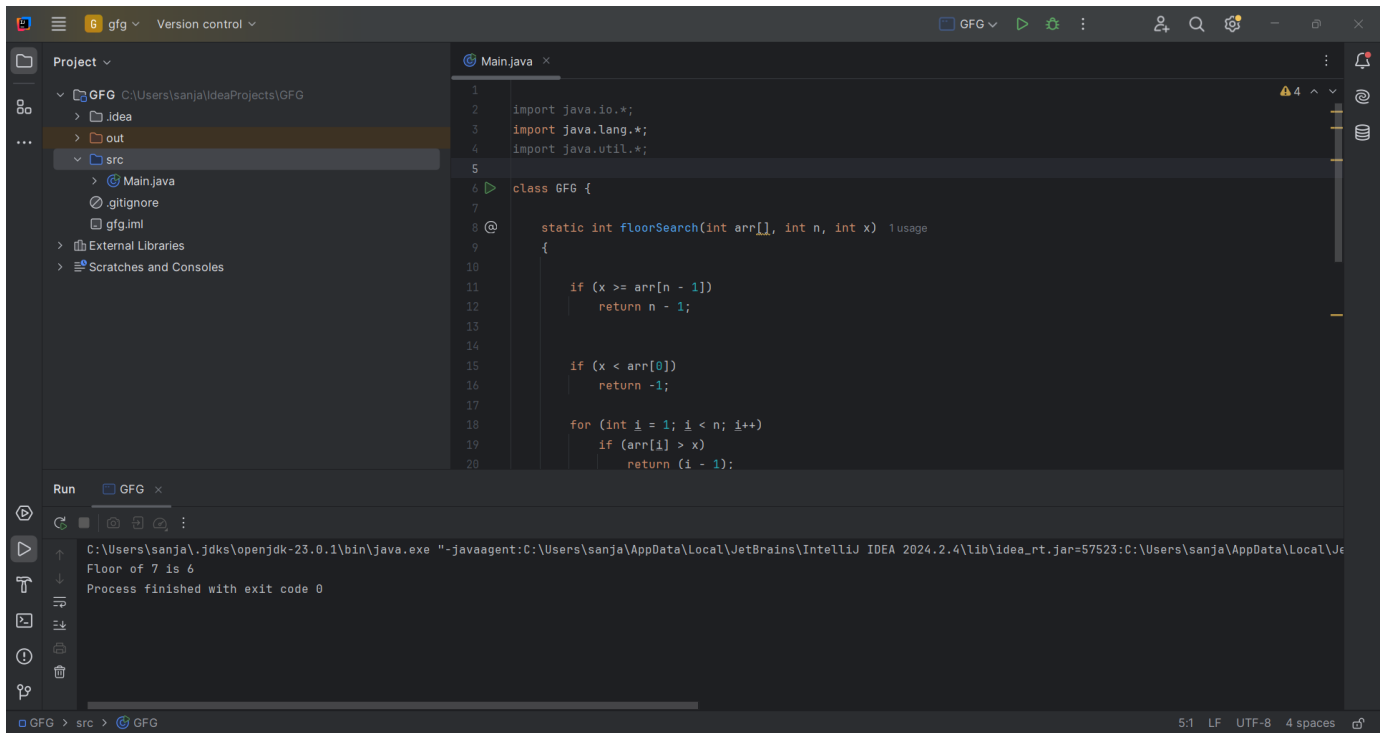
```
    else
```

```
        System.out.print("Floor of " + x + " is "  
            + arr[index]);
```

```
}
```

```
}
```

Output:



The screenshot shows an IDE with a project named 'GFG'. The 'Main.java' file is open, containing a static method 'floorSearch' that takes an array 'arr', its length 'n', and a target value 'x'. The method returns the index of the largest element less than or equal to 'x'. The console output shows the result of a search for '7' in an array, returning '6'.

```
1 import java.io.*;
2 import java.lang.*;
3 import java.util.*;
4
5 class GFG {
6
7     static int floorSearch(int arr[], int n, int x) {
8         if (x >= arr[n - 1])
9             return n - 1;
10
11         if (x < arr[0])
12             return -1;
13
14         for (int i = 1; i < n; i++)
15             if (arr[i] > x)
16                 return (i - 1);
17     }
18 }
19
20 public static void main (String[] args) {
21     // Example usage
22     int arr[] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};
23     int n = arr.length;
24     int x = 7;
25     int result = floorSearch(arr, n, x);
26     System.out.println("Floor of " + x + " is " + result);
27 }
```

