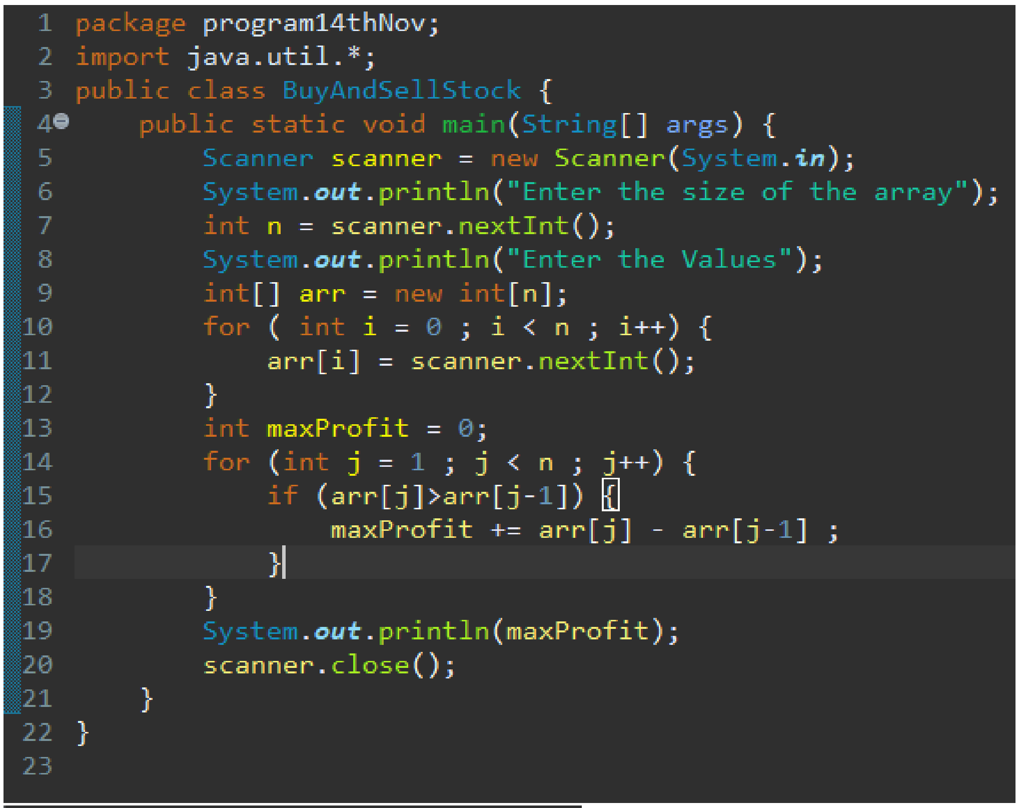
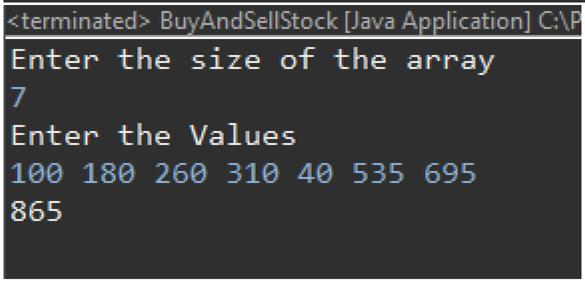
Problem Solving 14/11/2024

