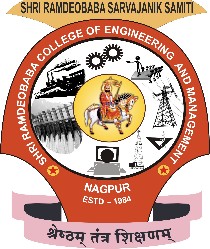
**Shri Ramdeobaba College of Engineering & Management Nagpur-13**

**Department of Computer Application**

**Session: 2023-2024**



**Submission for**

**Course Name:** Design Analysis and Algorithm Lab

**Course Code:** MCP546

**Name of the Student:** Jayesh Lalit Nandanwar

**Class Roll No:** 26

**Semester:** MCA II semester

**Shift:** 2

**Batch:** 2

Under the Guidance of

Prof. Manda Ukey

Date of submission: 23/02/2024

**Practical 3**

**Aim:** Perform binary search on the output files of practical2 and sequential file of practical\_0.

Display the time taken to search a key from these files occurring in the first, middle and last position / not found case.

Compare the time taken for all these cases.

**For file 1 from Practical 0 (Sequential unrepeated numbers):**

**Code:** import java.io.\*;

import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;

public class BinarySearchPrac0File1 {

public static void main(String[] args) throws IOException {

Scanner scanner = new Scanner(System.in);

FileReader f = new FileReader("./sequentialUnrepeatedNumbers.txt");

Scanner fileScanner = new Scanner(f);

List<Integer> intArray = new ArrayList<Integer>(); // maximum of 100000 elements in the file

while (fileScanner.hasNextInt()) {

intArray.add(fileScanner.nextInt());

}

fileScanner.close();

System.out.print("Enter Element To Find: ");

int elementToFind = scanner.nextInt();

long start = System.nanoTime();

int result=binarySearch(intArray, elementToFind);

long finish = System.nanoTime();

if (result == -1){

System.out.println("Element not present");

}

else{

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

}

long timeElapsed = finish - start;

System.out.println("\nTime taken for searching : " + timeElapsed + " nanoseconds");

scanner.close();

}

public static int binarySearch(List<Integer> intArray, int x){

int startInt = 0;

int endInt = intArray.size() - 1;

while (startInt <= endInt) {

int mid = (startInt + endInt) / 2;

if (intArray.get(mid) == x) {

return mid;

} else if (intArray.get(mid) > x) {

endInt = mid - 1;

} else {

startInt = mid + 1;

}

}

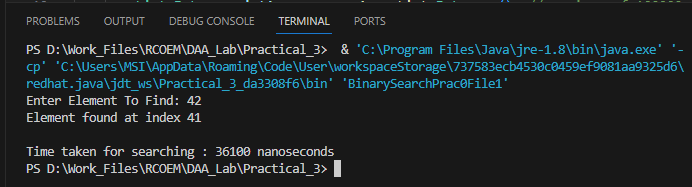
return -1;

}

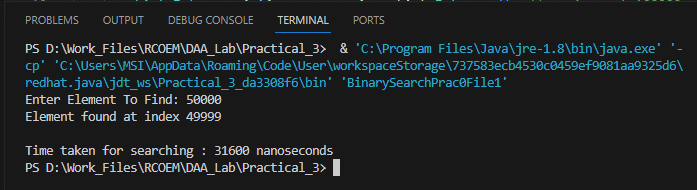
}

**Output:**

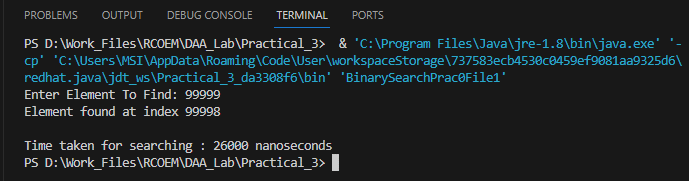
Element present at beginning:



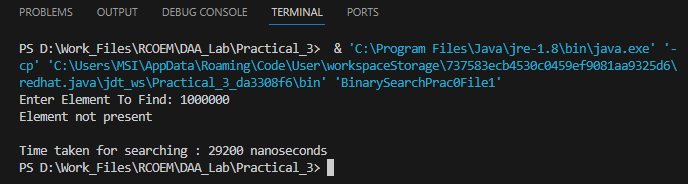
Element present at middle:



