**LAB#06**

**Lab Task#01:**

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.

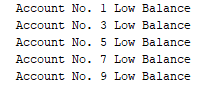
**Code: Output:**

public class LAB06 {

public static void main(String[] args) {

int[] balances = {5000, 12000, 7000, 15000, 8000, 20000, 1000, 25000, 6000, 1000000};

for (int i = 0; i < balances.length; i++) {

 if (balances[i] < 10000) {

System.out.println("Account No. " + (i + 1) + " Low Balance"); }

}

}

}

**Lab Task#02:**

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

**Code: Output:**

import java.util.Arrays;

public class LAB06 {

public static void main(String[] args) {

int[] numbers = {10, 20, 30, 40, 50, 60, 70};

Arrays.sort(numbers);

int search = 40;

int index = Arrays.binarySearch(numbers, search);

if (index >= 0) {

System.out.println("Element " + search + " found at index " + index);

} else {

System.out.println("Element " + search + " not found"); }

}

}

**Lab Task#03:**

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.

**Code: Output:**

import java.util.Arrays;

public class LAB06 {

public static void main(String[] args) {

int[] arr = {3, 4, -1, 1};

System.out.println("Smallest Missing +ve Integer: " + findMissing(arr));}

public static int findMissing(int[] arr) {

arr = Arrays.stream(arr).filter(x -> x > 0).sorted().toArray();

int left = 0, right = arr.length - 1;

while (left <= right) {

int mid = left + (right - left) / 2;

if (arr[mid] == mid + 1) {

left = mid + 1;

} else {

right = mid - 1; }

}

return left + 1; }

}

**Lab Task#04:**

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.

**Code: Output:**

public class LAB06 {

 public static void main(String[] args) {

int[] arr = {1, 2, 2, 2, 3, 4, 5};

int target = 2;

System.out.println(FirstOccurrence(arr, target));}

public static int FirstOccurrence(int[] arr, int target) {

int left = 0, right = arr.length - 1, result = -1;

while (left <= right) {

int mid = left + (right - left) / 2;

if (arr[mid] == target) {

result = mid;

right = mid - 1;

} else if (arr[mid] < target) {

left = mid + 1;

} else {

right = mid - 1; }

}

return result;}

}

**Home Task#01:**

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

**Code: Output:**

import java.util.Arrays;

public class LAB06 {

public static void main(String[] args) {

 int[] arr = {10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 110, 120, 130, 140, 150, 160, 170, 180, 190, 200};

int target = 70;

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 left = 0, right = arr.length - 1;

while (left <= right) {

int mid = left + (right - left) / 2;

if (arr[mid] == target) {

return mid;

} else if (arr[mid] < target) {

left = mid + 1;

} else {

right = mid - 1; }

}

return -1; }

}

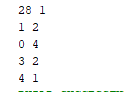
**Home Task#02:**

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.

**Code: Output:**

public class LAB06 {

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) {

boolean[] visited = new boolean[A.length];

for (int i = 0; i < A.length; i++) {

if (visited[i]) continue;

int count = 1;

for (int j = i + 1; j < A.length; j++) {

if (A[i] == A[j]) {

count++;

visited[j] = true; }

}

System.out.println(A[i] + " " + count);}

}

}

**Home Task#03:**

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.

**Code: Output:**

public class LAB06 {

public static void main(String[] args) {

double[] balances = {5000.0, 800.0, 1200.0, 100.0, 2500.0, 950.0, 200.0};

double threshold = 1000.0;

checkLowBalances(balances, threshold);}

public static void checkLowBalances(double[] balances, double threshold) {

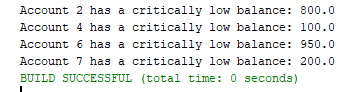
for (int i = 0; i < balances.length; i++) {

if (balances[i] < threshold) {

System.out.println("Account " + (i + 1) + " has a low balance: " + balances[i]);}

}

}

}