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

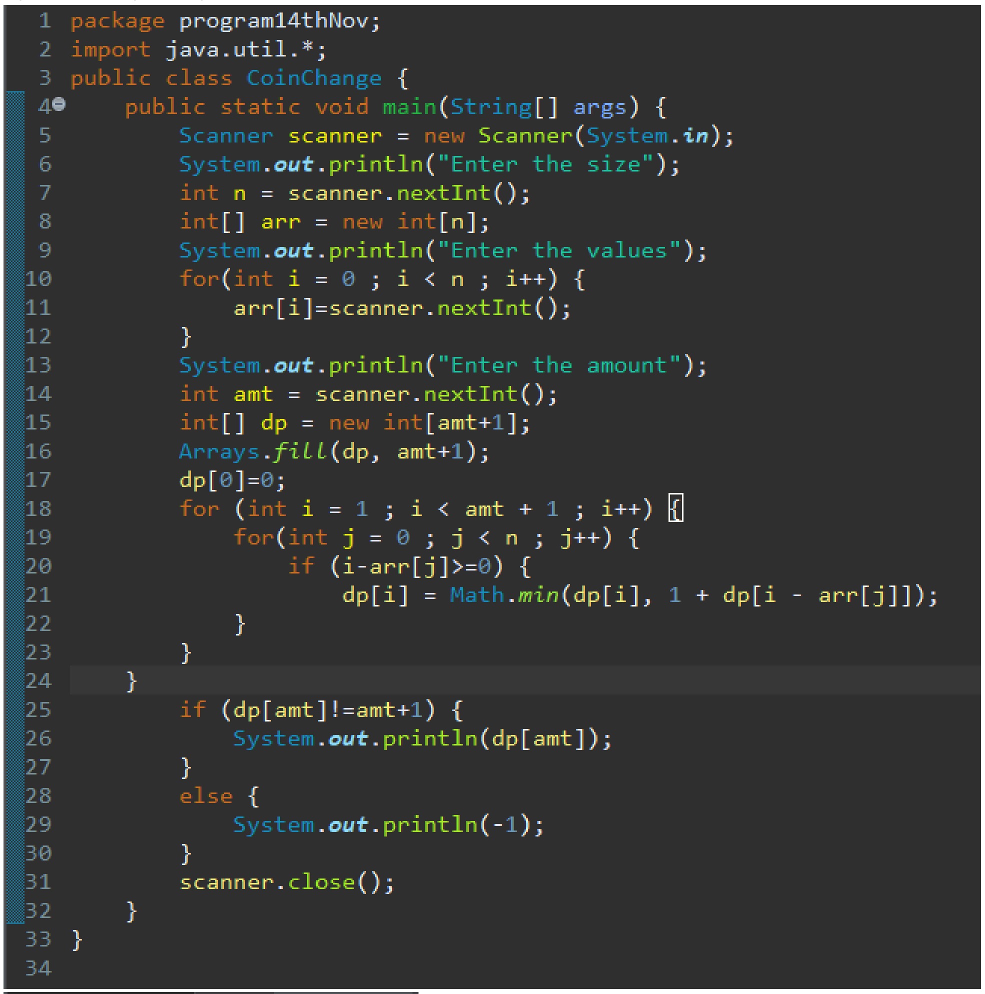


```java
package program14thNov;
import java.util.*;
public class BuyAndSellStock {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter the size of the array");
        int n = scanner.nextInt();
        System.out.println("Enter the Values");
        int[] arr = new int[n];
        for ( int i = 0 ; i < n ; i++) {
            arr[i] = scanner.nextInt();
        }
        int maxProfit = 0;
        for (int j = 1 ; j < n ; j++) {
            if (arr[j]>arr[j-1]) {
                maxProfit += arr[j] - arr[j-1] ;
            }
        }
        System.out.println(maxProfit);
        scanner.close();
    }
}
```

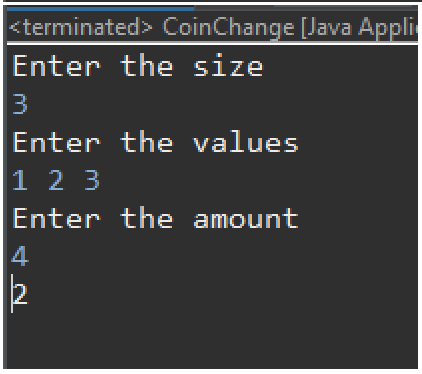


```
<terminated> BuyAndSellStock [Java Application] C:\P
Enter the size of the array
7
Enter the Values
100 180 260 310 40 535 695
865
```

2. Coin Change
   Time complexity: O(n * amt)
