DAA ASSIGNMENT 2

NAME:- Raunak Thanawala

Assignment:- Binary and Linear Search

Batch:- CE Group C

AIM:-

To Write an algorithm for Linear Search and Binary Search and to write a program to solve given problem using your algorithms and applying coding style in your programs.

CODE:-

*/\*\*Name:Raunak Thanawala*

*\* Date:10/8/24*

*\* Experiment: Binary and Linear Search*

*\*/*

#include <iostream>

#include <cstdlib>

namespace searching\_project {

*//Conducts Linear Search*

int LSearch(*const* int karr[], int n, int key) {

    for(int i = 0; i < n; ++i) {

        if (karr[i] == key) {

            return i;

        }

    }

    return -1;

}

*//Conducts Binary Search*

int BSearch(*const* int karr[], int start, int end, int key) {

    if(start <= end)

    {

        int mid = (start + end) / 2;

        if(karr[mid] == key) {

            return mid;

        }   else if(karr[mid] > key) {

            return BSearch(karr, start, mid-1, key);

        }   else {

            return BSearch(karr, mid+1, end, key);

        }

    }

    return -1;

}

*//Checks if Given Array is Sorted*

bool IsSorted(*const* int karr[], int n) {

    for(int i = 0; i < n-1; i++) {

        if(karr[i] > karr[i+1]) {

            return false;

        }

    }

    return true;

}

} *// namespace searching\_project*

int main() {

    int n;

    std::cout << "Enter the size of array: "<< std::endl;

    std::cin >> n;

    if(n == 0) {

        std::cerr << "Array is Empty"<< std::endl;

        std::exit(EXIT\_FAILURE);

    }

    int\* arr = new int[n];

    std::cout << "Enter the array: "<< std::endl;

    for(int j = 0; j < n; ++j) {

        std::cin >> arr[j];

    }

    int key;

    std::cout << "Enter the element to be searched in array: "<< std::endl;

    std::cin >> key;

    int linear\_search\_result = searching\_project::LSearch(arr,n,key);

    if(linear\_search\_result != -1) {

        std::cout << "Index at which " << key << " is located is "

                  << linear\_search\_result << " through Linear Search" << std::endl;

    }

    else {

        std::cout << key << " not found in array with Linear Search" << std::endl;

    }

*//Ensure Array is Sorted Before Binary Search*

    bool is\_sorted = searching\_project::IsSorted(arr,n);

    if(!is\_sorted) {

        std::cout<< "Array not sorted";

        std::exit(EXIT\_FAILURE);

    }

    int binary\_search\_result = searching\_project::BSearch(arr, 0, n-1, key);

    if(binary\_search\_result != -1) {

        std::cout << "Index at which " << key << " is located is "

                  << binary\_search\_result << " through Binary Search" << std::endl;