Run 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=57523:C:\Users\sanja\AppData\Local\Je
Floor of 7 is 6
Process finished with exit code 0

3.Check Equal Arrays:

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

Solution:

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

```
class GFG {
```

```
    public static boolean areEqual(int arr1[], int arr2[])
    {
```

```
        int N = arr1.length;
        int M = arr2.length;
```

```
        if (N != M)
            return false;
```

```

Map<Integer, Integer> map
    = new HashMap<Integer, Integer>();
int count = 0;
for (int i = 0; i < N; i++) {
    if (map.get(arr1[i]) == null)
        map.put(arr1[i], 1);
    else {
        count = map.get(arr1[i]);
        count++;
        map.put(arr1[i], count);
    }
}

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

    if (!map.containsKey(arr2[i]))
        return false;

    if (map.get(arr2[i]) == 0)
        return false;

    count = map.get(arr2[i]);
    --count;
    map.put(arr2[i], count);
}

return true;
}

public static void main(String[] args)
{
    int arr1[] = { 3, 5, 2, 5, 2 };
    int arr2[] = { 2, 3, 5, 5, 2 };

```

```

    if (areEqual(arr1, arr2))
        System.out.println("Yes");
    else
        System.out.println("No");
}
}

```

Output:

The screenshot shows an IDE window with a project named 'GFG'. The 'src' folder contains a file 'Main.java'. The code in 'Main.java' is as follows:

```

1
2 import java.io.*;
3 import java.util.*;
4
5 class GFG {
6
7     public static boolean areEqual(int arr1[], int arr2[]) {
8         {
9             int N = arr1.length;
10            int M = arr2.length;
11
12            if (N != M)
13                return false;
14
15            Map<Integer, Integer> map
16                = new HashMap<Integer, Integer>();
17            int count = 0;
18            for (int i = 0; i < N; i++) {
19                if (map.get(arr1[i]) == null)
20                    map.put(arr1[i], 1);

```

The 'Run' tab shows the execution command and 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=57794:C:\Users\sanja\AppData\Local\J
Yes
Process finished with exit code 0

```

The status bar at the bottom indicates the file is 'GFG' in the 'src' folder, with a line length of 44, 1 line feed (LF), UTF-8 encoding, and 4 spaces for indentation. The system tray shows a temperature of 87°F, a 'Haze' weather icon, and the time 13:39 on 20-11-2024.

4. Palindrome Linked List:

Time Complexity : $O(n)$

Solution:

```
class Node {
    int data;
    Node next;
    Node(int d) {
        data = d;
        next = null;
    }
}

class GfG {

    static Node reverseList(Node head) {
        Node prev = null;
        Node curr = head;
        Node next;

        while (curr != null) {
            next = curr.next;
            curr.next = prev;
            prev = curr;
            curr = next;
        }
        return prev;
    }

    static boolean isIdentical(Node n1, Node n2) {
        while (n1 != null && n2 != null) {
            if (n1.data != n2.data)
                return false;
            n1 = n1.next;
        }
    }
}
```

```
        n2 = n2.next;
    }
    return true;
}
```

```
static boolean isPalindrome(Node head) {
    if (head == null || head.next == null)
        return true;
```

```
    Node slow = head, fast = head;
```

```
    while (fast.next != null
        && fast.next.next != null) {
        slow = slow.next;
        fast = fast.next.next;
    }
```

```
    Node head2 = reverseList(slow.next);
    slow.next = null;
```

```
    boolean ret = isIdentical(head, head2);
```

```
    head2 = reverseList(head2);
    slow.next = head2;
```

```
    return ret;
}
```

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

```
    Node head = new Node(1);
    head.next = new Node(2);
    head.next.next = new Node(3);
    head.next.next.next = new Node(2);
    head.next.next.next.next = new Node(1);
```