   Space complexity: O(amt)

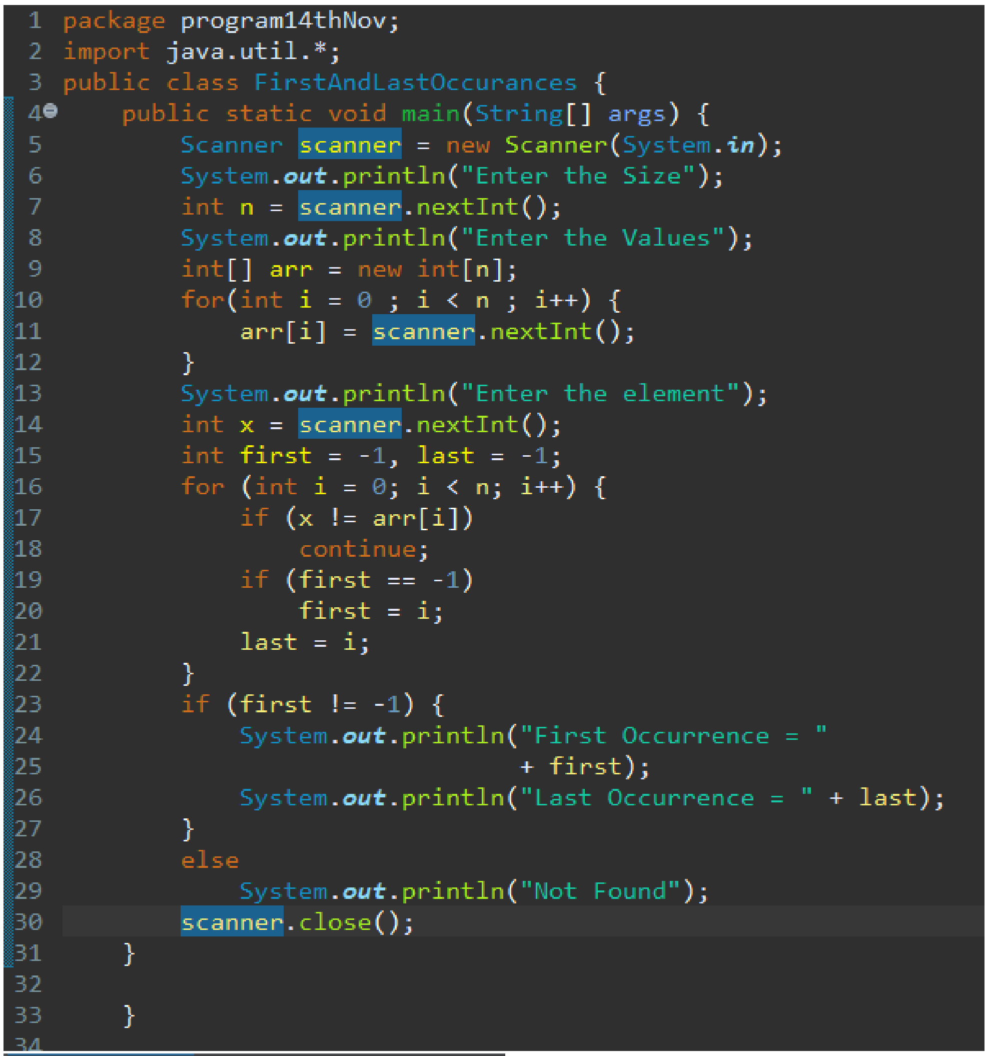


```java
package program14thNov;
import java.util.*;
public class CoinChange {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter the size");
        int n = scanner.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter the values");
        for(int i = 0 ; i < n ; i++) {
            arr[i]=scanner.nextInt();
        }
        System.out.println("Enter the amount");
        int amt = scanner.nextInt();
        int[] dp = new int[amt+1];
        Arrays.fill(dp, amt+1);
        dp[0]=0;
        for (int i = 1 ; i < amt + 1 ; i++) {
            for(int j = 0 ; j < n ; j++) {
                if (i-arr[j]>=0) {
                    dp[i] = Math.min(dp[i], 1 + dp[i - arr[j]]);
                }
            }
    }
        if (dp[amt]!=amt+1) {
            System.out.println(dp[amt]);
        }
        else {
            System.out.println(-1);
        }
        scanner.close();
    }
}
```