Element present at end:



Element not present**:**

****

**For file 2 from Practical 2 (Random Unrepeated Numbers Sorted Output):**

**Code:** import java.io.\*;

import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;

public class BinarySearchPrac2File1 {

public static void main(String[] args) throws IOException {

Scanner scanner = new Scanner(System.in);

FileReader f = new FileReader("./randomUnrepeatedNumbersSortedOutput.txt");

Scanner fileScanner = new Scanner(f);

List<Integer> intArray = new ArrayList<Integer>(); // maximum of 100000 elements in the file

while (fileScanner.hasNextInt()) {

intArray.add(fileScanner.nextInt());

}

fileScanner.close();

System.out.print("Enter Element To Find: ");

int elementToFind = scanner.nextInt();

long start = System.nanoTime();

int result=binarySearch(intArray, elementToFind);

long finish = System.nanoTime();

if (result == -1){

System.out.println("Element not present");

}

else{

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

}

long timeElapsed = finish - start;

System.out.println("\nTime taken for searching : " + timeElapsed + " nanoseconds");

scanner.close();

}

public static int binarySearch(List<Integer> intArray, int x){

int startInt = 0;

int endInt = intArray.size() - 1;

while (startInt <= endInt) {

int mid = (startInt + endInt) / 2;

if (intArray.get(mid) == x) {

return mid;

} else if (intArray.get(mid) > x) {

endInt = mid - 1;

} else {

startInt = mid + 1;

}

}

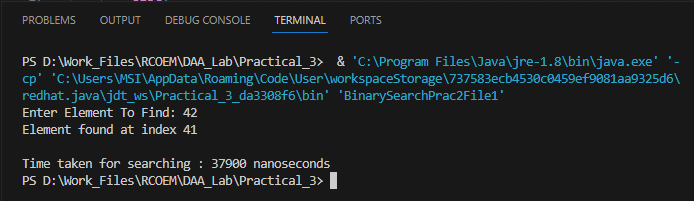
return -1;

}

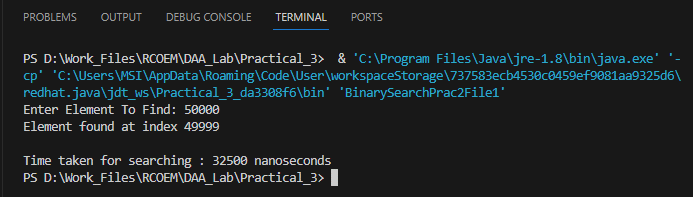
}

**Output:**

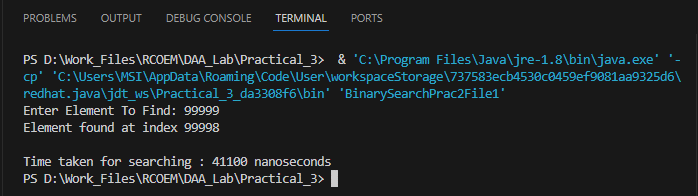
Element present at beginning:



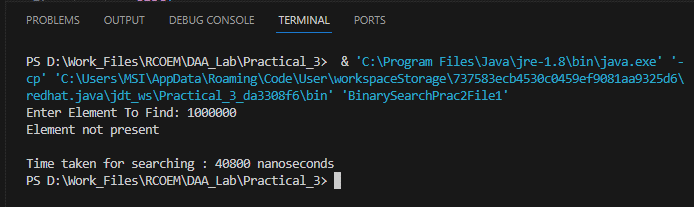
Element present at middle:



Element present at end:



Element not present:



**For file 3 from Practical 2 (Random Repeated Numbers Sorted Output):**

**Code:** import java.io.\*;

import java.util.ArrayList;

import java.util.List;

import java.util.Scanner;

public class BinarySearchPrac2File2 {

public static void main(String[] args) throws IOException {

Scanner scanner = new Scanner(System.in);

FileReader f = new FileReader("./randomRepeatedNumbersSortedOutput.txt");

Scanner fileScanner = new Scanner(f);