```
boolean result = isPalindrome(head);
```

```
if (result)
```

```
    System.out.println("true");
```

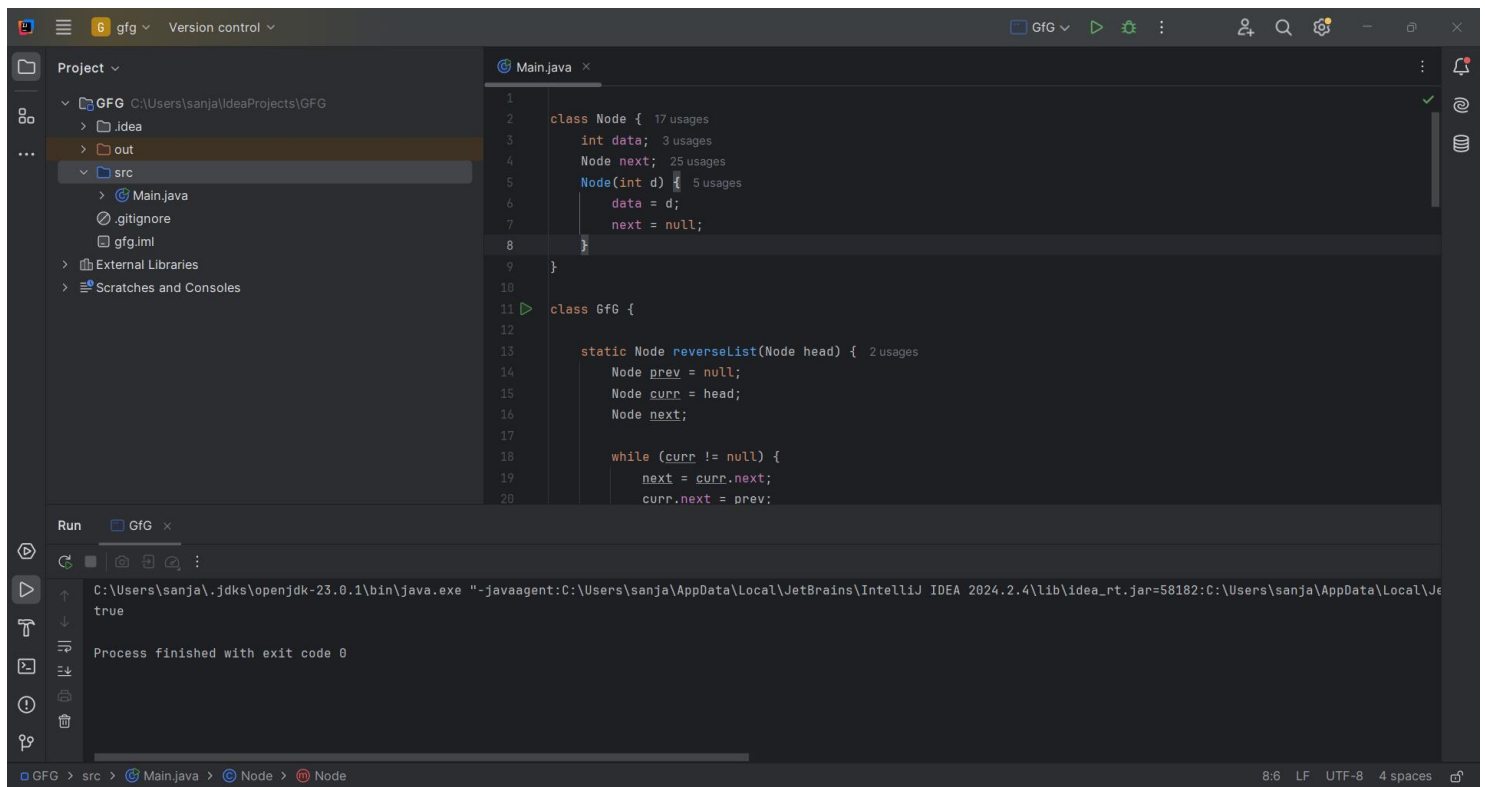
```
else
```

