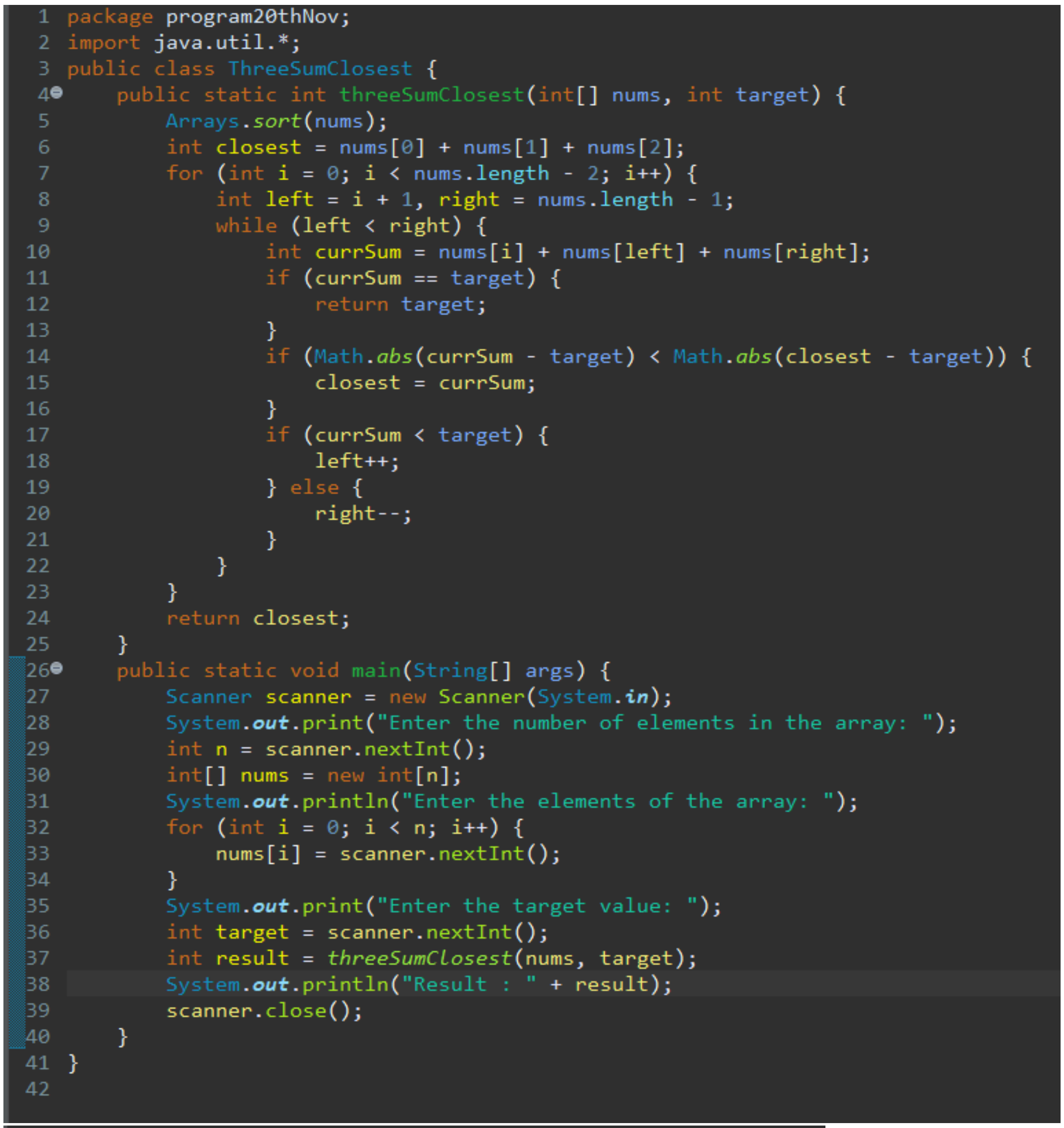
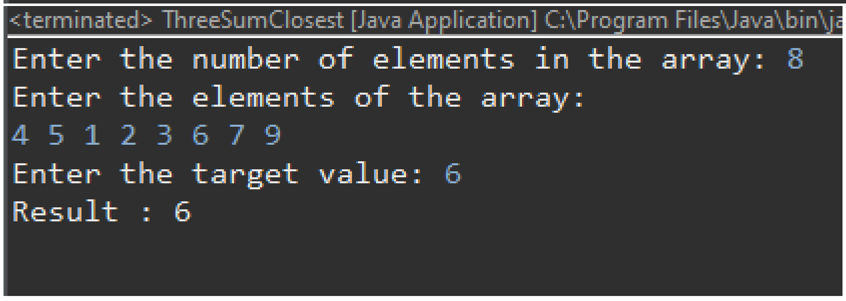
Problem Solving – 20/11/2024

1. Three Sum Closest
   Time complexity: $O(n^2)$
   Space complexity: O(log n)



```java
package program20thNov;
import java.util.*;
public class ThreeSumClosest {
    public static int threeSumClosest(int[] nums, int target) {
        Arrays.sort(nums);
        int closest = nums[0] + nums[1] + nums[2];
        for (int i = 0; i < nums.length - 2; i++) {
            int left = i + 1, right = nums.length - 1;
            while (left < right) {
                int currSum = nums[i] + nums[left] + nums[right];
                if (currSum == target) {
                    return target;
                }
                if (Math.abs(currSum - target) < Math.abs(closest - target)) {
                    closest = currSum;
                }
                if (currSum < target) {
                    left++;
                } else {
                    right--;
                }
            }
        }
        return closest;
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of elements in the array: ");
        int n = scanner.nextInt();
        int[] nums = new int[n];
        System.out.println("Enter the elements of the array: ");
        for (int i = 0; i < n; i++) {
            nums[i] = scanner.nextInt();
        }
        System.out.print("Enter the target value: ");
        int target = scanner.nextInt();
        int result = threeSumClosest(nums, target);
        System.out.println("Result : " + result);
        scanner.close();
    }
}
```