    }

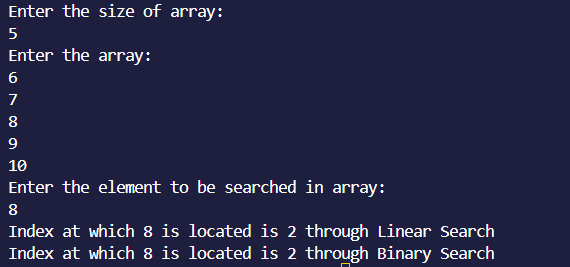
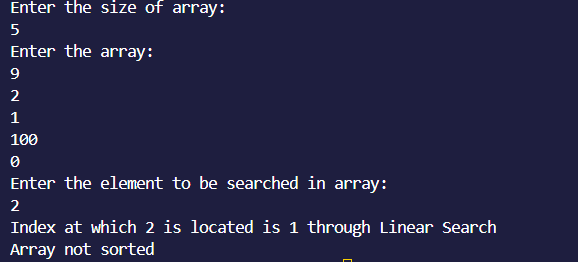
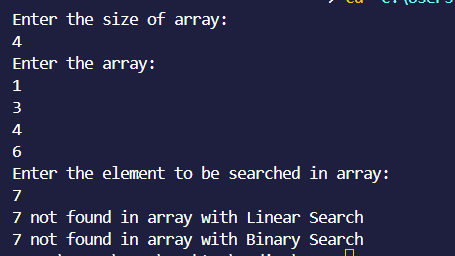
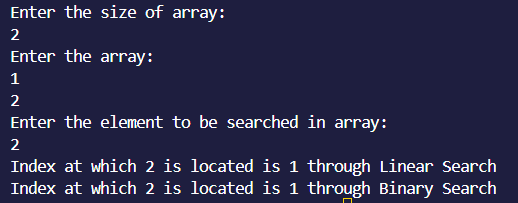
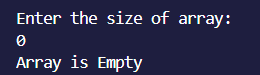
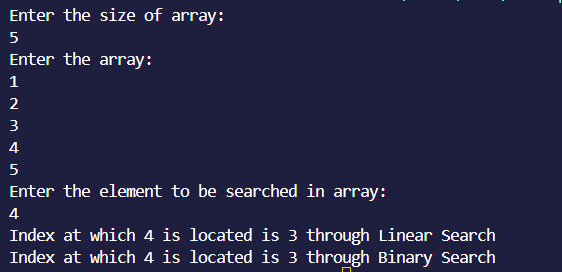
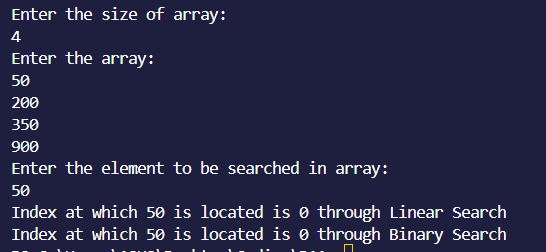
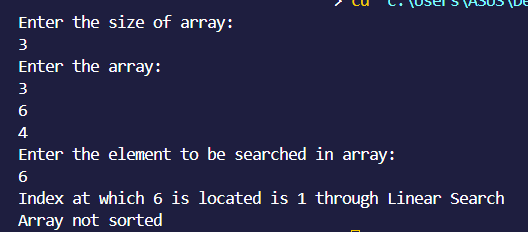
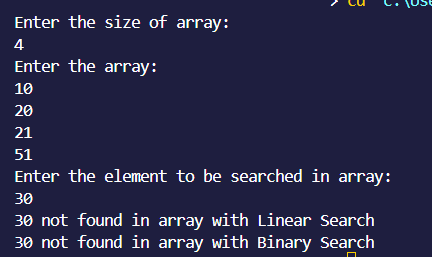
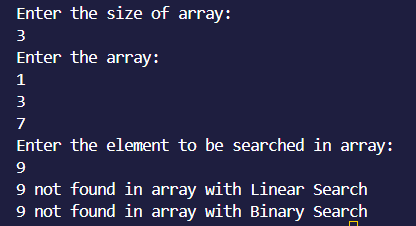
    else {

        std::cout << key << " not found in array with Binary Search" << std::endl;

    }

}

TESTCASES IN CODE :-

1. 
2. 
3. 
4. 
5. 
6. 
7. 
8. 
9. 
10. 

CONCLUSION :-

So In This Experiment we have Written Pseudo Code Algorithms for Linear and Binary Search with Input and Output given outside of the algorithms.

We have also written 5 Test Cases Each for Linear and Binary Search where half of them are Positive and half of them are Negative. The testcases are written by giving all the inputs and then showing the ideal output.

Then we wrote a program following the c++ google style guide such that the ideal outputs were obtained in the program from the testcases.

We then executed the Program to give the Outputs for the given testcases.

We have also found out the Time Complexity of Linear and Binary Search with their Mathematical Analyses.