List<Integer> intArray = new ArrayList<Integer>(); // maximum of 100000 elements in the file

while (fileScanner.hasNextInt()) {

intArray.add(fileScanner.nextInt());

}

fileScanner.close();

System.out.print("Enter Element To Find: ");

int elementToFind = scanner.nextInt();

long start = System.nanoTime();

int result=binarySearch(intArray, elementToFind);

long finish = System.nanoTime();

if (result == -1){

System.out.println("Element not present");

}

else{

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

}

long timeElapsed = finish - start;

System.out.println("\nTime taken for searching : " + timeElapsed + " nanoseconds");

scanner.close();

}

public static int binarySearch(List<Integer> intArray, int x){

int startInt = 0;

int endInt = intArray.size() - 1;

while (startInt <= endInt) {

int mid = (startInt + endInt) / 2;

if (intArray.get(mid) == x) {

return mid;

} else if (intArray.get(mid) > x) {

endInt = mid - 1;

} else {

startInt = mid + 1;

}

}

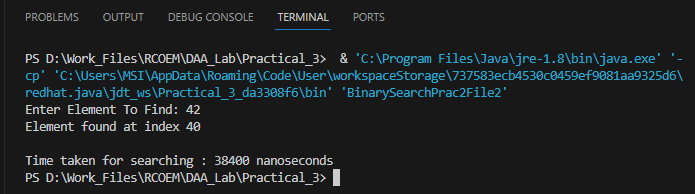
return -1;

}

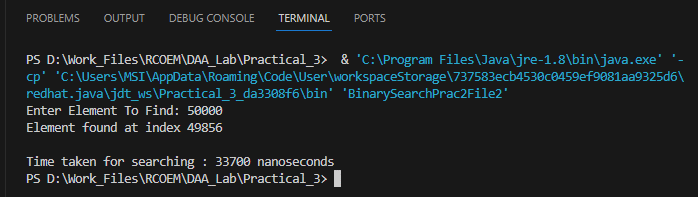
}

**Output:**

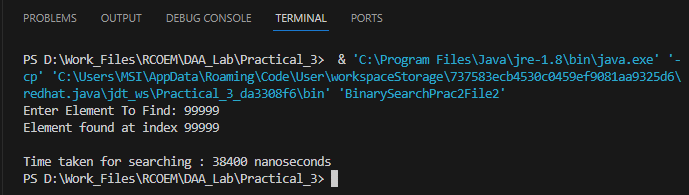
Element present at beginning:



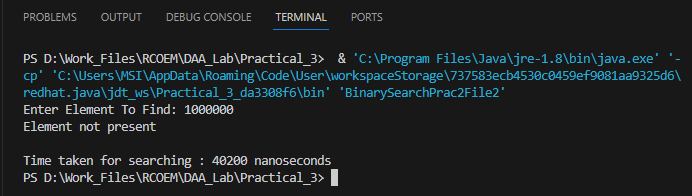
Element present at middle:



Element present at end:



Element not present:



**Comparison between times taken:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **File**  **File 1**  From Practical 0  (Sequential unrepeated numbers) | **Element At** | **Element** | **Found At Index** | **Time Taken (in nanoseconds)** |
| **File 1**  From Practical 0  (Sequential unrepeated numbers) | Start | 42 | 41 | 36100 |
| Middle | 50000 | 49999 | 31600 |
| End | 99999 | 99998 | 26000 |
| Not Found | 1000000 | - | 29200 |
| **File 2**  From Practical 2  (Random Unrepeated Numbers Sorted Output) | Start  **File 2**  From Practical 2  (Random Unrepeated Numbers Sorted Output) | 42 | 41 | 37900 |
| Middle | 50000 | 49999 | 32500 |
| End | 99999 | 99998 | 41100 |
| Not Found  **File 3**  From Practical 2  (Random Repeated Numbers Sorted Output) | 1000000 | - | 40800 |
| **File 3**  From Practical 2  (Random Repeated Numbers Sorted Output) | Start | 42 | 40 | 38400 |
| Middle | 50000 | 49856 | 33700 |
| End | 99999 | 99999 | 38400 |
| Not Found | 1000000 | - | 40200 |