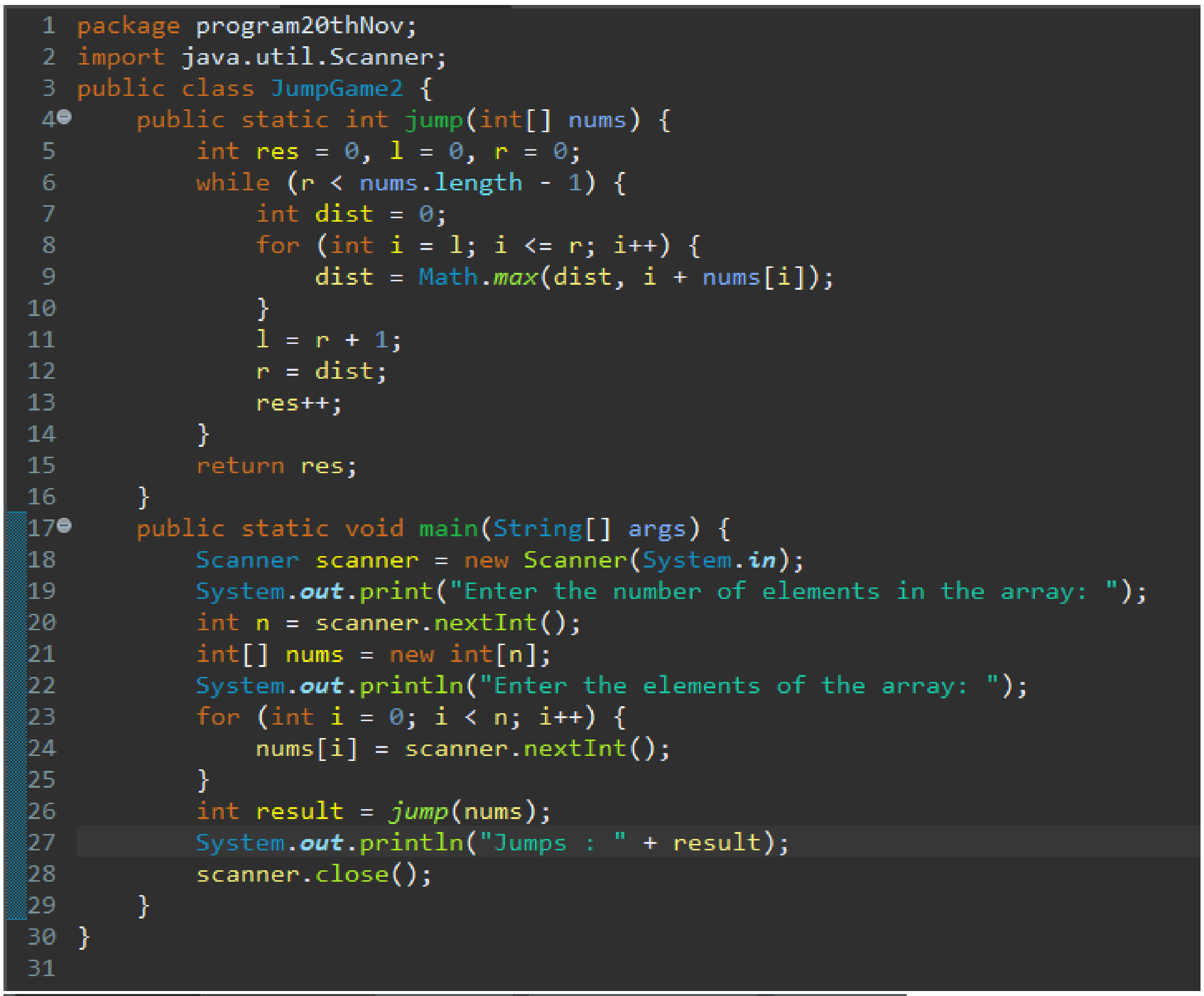


```
<terminated> ThreeSumClosest [Java Application] C:\Program Files\Java\bin\ja
Enter the number of elements in the array: 8
Enter the elements of the array:
4 5 1 2 3 6 7 9
Enter the target value: 6
Result : 6
```

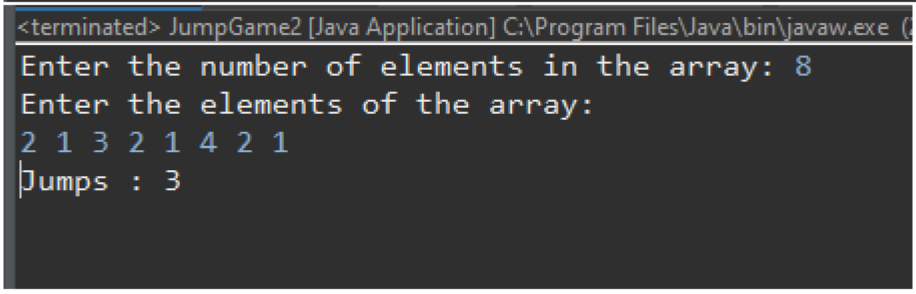
2. Jump Game 2

   Time complexity: O(n²)

   Space complexity: O(1)

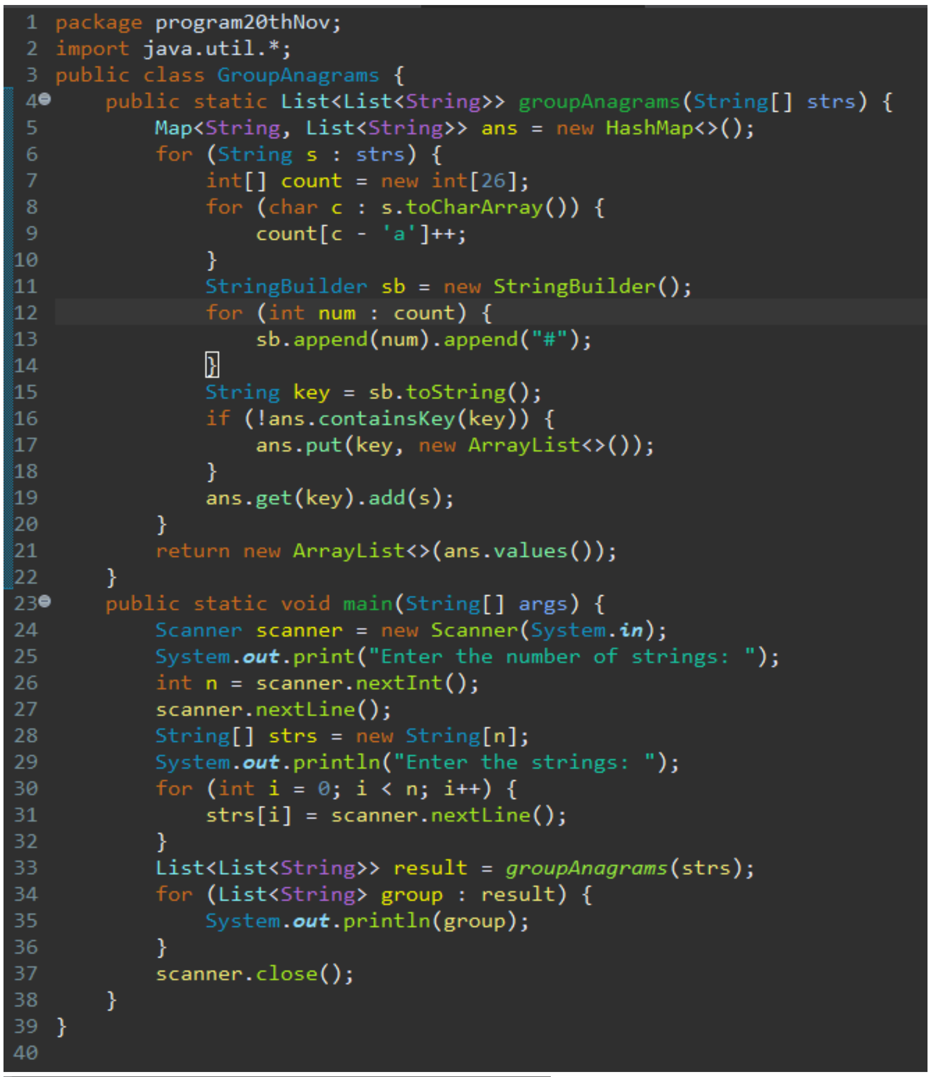


```java
package program20thNov;
import java.util.Scanner;
public class JumpGame2 {
    public static int jump(int[] nums) {
        int res = 0, l = 0, r = 0;
        while (r < nums.length - 1) {
            int dist = 0;
            for (int i = l; i <= r; i++) {
                dist = Math.max(dist, i + nums[i]);
            }
            l = r + 1;
            r = dist;
            res++;
        }
        return res;
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of elements in the array: ");
        int n = scanner.nextInt();
        int[] nums = new int[n];
        System.out.println("Enter the elements of the array: ");
        for (int i = 0; i < n; i++) {
            nums[i] = scanner.nextInt();
        }
        int result = jump(nums);
        System.out.println("Jumps : " + result);
        scanner.close();
    }
}
```