```
    System.out.println("false");
```

```
}
```

```
}
```

Output:



The screenshot shows an IDE window with a project named 'GFG'. The 'src' folder contains 'Main.java' and 'gfg.iml'. The 'Main.java' file is open, showing the following code:

```
1 class Node {
2     int data;
3     Node next;
4     Node(int d) {
5         data = d;
6         next = null;
7     }
8 }
9
10
11 class GFG {
12
13     static Node reverseList(Node head) {
14         Node prev = null;
15         Node curr = head;
16         Node next;
17
18         while (curr != null) {
19             next = curr.next;
20             curr.next = prev;
```

The 'Run' tab shows the execution of the program. The command executed is:

```
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=58182:C:\Users\sanja\AppData\Local\Je
```

The output is:

```
true
```

The process finished with exit code 0.

5.Balanced Tree Check:

Time Complexity : $O(n^2)$

Solution:

```
class Node {
    int data;
    Node left, right;
    Node(int d)
    {
        data = d;
        left = right = null;
    }
}

class BinaryTree {
    Node root;

    boolean isBalanced(Node node)
    {
        if (node == null)
            return true;

        lh = height(node.left);
        rh = height(node.right);

        if (Math.abs(lh - rh) <= 1 && isBalanced(node.left)
            && isBalanced(node.right))
            return true;

        return false;
    }
}
```

```
int height(Node node)
{

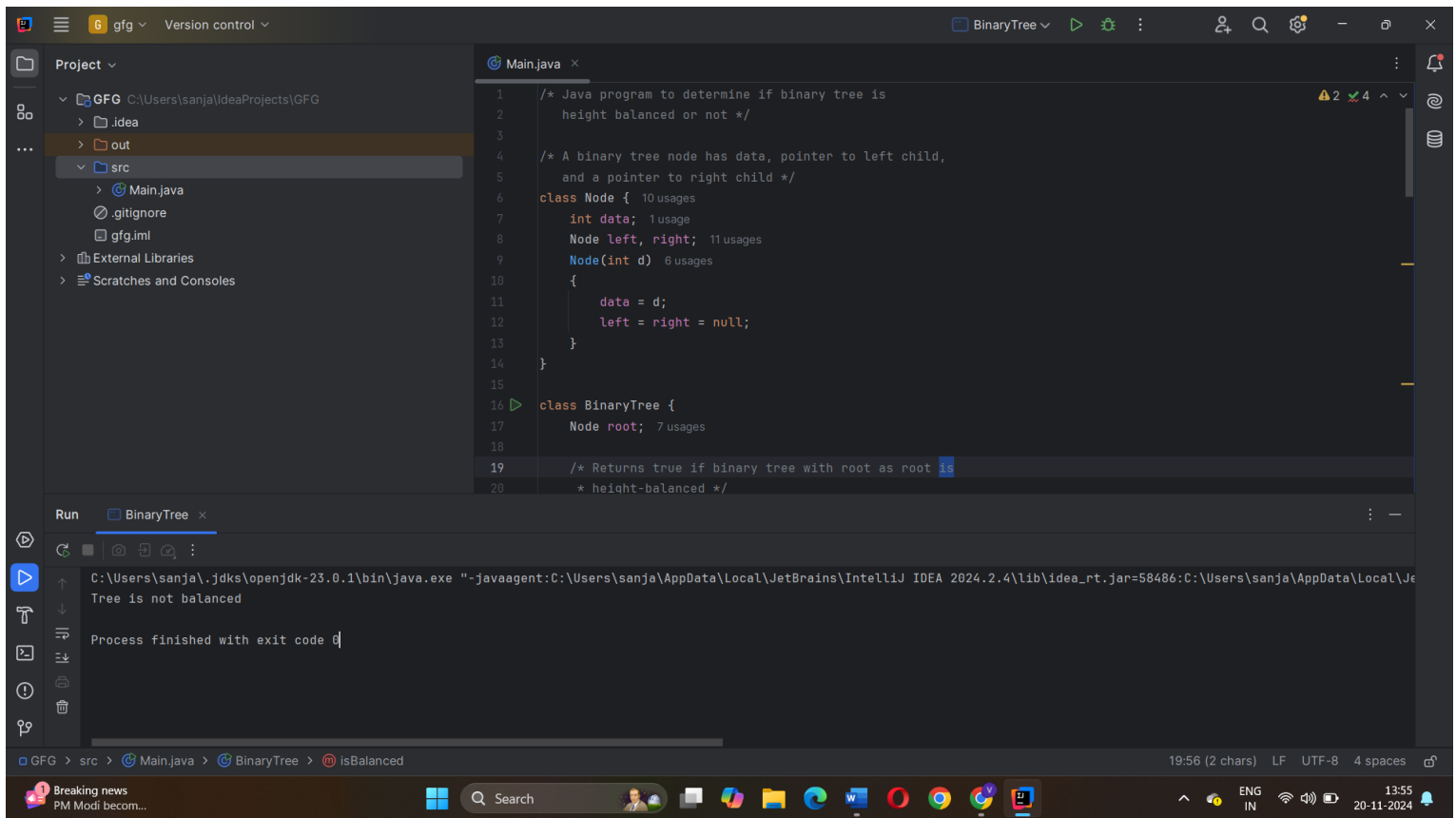
    if (node == null)
        return 0;