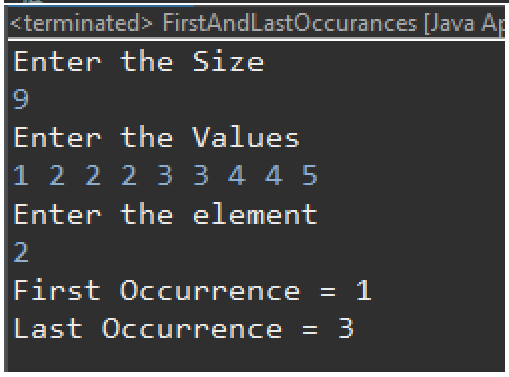
```
<terminated> CoinChange [Java Appli
Enter the size
3
Enter the values
1 2 3
Enter the amount
4
2
```

3. First and Last Occurrences
   Time complexity: O(n)
   Space complexity: O(1)



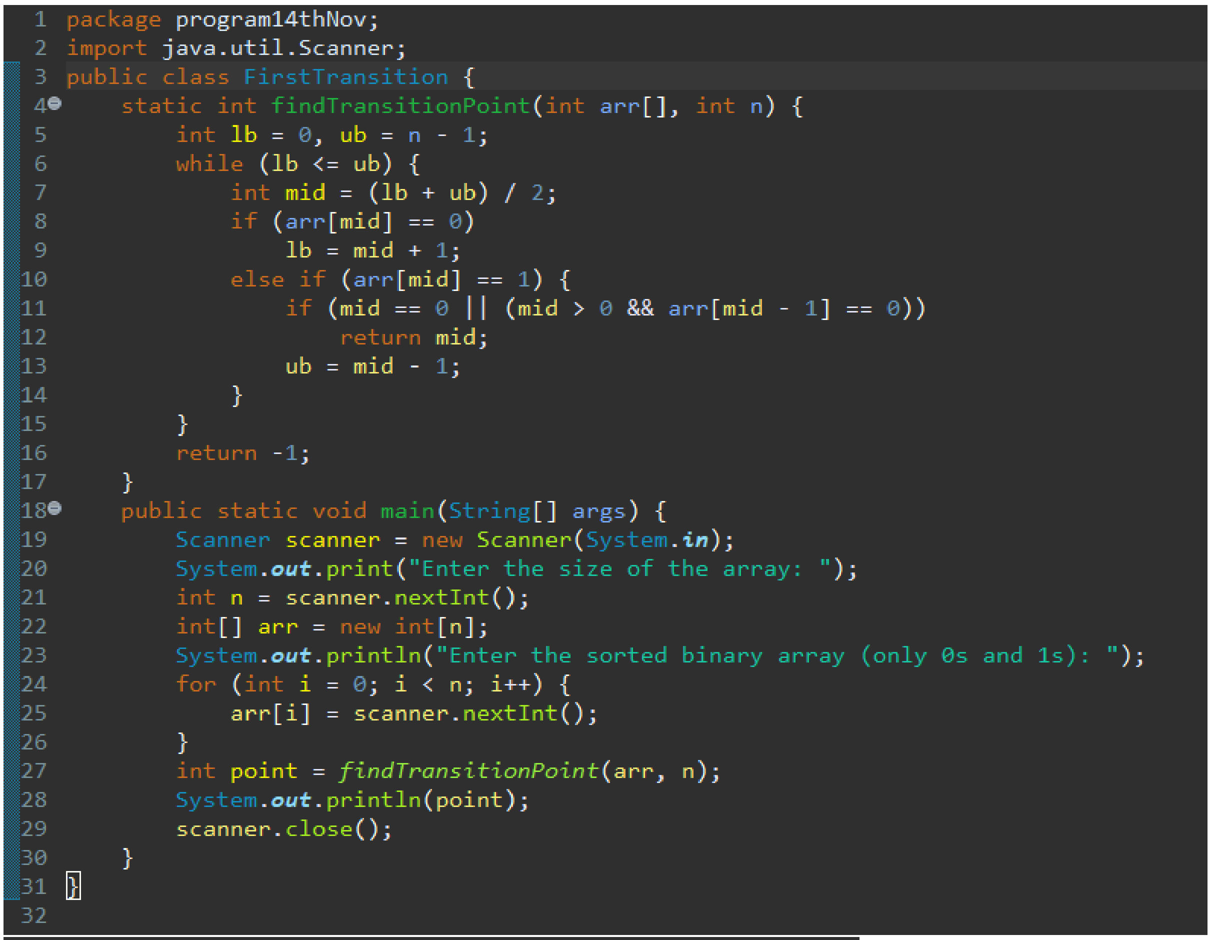
```java
package program14thNov;
import java.util.*;
public class FirstAndLastOccurances {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter the Size");
        int n = scanner.nextInt();
        System.out.println("Enter the Values");
        int[] arr = new int[n];
        for(int i = 0 ; i < n ; i++) {
            arr[i] = scanner.nextInt();
        }
        System.out.println("Enter the element");
        int x = scanner.nextInt();
        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");
        scanner.close();
    }

    }
```

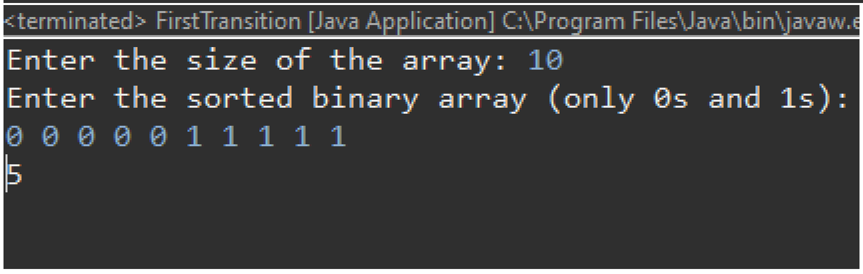


```
<terminated> FirstAndLastOccurances [Java Ap
Enter the Size
9
Enter the Values
1 2 2 2 3 3 4 4 5
Enter the element
2
First Occurrence = 1
Last Occurrence = 3
```