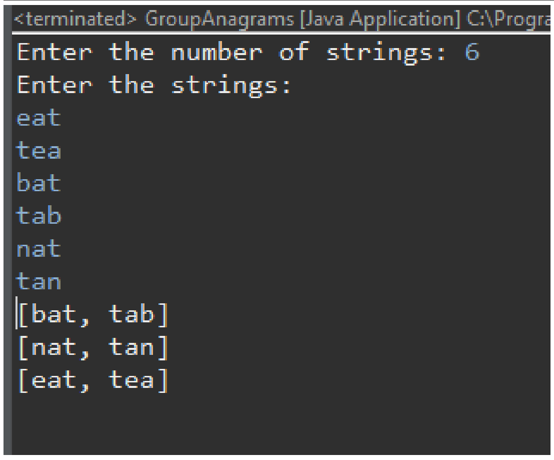


```
<terminated> JumpGame2 [Java Application] C:\Program Files\Java\bin\javaw.exe (
Enter the number of elements in the array: 8
Enter the elements of the array:
2 1 3 2 1 4 2 1
Jumps : 3
```

3. Group Anagram
   Time complexity: O(n * k)
   Space complexity: O(n * k)

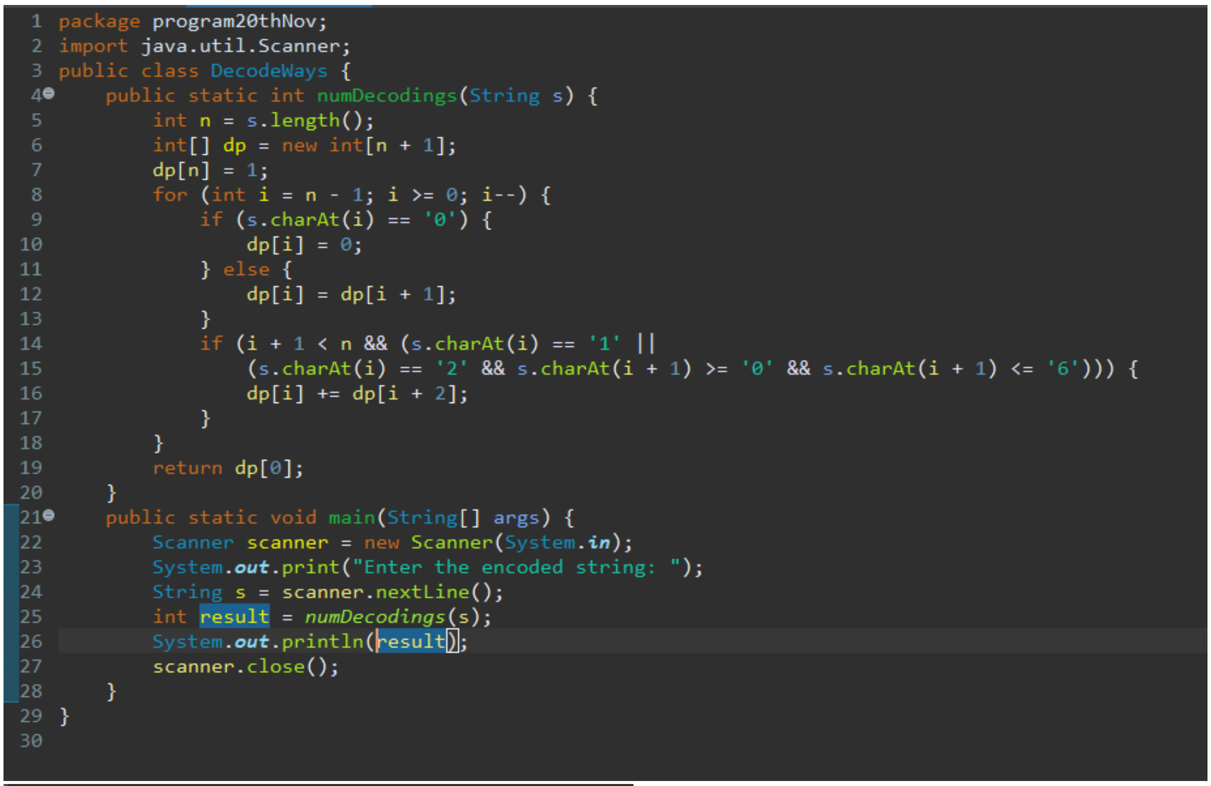


```java
package program20thNov;
import java.util.*;
public class GroupAnagrams {
    public static List<List<String>> groupAnagrams(String[] strs) {
        Map<String, List<String>> ans = new HashMap<>();
        for (String s : strs) {
            int[] count = new int[26];
            for (char c : s.toCharArray()) {
                count[c - 'a']++;
            }
            StringBuilder sb = new StringBuilder();
            for (int num : count) {
                sb.append(num).append("#");
            }
            String key = sb.toString();
            if (!ans.containsKey(key)) {
                ans.put(key, new ArrayList<>());
            }
            ans.get(key).add(s);
        }
        return new ArrayList<>(ans.values());
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of strings: ");
        int n = scanner.nextInt();
        scanner.nextLine();
        String[] strs = new String[n];
        System.out.println("Enter the strings: ");
        for (int i = 0; i < n; i++) {
            strs[i] = scanner.nextLine();
        }
        List<List<String>> result = groupAnagrams(strs);
        for (List<String> group : result) {
            System.out.println(group);
        }
        scanner.close();
    }
}
```

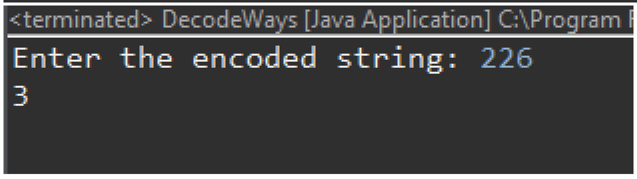


```
<terminated> GroupAnagrams [Java Application] C:\Progra
Enter the number of strings: 6
Enter the strings:
eat
tea
bat
tab
nat
tan
[bat, tab]
[nat, tan]
[eat, tea]
```