    return 1
        + Math.max(height(node.left),
            height(node.right));
}

public static void main(String args[])
{
    BinaryTree tree = new BinaryTree();
    tree.root = new Node(1);
    tree.root.left = new Node(2);
    tree.root.right = new Node(3);
    tree.root.left.left = new Node(4);
    tree.root.left.right = new Node(5);
    tree.root.left.left.left = new Node(8);

    if (tree.isBalanced(tree.root))
        System.out.println("Tree is balanced");
    else
        System.out.println("Tree is not balanced");
}
}
```

Output:



```
1  /* Java program to determine if binary tree is
2     height balanced or not */
3
4  /* A binary tree node has data, pointer to left child,
5     and a pointer to right child */
6  class Node {
7      int data;
8      Node left, right;
9      Node(int d) {
10         data = d;
11         left = right = null;
12     }
13 }
14
15
16 class BinaryTree {
17     Node root;
18
19     /* Returns true if binary tree with root as root is
20      * height-balanced */
```

Run BinaryTree

```
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=58486:C:\Users\sanja\AppData\Local\Je
Tree is not balanced
Process finished with exit code 0
```

6.Triplet Sum In Array:

Time Complexity : $O(n^2)$

Solution:

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

```
public class Main {
```

```
    static boolean find3Numbers(int[] arr, int sum)
```

```
{
```

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

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

