Name: **Shaheer Javaid** Roll no: **2023F-BSE-238** Section: **E**

LAB NO 6

DATA STRUCTURES AND ALGORITHMS

OBJECTIVE: To find an element in linear array using Linear Search and Binary Search.

TASK NO 1:

Declare an array of size 10 to store account balances. Initialize with values 0 to 1000000. Check all array if any value is less than 10000. Show message:

Account No. Low Balance

Account No. Low Balance

INPUT:

```java
package shaheer.javaid;

public class ShaheerJavaid {
    public static void main(String[] args) {
        int[] accountBalances = {5000, 20000, 15000, 9000, 30000, 12000, 8500, 10000, 22000, 7000};
        System.out.println("Accounts with Low Balances:");
        for (int i = 0; i < accountBalances.length; i++) {
            if (accountBalances[i] < 10000) {
                System.out.println("Account No. " + (i + 1) + ": Low Balance (" + accountBalances[i] + ")");
            }
        }
    }
}
```

OUTPUT:

```
Output - Shaheer Javaid (run)
run:
Accounts with Low Balances:
Account No. 1: Low Balance (5000)
Account No. 4: Low Balance (9000)
Account No. 7: Low Balance (8500)
Account No. 10: Low Balance (7000)
BUILD SUCCESSFUL (total time: 0 seconds)
```

TASK NO 2:

Write a program to search in array using Array built-in class.

INPUT:

```java
public class ShaheerJavaid {
    public static void main(String[] args) {
        int[] sortedArray = {10, 20, 30, 40, 50, 60, 70, 80, 90, 100};
        int target = 40;

        int index = Arrays.binarySearch(sortedArray, target);

        if (index >= 0) {
            System.out.println("Element " + target + " found at index: " + index);
        } else {
            System.out.println("Element " + target + " not found.");
        }
    }
}
```

OUTPUT:

```
Output - Shaheer Javaid (run)
run:
Element 40 found at index: 3
BUILD SUCCESSFUL (total time: 0 seconds)
```

Name: **Shaheer Javaid** Roll no: **2023F-BSE-238** Section: **E**

TASK NO 3:

Given an unsorted array `arr` of integers, find the smallest positive integer that is **missing** from the array. You need to implement this using **binary search**. The array can contain both negative numbers and positive numbers, and you can assume that the array does not have duplicates.

INPUT:

```java
public class ShaheerJavaid {
    public static void main(String[] args) {
        int[] arr = {3, 4, -1, 1};

        int missingPositive = findSmallestMissingPositive(arr);

        System.out.println("The smallest missing positive integer is: " + missingPositive);
    }

    public static int findSmallestMissingPositive(int[] arr) {
        Arrays.sort(arr);

        int smallestMissing = 1;

        for (int i = 0; i < arr.length; i++) {
            // Skip non-positive numbers
            if (arr[i] == smallestMissing) {
                smallestMissing++;
            }
        }
        return smallestMissing;
    }
}
```

OUTPUT:

```
Output - Shaheer Javaid (run)
run:
The smallest missing positive integer is: 2
BUILD SUCCESSFUL (total time: 0 seconds)
```

TASK NO 4:

You are given a sorted array `arr[]` and a target element `target`. Your task is to find the **first occurrence** of the target in the array using binary search. If the target is not found, return `-1`. You are given a sorted array `arr[]` and a target element `target`. Your task is to find the **first occurrence** of the target in the array using binary search. If the target is not found, return `-1`.

INPUT:

```java
package shaheer.javaid;
public class ShaheerJavaid {
    public static void main(String[] args) {

        int[] arr = {1, 2, 2, 2, 3, 4, 5, 6, 7, 8};
        int target = 2;
        int firstOccurrenceIndex = findFirstOccurrence(arr, target);
        if (firstOccurrenceIndex != -1) {
            System.out.println("First occurrence of " + target + " is at index: " + firstOccurrenceIndex);
        } else {
            System.out.println("Element " + target + " not found.");
        }
    }

    public static int findFirstOccurrence(int[] arr, int target) {
        int low = 0;
        int high = arr.length - 1;
        int result = -1;
        while (low <= high) {
            int mid = (low + high) / 2;

            if (arr[mid] == target) {
                result = mid;
                high = mid - 1;
            }
            else if (arr[mid] < target) {
                low = mid + 1;
            }
            }
            else {
                high = mid - 1;
            }
        }
        return result;
    }
}
```

OUTPUT:

```
Output - Shaheer Javaid (run)
First occurrence of 2 is at index: 1
BUILD SUCCESSFUL (total time: 0 seconds)
```

Name: **Shaheer Javaid** Roll no: **2023F-BSE-238** Section: **E**

HOME TASKS

TASK NO 1:

Write a program initializing array of size 20 and search an element using binary search.