4. Decode Ways
   Time complexity: O(n)
   Space complexity: O(n)

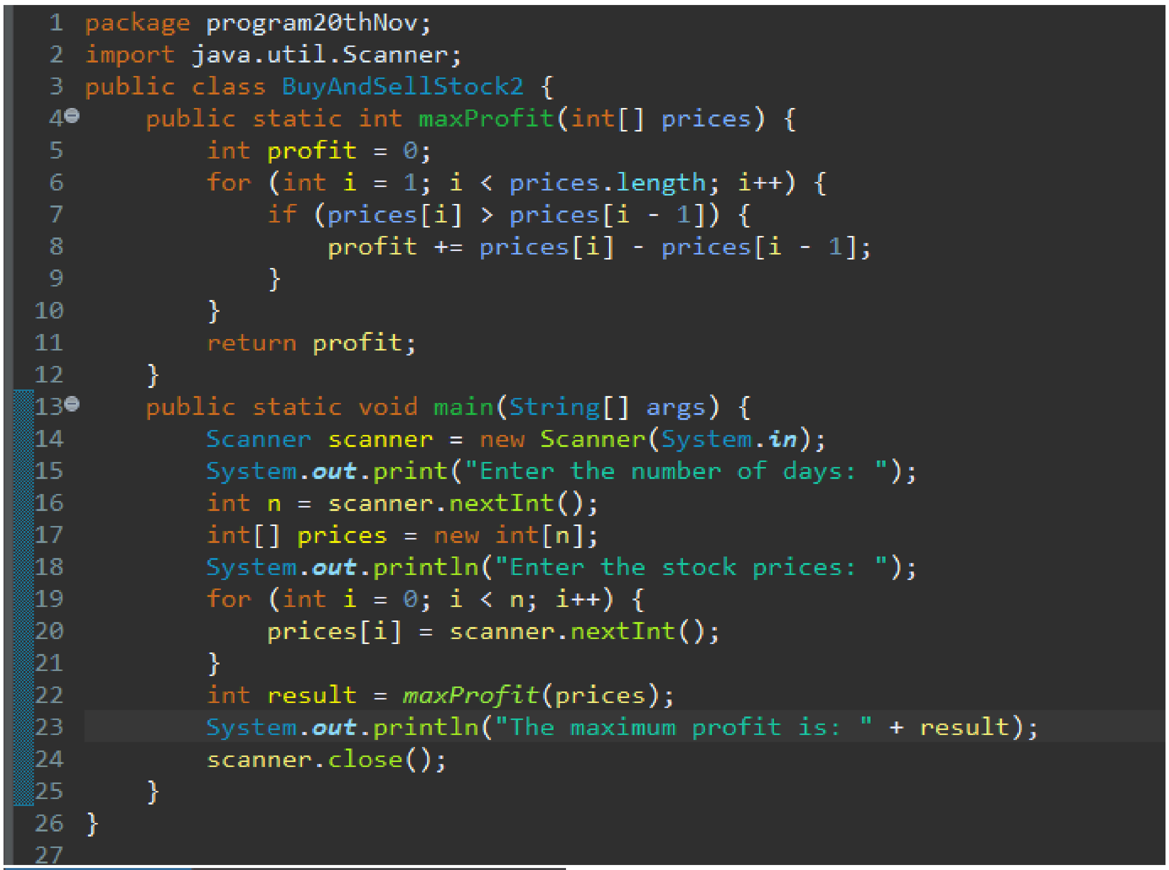


```java
package program20thNov;
import java.util.Scanner;
public class DecodeWays {
    public static int numDecodings(String s) {
        int n = s.length();
        int[] dp = new int[n + 1];
        dp[n] = 1;
        for (int i = n - 1; i >= 0; i--) {
            if (s.charAt(i) == '0') {
                dp[i] = 0;
            } else {
                dp[i] = dp[i + 1];
            }
            if (i + 1 < n && (s.charAt(i) == '1' ||
                (s.charAt(i) == '2' && s.charAt(i + 1) >= '0' && s.charAt(i + 1) <= '6'))) {
                dp[i] += dp[i + 2];
            }
        }
        return dp[0];
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the encoded string: ");
        String s = scanner.nextLine();
        int result = numDecodings(s);
        System.out.println(result);
        scanner.close();
    }
}
```



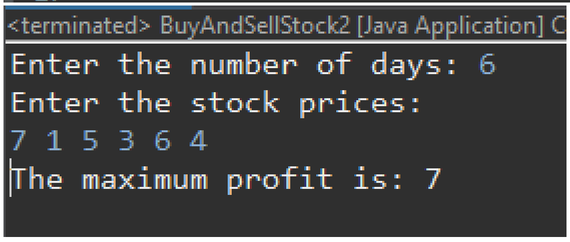
```
<terminated> DecodeWays [Java Application] C:\Program F
Enter the encoded string: 226
3
```

5. Buy and Sell Stock 2
   Time complexity: O(n)
   Space complexity: O(1)



```java
package program20thNov;
import java.util.Scanner;
public class BuyAndSellStock2 {
    public static int maxProfit(int[] prices) {
        int profit = 0;
        for (int i = 1; i < prices.length; i++) {
            if (prices[i] > prices[i - 1]) {
                profit += prices[i] - prices[i - 1];
            }
        }
        return profit;
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of days: ");
        int n = scanner.nextInt();
        int[] prices = new int[n];
        System.out.println("Enter the stock prices: ");
        for (int i = 0; i < n; i++) {
            prices[i] = scanner.nextInt();
        }
        int result = maxProfit(prices);
        System.out.println("The maximum profit is: " + result);
        scanner.close();
    }
}

```

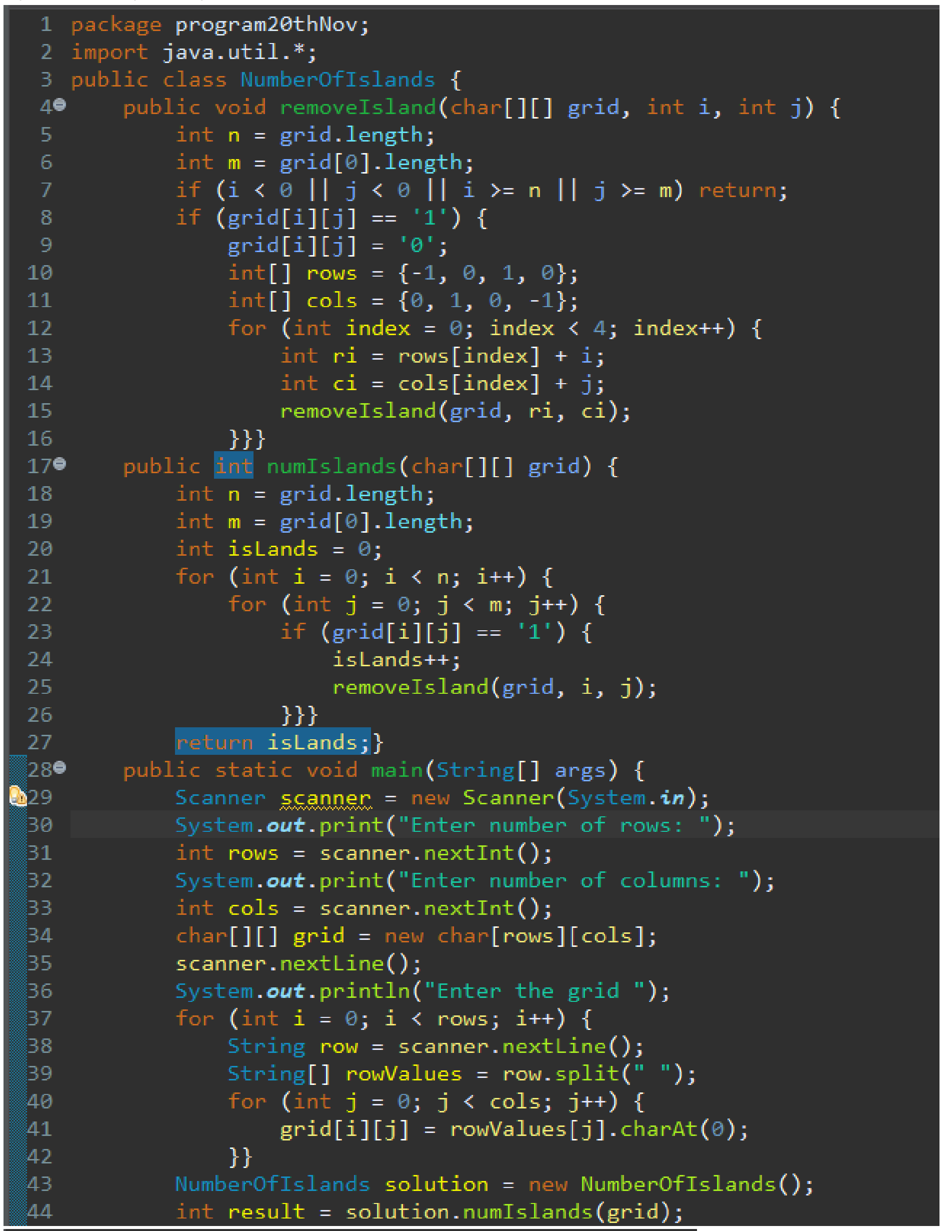


```
<terminated> BuyAndSellStock2 [Java Application] C
Enter the number of days: 6
Enter the stock prices:
7 1 5 3 6 4
The maximum profit is: 7
```