4. First Transitions
   Time complexity: O(log n)
   Space complexity: O(1)

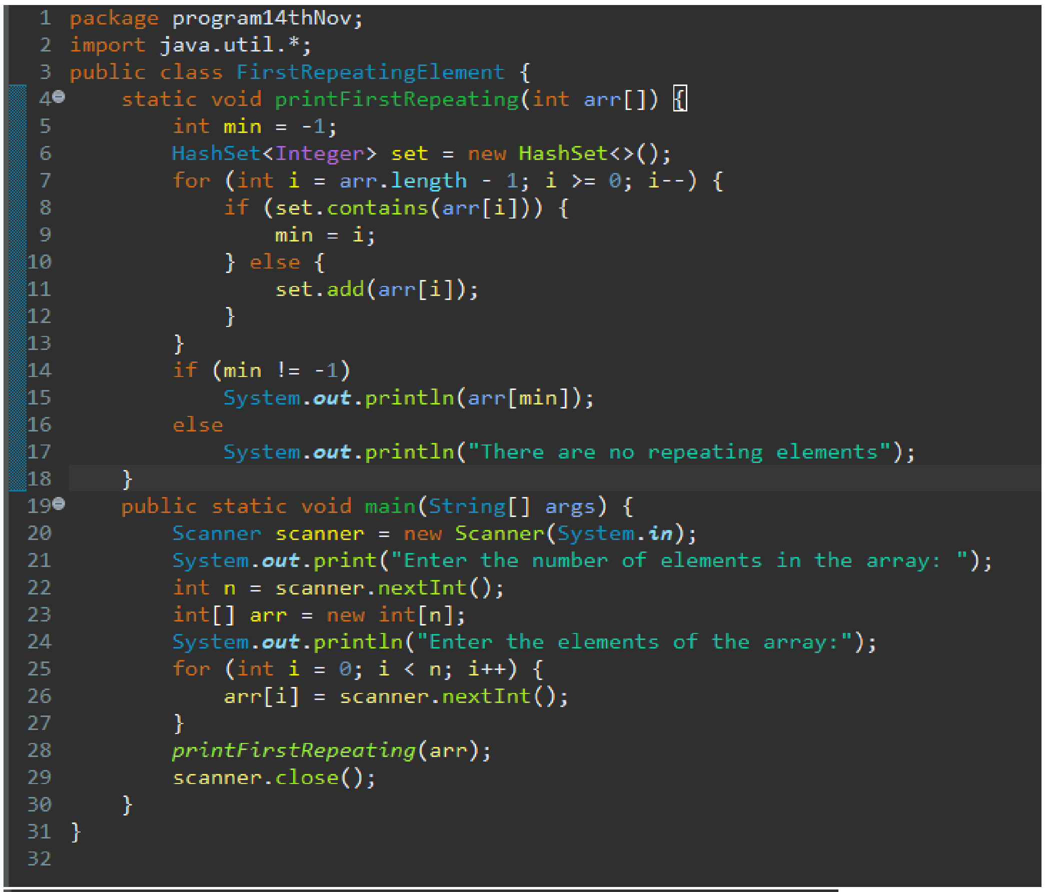


```java
package program14thNov;
import java.util.Scanner;
public class FirstTransition {
    static int findTransitionPoint(int arr[], int n) {
        int lb = 0, ub = n - 1;
        while (lb <= ub) {
            int mid = (lb + ub) / 2;
            if (arr[mid] == 0)
                lb = mid + 1;
            else if (arr[mid] == 1) {
                if (mid == 0 || (mid > 0 && arr[mid - 1] == 0))
                    return mid;
                ub = mid - 1;
            }
        }
        return -1;
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the size of the array: ");
        int n = scanner.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter the sorted binary array (only 0s and 1s): ");
        for (int i = 0; i < n; i++) {
            arr[i] = scanner.nextInt();
        }
        int point = findTransitionPoint(arr, n);
        System.out.println(point);
        scanner.close();
    }
}
```

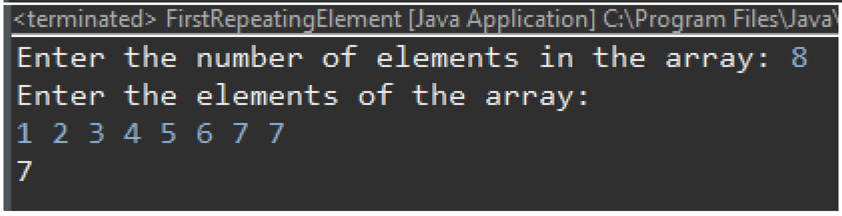


```
<terminated> FirstTransition [Java Application] C:\Program Files\Java\bin\javaw.e
Enter the size of the array: 10
Enter the sorted binary array (only 0s and 1s):
0 0 0 0 0 1 1 1 1 1
5
```

5. First Repeating Element
   Time complexity: O(n)
   Space complexity: O(n)



```java
package program14thNov;
import java.util.*;
public class FirstRepeatingElement {
    static void printFirstRepeating(int arr[]) {
        int min = -1;
        HashSet<Integer> set = new HashSet<>();
        for (int i = arr.length - 1; i >= 0; i--) {
            if (set.contains(arr[i])) {
                min = i;
            } else {
                set.add(arr[i]);
            }
        }
        if (min != -1)
            System.out.println(arr[min]);
        else
            System.out.println("There are no repeating elements");
    }
    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();
        }
        printFirstRepeating(arr);
        scanner.close();
    }
}
```

