```java
import java.util.Arrays;

public class BinarySearchExample {
    public static void main(String[] args) {
        int[] arr = {20, 100, 70, 3, 50, 72, 15};
        System.out.println("Original array: " + Arrays.toString(arr));
        mergeSort(arr, 0, arr.length - 1);
        System.out.println("Sorted array: " + Arrays.toString(arr));
        int key = 15;
        long start = System.nanoTime();
        int elementIndex = binarySearch(arr, key);
        long end = System.nanoTime();
        if (elementIndex == -1) {
            System.out.println("Binary Search -" + " Index ");
        } else {
            System.out.println("Binary Search -" + " Index " + elementIndex + " = " + arr[elementIndex]);
        }
        System.out.println("Time Binary search = " + (end - start) + " nanoseconds");
        start = System.nanoTime();
        int linearElementIndex = linearSearch(arr, key);
        end = System.nanoTime();
        if (linearElementIndex == -1) {
            System.out.println("Linear Search -" + " Index");
        } else {
            System.out.println("Linear Search -" + " Index " + linearElementIndex + " = " + arr[linearElementIndex]);
        }
        System.out.println("Time Linear search = " + (end - start) + " nanoseconds");
    }

    public static void mergeSort(int[] arr, int l, int r) {
        if (l < r) {
            int m = (l + r) / 2;
            mergeSort(arr, l, m);
            mergeSort(arr, m + 1, r);
            merge(arr, l, m, r);
        }
    }

    public 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, 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++;
        }
    }

    public static int binarySearch(int[] arr, int key) {
        int left = 0;
        int right = arr.length - 1;
        while (left <= right) {
            int mid = (left + right) / 2;
            if (arr[mid] == key) {
                return mid;
            } else if (arr[mid] < key) {
                left = mid + 1;
            } else {
                right = mid - 1;
            }
        }
        return -1;
    }

    public static int linearSearch(int[] arr, int key) {
        for (int i = 0; i < arr.length; i++) {
            if (arr[i] == key) {
                return i;
            }
        }
        return -1;
    }
}
```