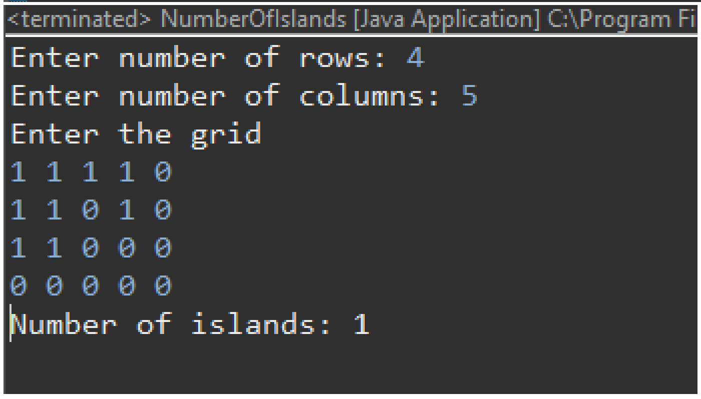
6. Number of Islands

Time complexity: O(n * m)

Space complexity: O(n * m)

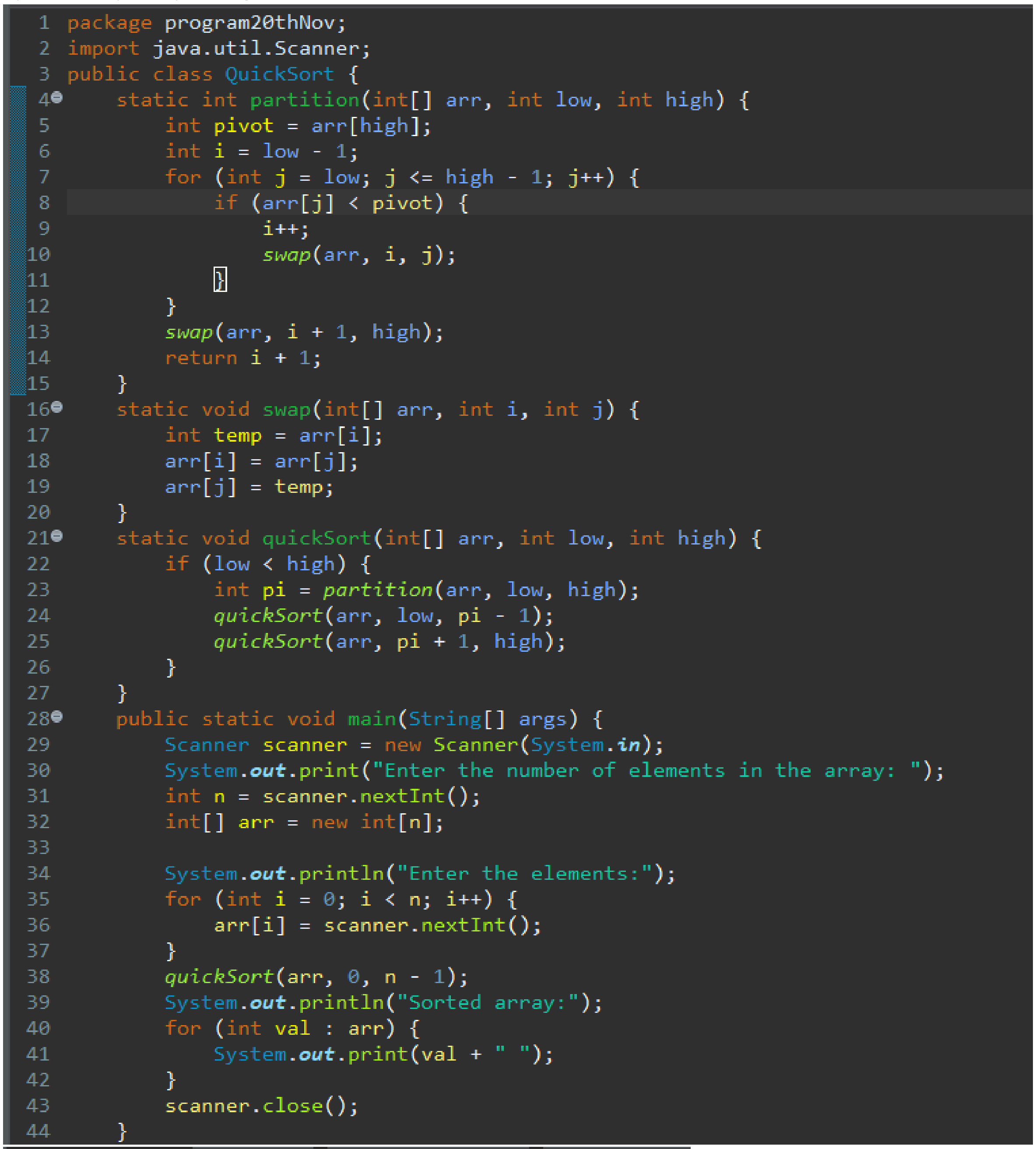


```java
package program20thNov;
import java.util.*;
public class NumberOfIslands {
    public void removeIsland(char[][] grid, int i, int j) {
        int n = grid.length;
        int m = grid[0].length;
        if (i < 0 || j < 0 || i >= n || j >= m) return;
        if (grid[i][j] == '1') {
            grid[i][j] = '0';
            int[] rows = {-1, 0, 1, 0};
            int[] cols = {0, 1, 0, -1};
            for (int index = 0; index < 4; index++) {
                int ri = rows[index] + i;
                int ci = cols[index] + j;
                removeIsland(grid, ri, ci);
            }}}
    public int numIslands(char[][] grid) {
        int n = grid.length;
        int m = grid[0].length;
        int isLands = 0;
        for (int i = 0; i < n; i++) {
            for (int j = 0; j < m; j++) {
                if (grid[i][j] == '1') {
                    isLands++;
                    removeIsland(grid, i, j);
                }}}
        return isLands;}
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter number of rows: ");
        int rows = scanner.nextInt();
        System.out.print("Enter number of columns: ");
        int cols = scanner.nextInt();
        char[][] grid = new char[rows][cols];
        scanner.nextLine();
        System.out.println("Enter the grid ");
        for (int i = 0; i < rows; i++) {
            String row = scanner.nextLine();
            String[] rowValues = row.split(" ");
            for (int j = 0; j < cols; j++) {
                grid[i][j] = rowValues[j].charAt(0);
            }}
        NumberOfIslands solution = new NumberOfIslands();
        int result = solution.numIslands(grid);
```