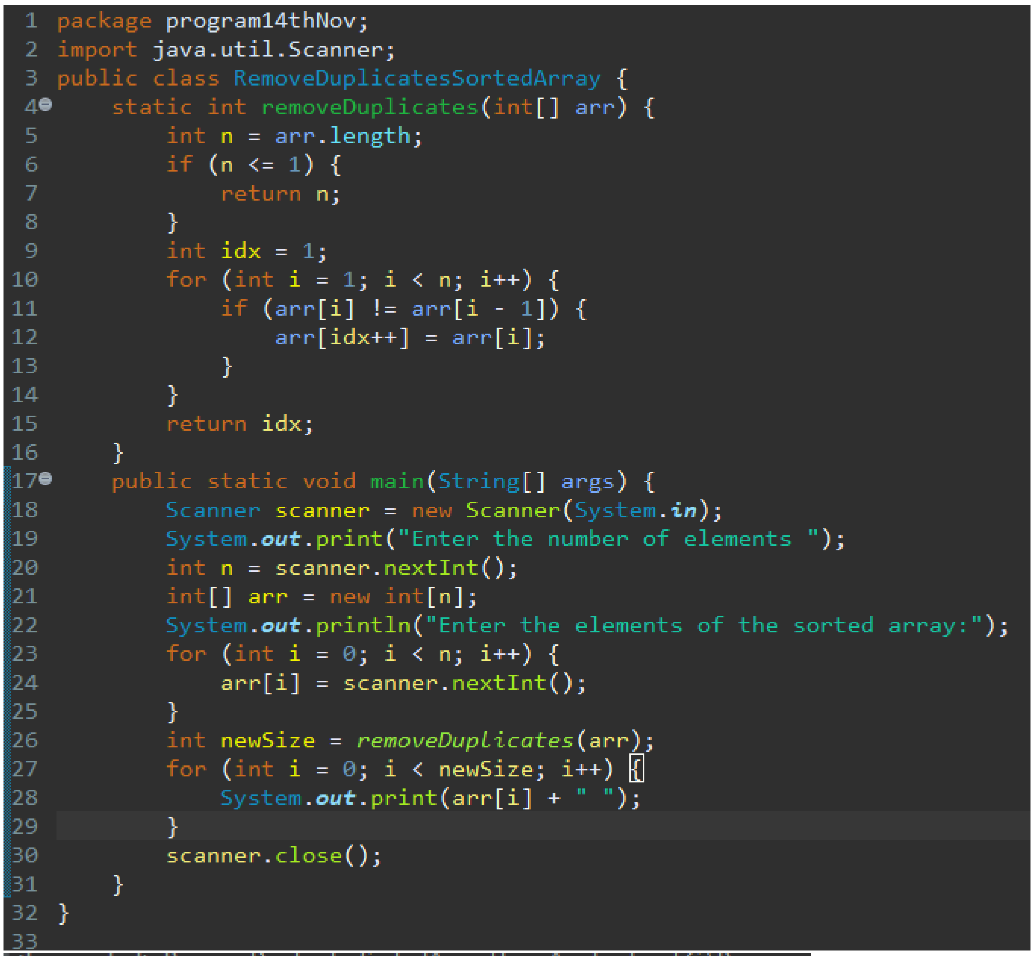


```
<terminated> FirstRepeatingElement [Java Application] C:\Program Files\Java\
Enter the number of elements in the array: 8
Enter the elements of the array:
1 2 3 4 5 6 7 7
7
```

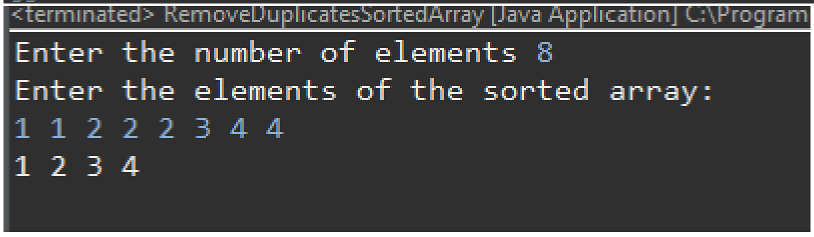
6. Remove Duplicates Sorted Array

   Time complexity: O(n)

   Space complexity: O(1)

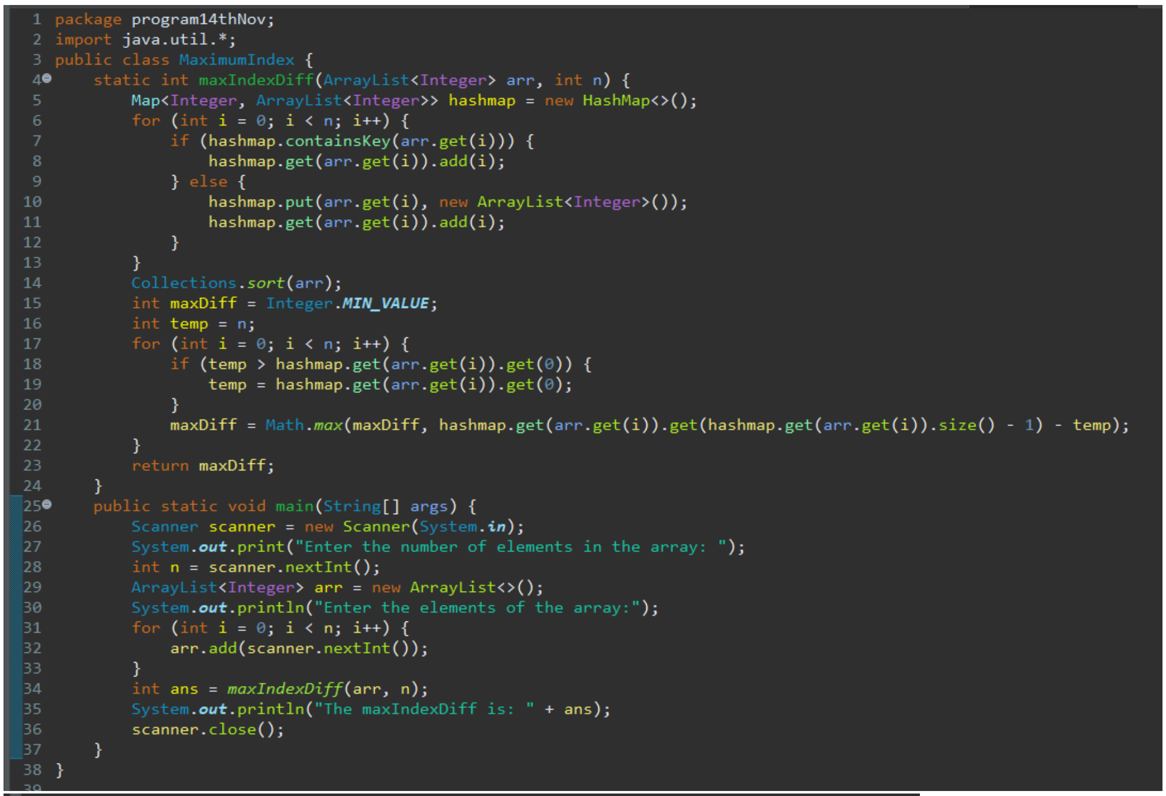


```java
package program14thNov;
import java.util.Scanner;
public class RemoveDuplicatesSortedArray {
    static int removeDuplicates(int[] arr) {
        int n = arr.length;
        if (n <= 1) {
            return n;
        }
        int idx = 1;
        for (int i = 1; i < n; i++) {
            if (arr[i] != arr[i - 1]) {
                arr[idx++] = arr[i];
            }
        }
        return idx;
    }
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of elements ");
        int n = scanner.nextInt();
        int[] arr = new int[n];
        System.out.println("Enter the elements of the sorted array:");
        for (int i = 0; i < n; i++) {
            arr[i] = scanner.nextInt();
        }
        int newSize = removeDuplicates(arr);
        for (int i = 0; i < newSize; i++) {
            System.out.print(arr[i] + " ");
        }
        scanner.close();
    }
}
```