```
    for (int i = 0; i < n - 2; i++) {
```

```
        int l = i + 1;
```

```
        int r = n - 1;
```

```

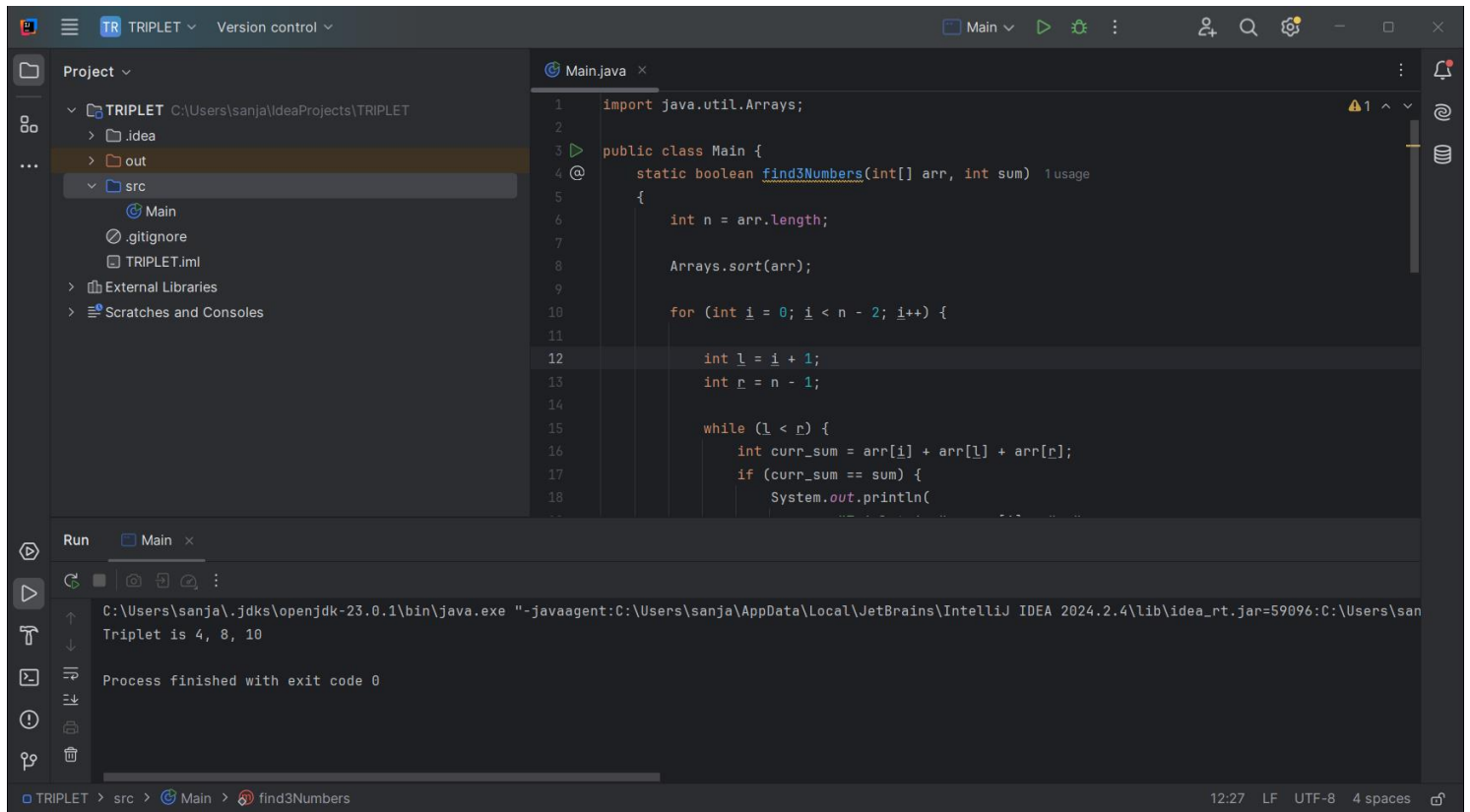
while (l < r) {
    int curr_sum = arr[i] + arr[l] + arr[r];
    if (curr_sum == sum) {
        System.out.println(
            "Triplet is " + arr[i] + ", "
            + arr[l] + ", " + arr[r]);
        return true;
    }
    else if (curr_sum < sum) {
        l++;
    }
    else {
        r--;
    }
}

return false;
}

public static void main(String[] args)
{
    int[] arr = { 1, 4, 45, 6, 10, 8 };
    int sum = 22;
    find3Numbers(arr, sum);
}
}

```

Output:



The screenshot displays the IntelliJ IDEA IDE interface. The left sidebar shows the project structure for 'TRIPLET', with the 'src' directory selected. The main editor window shows the 'Main.java' file with the following code:

```
1 import java.util.Arrays;
2
3 public class Main {
4     static boolean find3Numbers(int[] arr, int sum) 1usage
5     {
6         int n = arr.length;
7
8         Arrays.sort(arr);
9
10        for (int i = 0; i < n - 2; i++) {
11
12            int l = i + 1;
13            int r = n - 1;
14
15            while (l < r) {
16                int curr_sum = arr[i] + arr[l] + arr[r];
17                if (curr_sum == sum) {
18                    System.out.println(
```

The bottom panel shows the 'Run' tab 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=59096:C:\Users\sanja\..."`. The output is: `Triplet is 4, 8, 10`. Below the output, it states: `Process finished with exit code 0`. The status bar at the bottom indicates the file path: `TRIPLET > src > Main > find3Numbers`, the time: `12:27`, and the encoding: `LF UTF-8 4 spaces`.