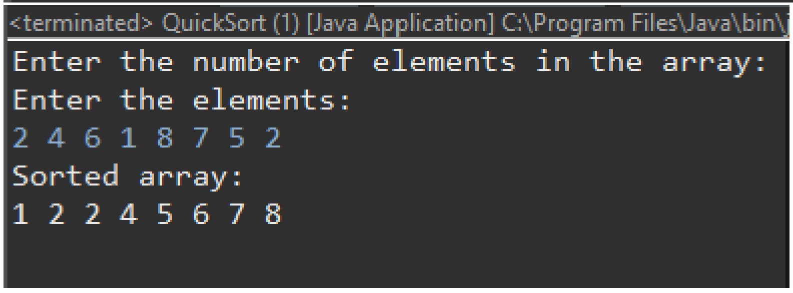
```
<terminated> NumberOfIslands [Java Application] C:\Program Fi
Enter number of rows: 4
Enter number of columns: 5
Enter the grid
1 1 1 1 0
1 1 0 1 0
1 1 0 0 0
0 0 0 0 0
Number of islands: 1
```

7. Quick Sort
Time complexity: O(n * log n)
Space complexity: O(log n)



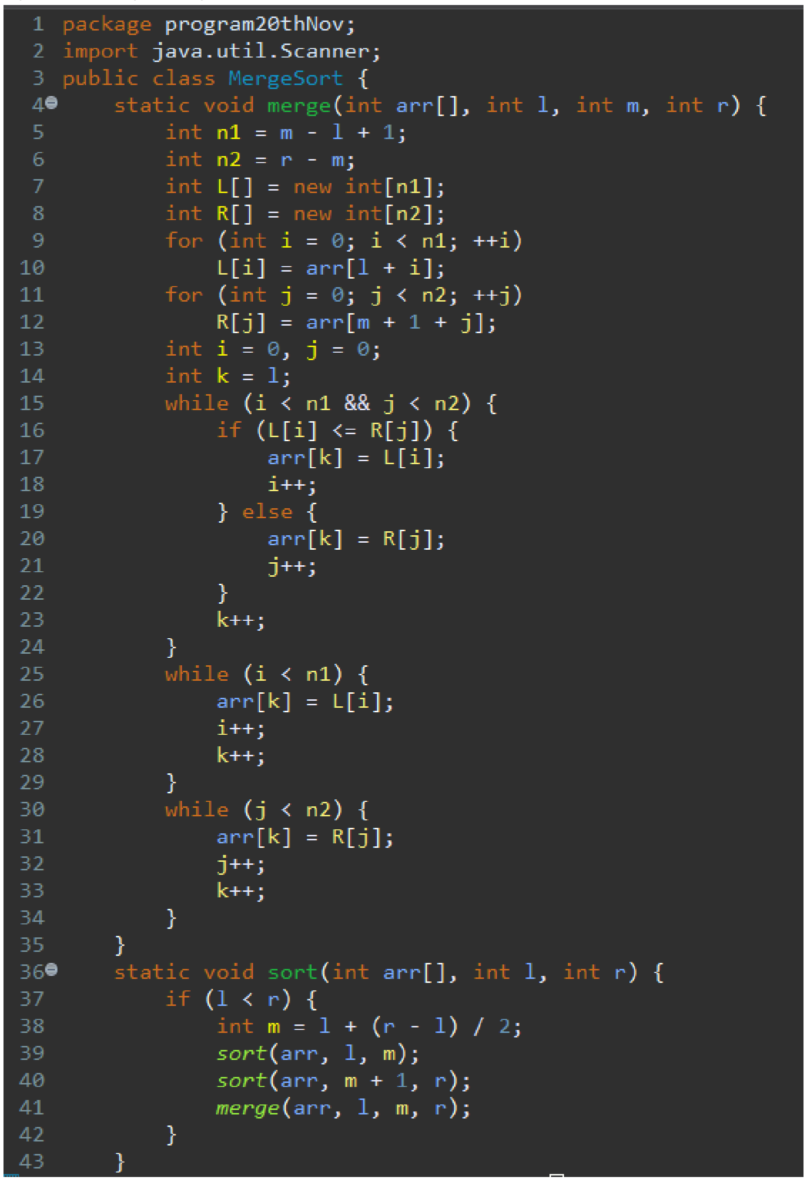
```java
package program20thNov;
import java.util.Scanner;
public class QuickSort {
    static int partition(int[] arr, int low, int high) {
        int pivot = arr[high];
        int i = low - 1;
        for (int j = low; j <= high - 1; j++) {
            if (arr[j] < pivot) {
                i++;
                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) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of elements in the array: ");
        int n = scanner.nextInt();
        int[] arr = new int[n];

        System.out.println("Enter the elements:");
        for (int i = 0; i < n; i++) {
            arr[i] = scanner.nextInt();
        }
        quickSort(arr, 0, n - 1);
        System.out.println("Sorted array:");
        for (int val : arr) {
            System.out.print(val + " ");
        }
        scanner.close();
    }
```