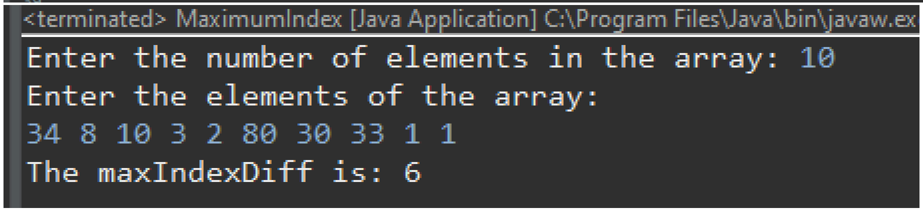


```
<terminated> RemoveDuplicatesSortedArray [Java Application] C:\Program
Enter the number of elements 8
Enter the elements of the sorted array:
1 1 2 2 2 3 4 4
1 2 3 4
```

7. Maximum Index
   Time complexity: O(n log n)
   Space complexity: O(n)



```java
package program14thNov;
import java.util.*;
public class MaximumIndex {
    static int maxIndexDiff(ArrayList<Integer> arr, int n) {
        Map<Integer, ArrayList<Integer>> hashmap = new HashMap<>();
        for (int i = 0; i < n; i++) {
            if (hashmap.containsKey(arr.get(i))) {
                hashmap.get(arr.get(i)).add(i);
            } else {
                hashmap.put(arr.get(i), new ArrayList<Integer>());
                hashmap.get(arr.get(i)).add(i);
            }
        }
        Collections.sort(arr);
        int maxDiff = Integer.MIN_VALUE;
        int temp = n;
        for (int i = 0; i < n; i++) {
            if (temp > hashmap.get(arr.get(i)).get(0)) {
                temp = hashmap.get(arr.get(i)).get(0);
            }
            maxDiff = Math.max(maxDiff, hashmap.get(arr.get(i)).get(hashmap.get(arr.get(i)).size() - 1) - temp);
        }
        return maxDiff;
    }
    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();
        ArrayList<Integer> arr = new ArrayList<>();
        System.out.println("Enter the elements of the array:");
        for (int i = 0; i < n; i++) {
            arr.add(scanner.nextInt());
        }
        int ans = maxIndexDiff(arr, n);
        System.out.println("The maxIndexDiff is: " + ans);
        scanner.close();
    }
}
```