INPUT:

```java
package shaheer.javaid;
import java.util.Arrays;

public class ShaheerJavaid {
    public static void main(String[] args) {
        int[] arr = new int[20];
        for (int i = 0; i < arr.length; i++) {
            arr[i] = (int) (Math.random() * 100);
        }

        Arrays.sort(arr);

        System.out.println("Sorted Array: " + Arrays.toString(arr));

        int target = 50;
        int index = binarySearch(arr, target);

        if (index != -1) {
            System.out.println("Element " + target + " found at index: " + index);
        } else {
            System.out.println("Element " + target + " not found.");
        }
    }

    public static int binarySearch(int[] arr, int target) {
        int low = 0;
        int high = arr.length - 1;

        while (low <= high) {
            int mid = (low + high) / 2;

            if (arr[mid] == target) {
                return mid;
            } else if (arr[mid] < target) {
                low = mid + 1;
            } else {
                high = mid - 1;
            }
        }
        return -1;
    }
}
```

OUTPUT:

```
Output - Shaheer Javaid (run)
run:
Sorted Array: [2, 9, 22, 26, 30, 37, 41, 42, 43, 48, 51, 52, 54, 57, 57, 79, 83, 85, 88, 90]
Element 50 not found.
BUILD SUCCESSFUL (total time: 0 seconds)
```

TASK NO 2

Write a function called occurrences that, given an array of numbers A, prints all the distinct values in A each followed by its number of occurrences.

For example, if A = (28, 1, 0, 1, 0, 3, 4, 0, 0, 3), the function should output the following five lines (here separated by a semicolon) “28 1; 1 2; 0 4; 3 2; 4 1”.

INPUT:

```java
package shaheer.javaid;
import java.util.HashMap;
import java.util.Map;

public class ShaheerJavaid {
    public static void main(String[] args) {
        int[] A = {28, 1, 0, 1, 0, 3, 4, 0, 0, 3};
        occurrences(A);
    }
    public static void occurrences(int[] A) {
        Map<Integer, Integer> countMap = new HashMap<>();
        for (int num : A) {
            countMap.put(num, countMap.getOrDefault(num, 0) + 1);
        }
        StringBuilder result = new StringBuilder();
        for (Map.Entry<Integer, Integer> entry : countMap.entrySet()) {
            result.append(entry.getKey())
                  .append(" ")
                  .append(entry.getValue())
                  .append("; ");
        }
        if (result.length() > 0) {
            result.setLength(result.length() - 2);
        }
        System.out.println(result.toString());
    }
}
```

OUTPUT:

utput - Shaheer Javaid (run)

```
run:
0 4; 1 2; 3 2; 4 1; 28 1
BUILD SUCCESSFUL (total time: 0 seconds)
```

Name: **Shaheer Javaid** Roll no: **2023F-BSE-238** Section: **E**

TASK NO 3:

Assume a bank's system needs to identify accounts with critically low balances and alert the user. Test the function with various balance values to ensure it correctly identifies all accounts below the threshold.

INPUT:

```java
package shaheer.javaid;

public class ShaheerJavaid {
    public static void main(String[] args) {
        double[] balances = {12000.50, 8000.30, 5000.00, 3000.25, 15000.75, 9500.00, 2000.00};
        double threshold = 10000.00;
        checkLowBalances(balances, threshold);
    }

    public static void checkLowBalances(double[] balances, double threshold) {
        for (int i = 0; i < balances.length; i++) {
            if (balances[i] < threshold) {
                System.out.println("Account No. " + (i + 1) + " has critically low balance: $" + balances[i]);
            }
        }
    }
}
```

OUTPUT:

```
Output - Shaheer Javaid (run)
run:
Account No. 2 has critically low balance: $8000.3
Account No. 3 has critically low balance: $5000.0
Account No. 4 has critically low balance: $3000.25
Account No. 6 has critically low balance: $9500.0
Account No. 7 has critically low balance: $2000.0
BUILD SUCCESSFUL (total time: 0 seconds)
```