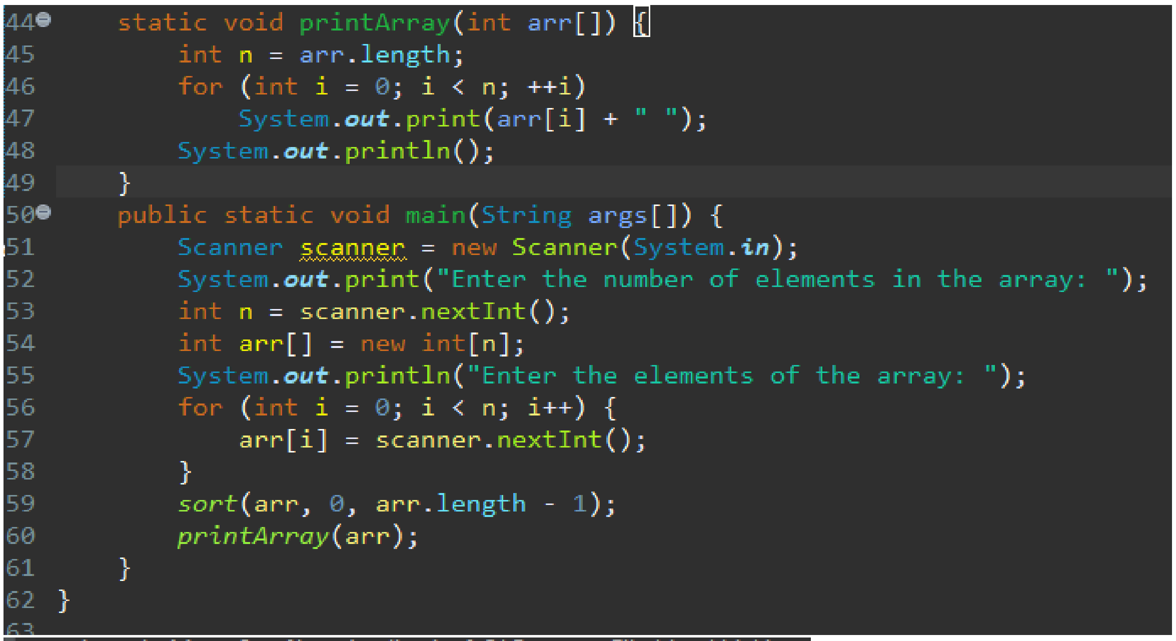


```
<terminated> QuickSort (1) [Java Application] C:\Program Files\Java\bin\j
Enter the number of elements in the array:
Enter the elements:
2 4 6 1 8 7 5 2
Sorted array:
1 2 2 4 5 6 7 8
```

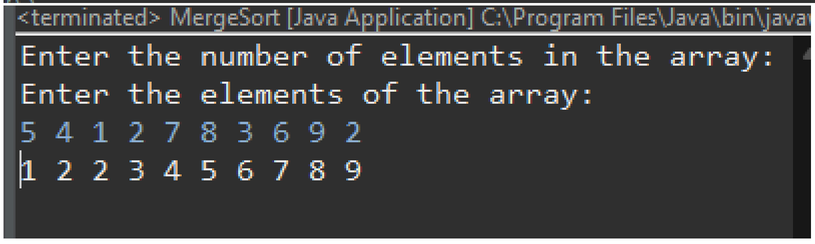
8. Merge Sort
Time complexity: O(n log n)
Space complexity: O(n)



```
package program20thNov;
import java.util.Scanner;
public class MergeSort {
    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);
        }
    }
```



```java
    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[]) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of elements in the array: ");
        int n = scanner.nextInt();
        int arr[] = new int[n];
        System.out.println("Enter the elements of the array: ");
        for (int i = 0; i < n; i++) {
            arr[i] = scanner.nextInt();
        }
        sort(arr, 0, arr.length - 1);
        printArray(arr);
    }
}
```

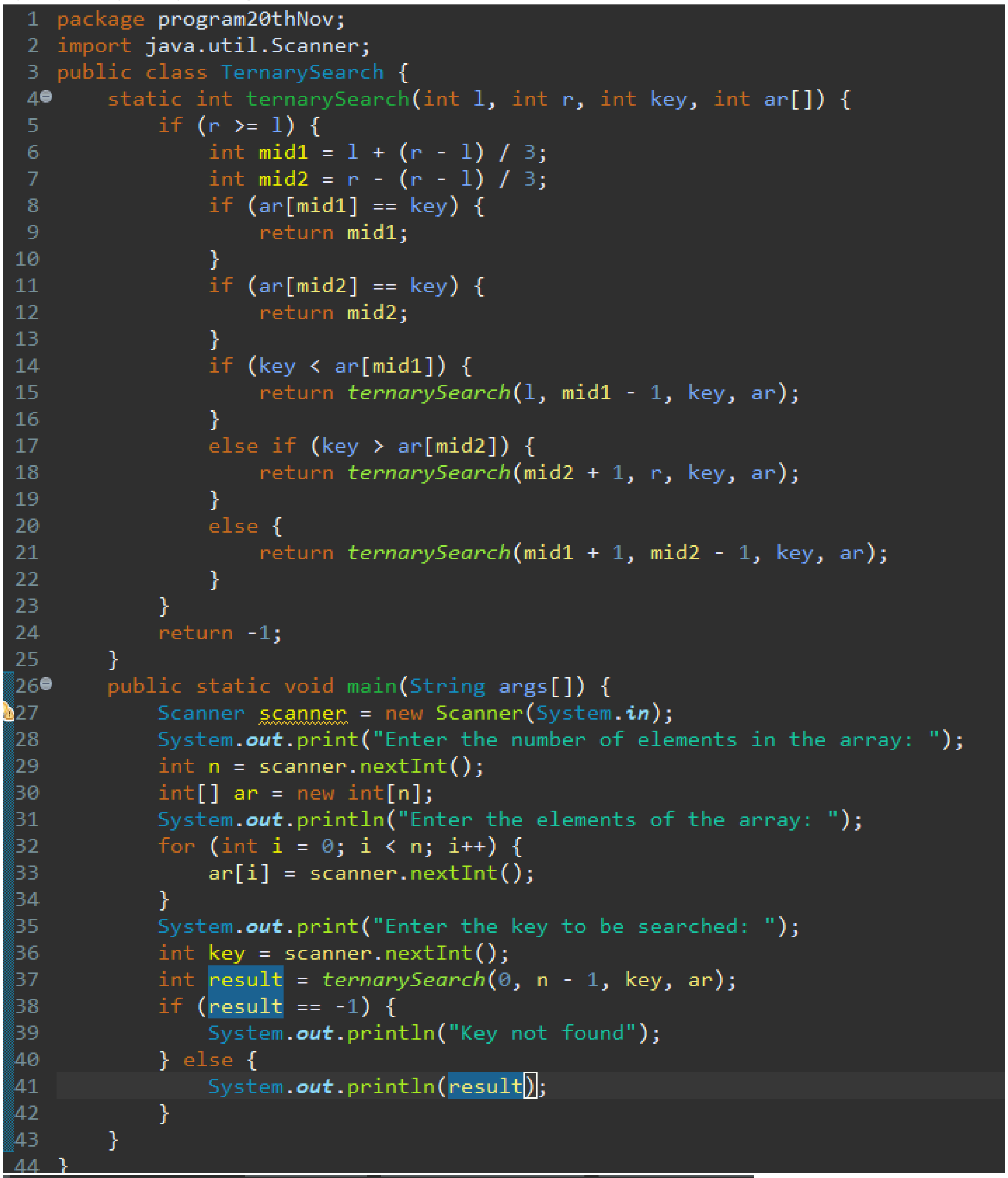


```
<terminated> MergeSort [Java Application] C:\Program Files\Java\bin\java
Enter the number of elements in the array:
Enter the elements of the array:
5 4 1 2 7 8 3 6 9 2
1 2 2 3 4 5 6 7 8 9
```