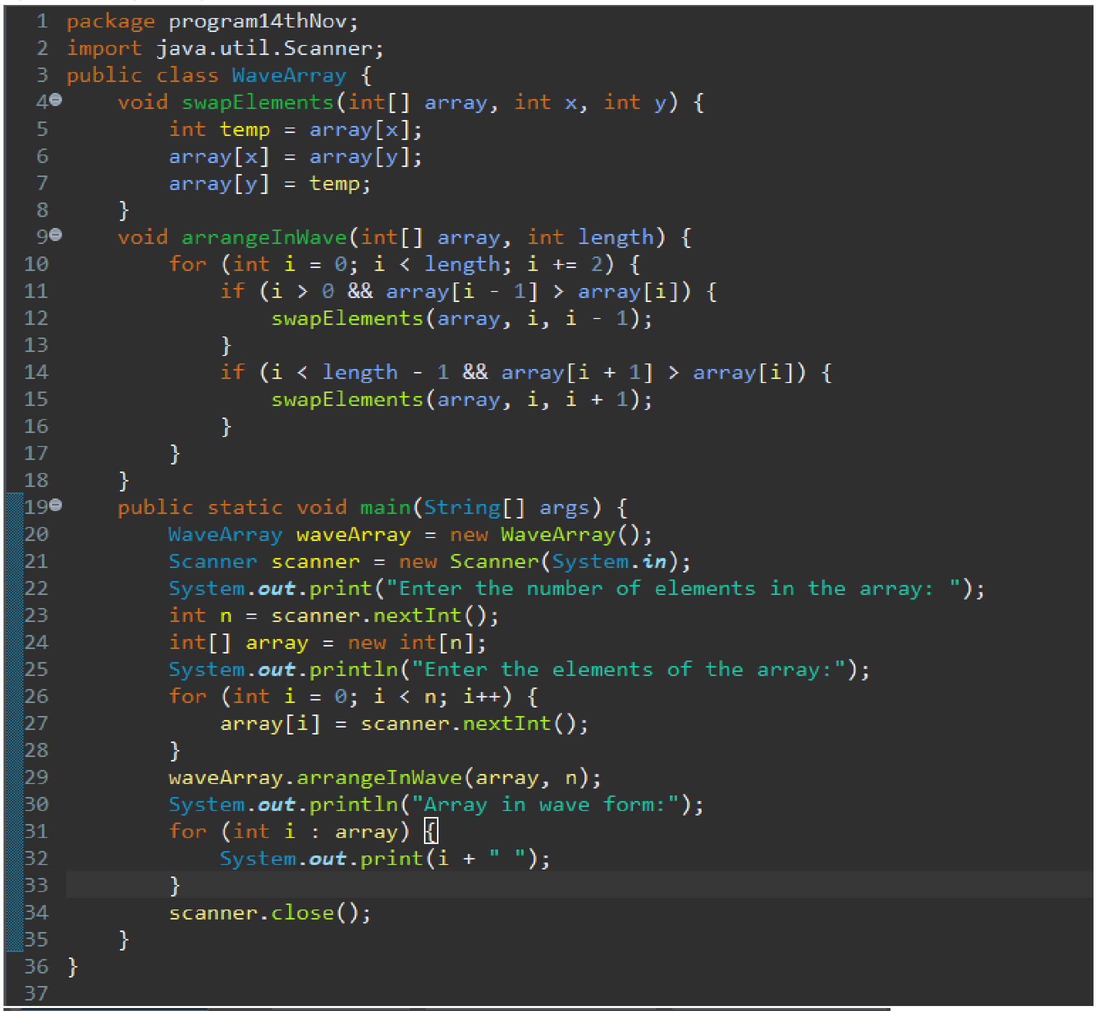


```
<terminated> MaximumIndex [Java Application] C:\Program Files\Java\bin\javaw.ex
Enter the number of elements in the array: 10
Enter the elements of the array:
34 8 10 3 2 80 30 33 1 1
The maxIndexDiff is: 6
```

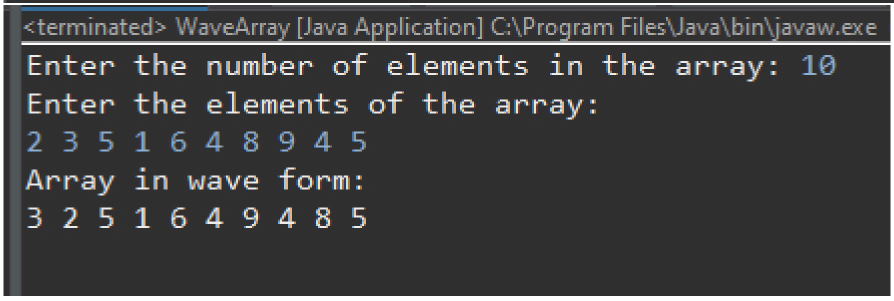
8. Wave Array

Time complexity: O(n)

Space complexity: O(1)



```java
package program14thNov;
import java.util.Scanner;
public class WaveArray {
    void swapElements(int[] array, int x, int y) {
        int temp = array[x];
        array[x] = array[y];
        array[y] = temp;
    }
    void arrangeInWave(int[] array, int length) {
        for (int i = 0; i < length; i += 2) {
            if (i > 0 && array[i - 1] > array[i]) {
                swapElements(array, i, i - 1);
            }
            if (i < length - 1 && array[i + 1] > array[i]) {
                swapElements(array, i, i + 1);
            }
        }
    }
    public static void main(String[] args) {
        WaveArray waveArray = new WaveArray();
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter the number of elements in the array: ");
        int n = scanner.nextInt();
        int[] array = new int[n];
        System.out.println("Enter the elements of the array:");
        for (int i = 0; i < n; i++) {
            array[i] = scanner.nextInt();
        }
        waveArray.arrangeInWave(array, n);
        System.out.println("Array in wave form:");
        for (int i : array) {
            System.out.print(i + " ");
        }
        scanner.close();
    }
}
```



```
<terminated> WaveArray [Java Application] C:\Program Files\Java\bin\javaw.exe
Enter the number of elements in the array: 10
Enter the elements of the array:
2 3 5 1 6 4 8 9 4 5
Array in wave form:
3 2 5 1 6 4 9 4 8 5
```