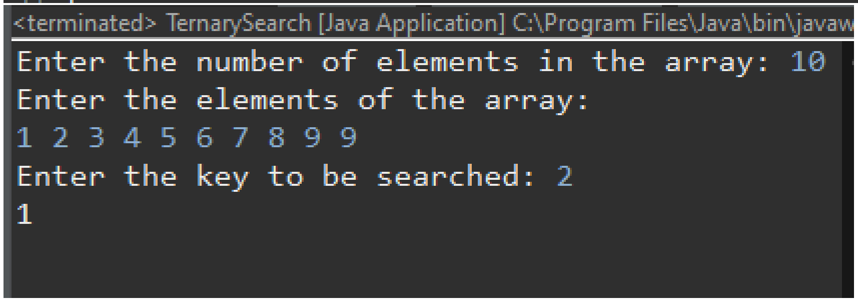
9. Ternary Search

Time complexity: $O(\log_3 n)$

Space complexity: $O(\log n)$



```java
package program20thNov;
import java.util.Scanner;
public class TernarySearch {
    static int ternarySearch(int l, int r, int key, int ar[]) {
        if (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]) {
                return ternarySearch(l, mid1 - 1, key, ar);
            }
            else if (key > ar[mid2]) {
                return ternarySearch(mid2 + 1, r, key, ar);
            }
            else {
                return ternarySearch(mid1 + 1, mid2 - 1, key, ar);
            }
        }
        return -1;
    }
    public static void main(String args[]) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of elements in the array: ");
        int n = scanner.nextInt();
        int[] ar = new int[n];
        System.out.println("Enter the elements of the array: ");
        for (int i = 0; i < n; i++) {
            ar[i] = scanner.nextInt();
        }
        System.out.print("Enter the key to be searched: ");
        int key = scanner.nextInt();
        int result = ternarySearch(0, n - 1, key, ar);
        if (result == -1) {
            System.out.println("Key not found");
        } else {
            System.out.println(result);
        }
    }
}
```

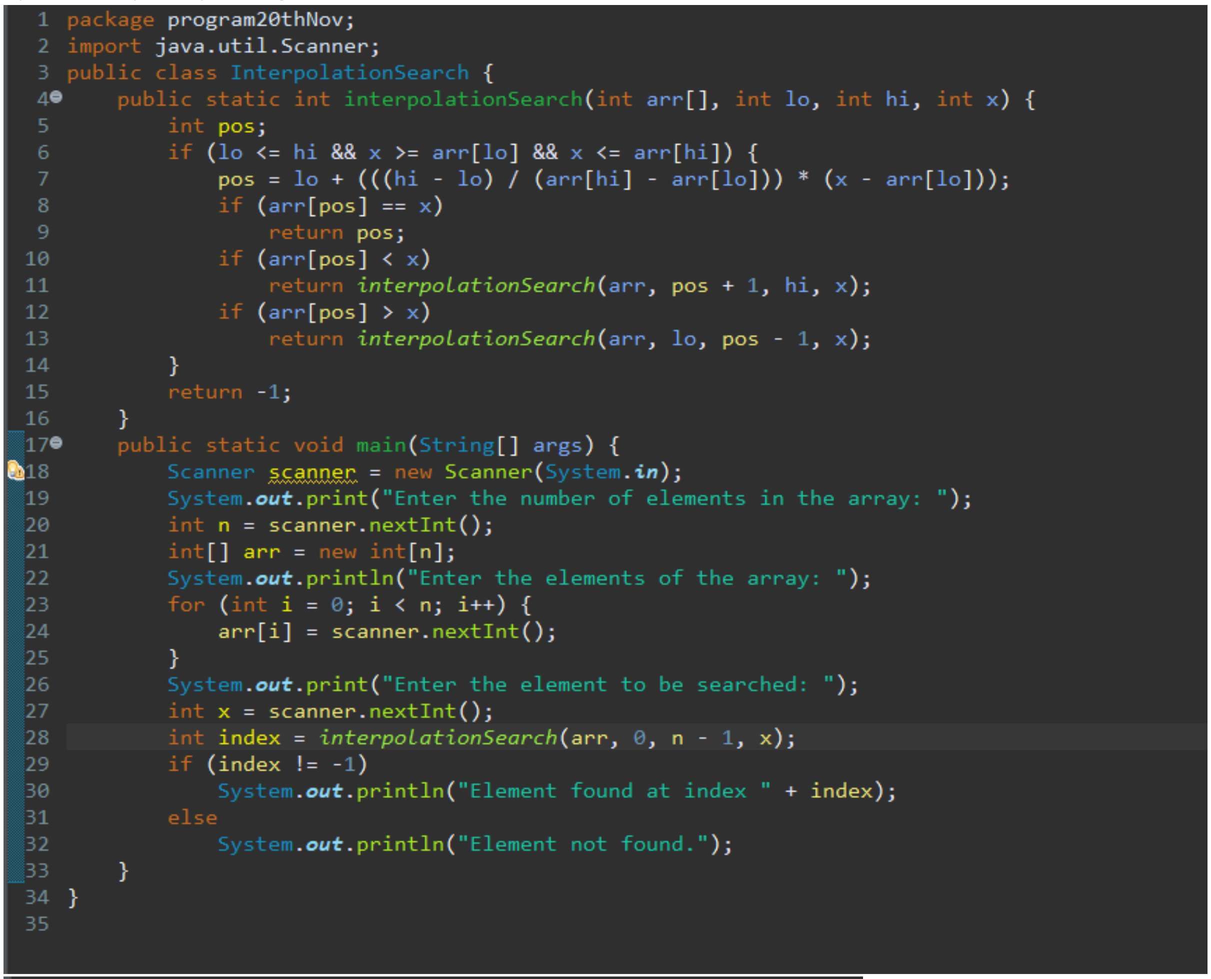


```
<terminated> TernarySearch [Java Application] C:\Program Files\Java\bin\javaw
Enter the number of elements in the array: 10
Enter the elements of the array:
1 2 3 4 5 6 7 8 9 9
Enter the key to be searched: 2
1
```

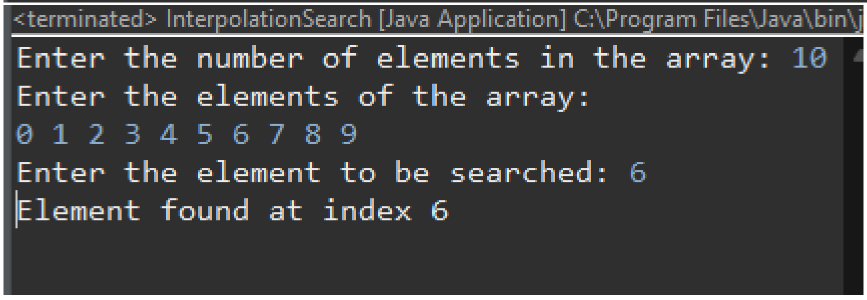
10. Interpolation Search

Time complexity: O(log n)

Space complexity: O(log n)



```java
package program20thNov;
import java.util.Scanner;
public class InterpolationSearch {
    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) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of elements in the array: ");
        int n = scanner.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter the elements of the array: ");
        for (int i = 0; i < n; i++) {
            arr[i] = scanner.nextInt();
        }
        System.out.print("Enter the element to be searched: ");
        int x = scanner.nextInt();
        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.");
    }
}
```



```
<terminated> InterpolationSearch [Java Application] C:\Program Files\Java\bin\j
Enter the number of elements in the array: 10
Enter the elements of the array:
0 1 2 3 4 5 6 7 8 9
Enter the element to be searched: 6
Element found at index 6
```