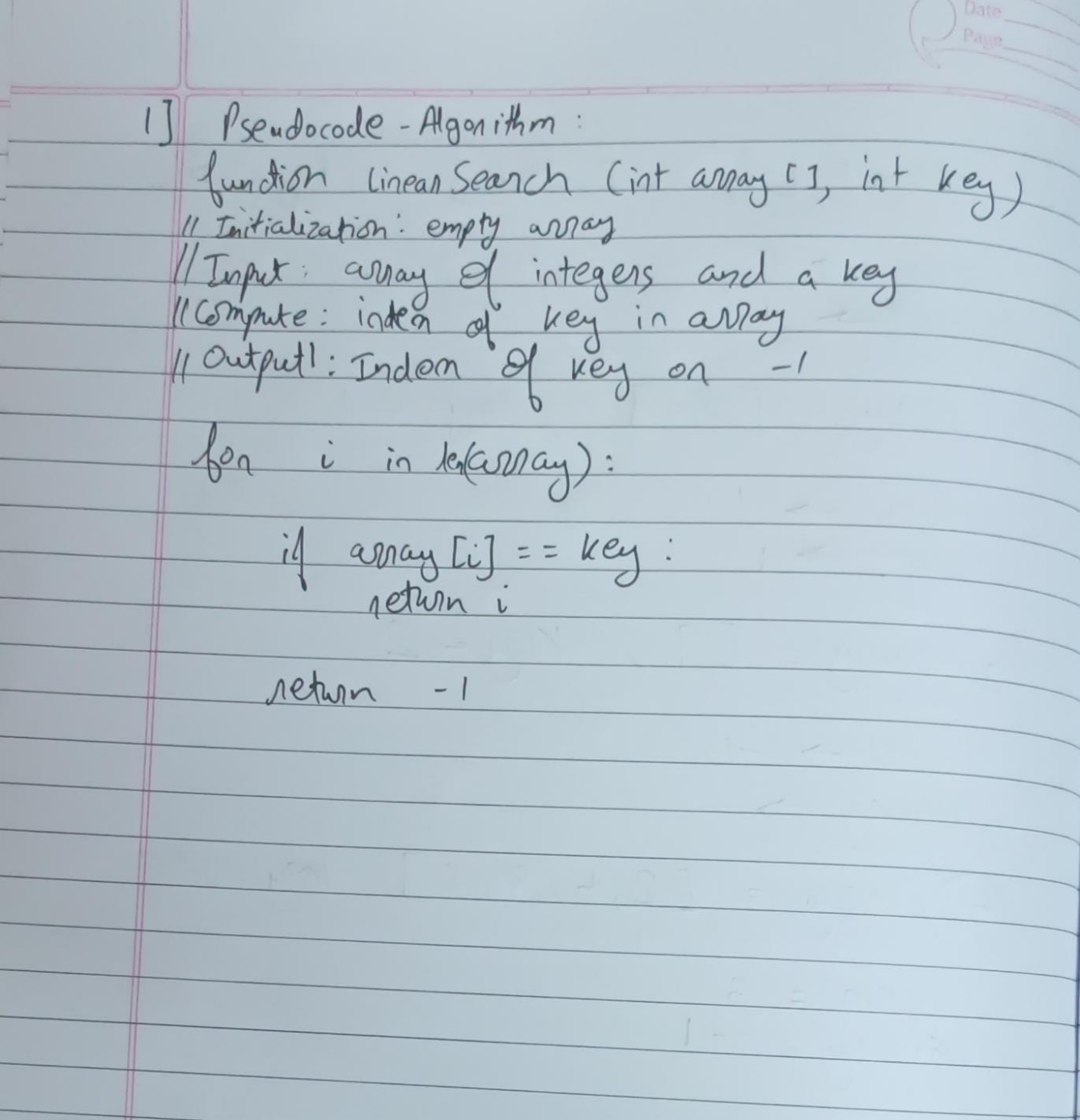
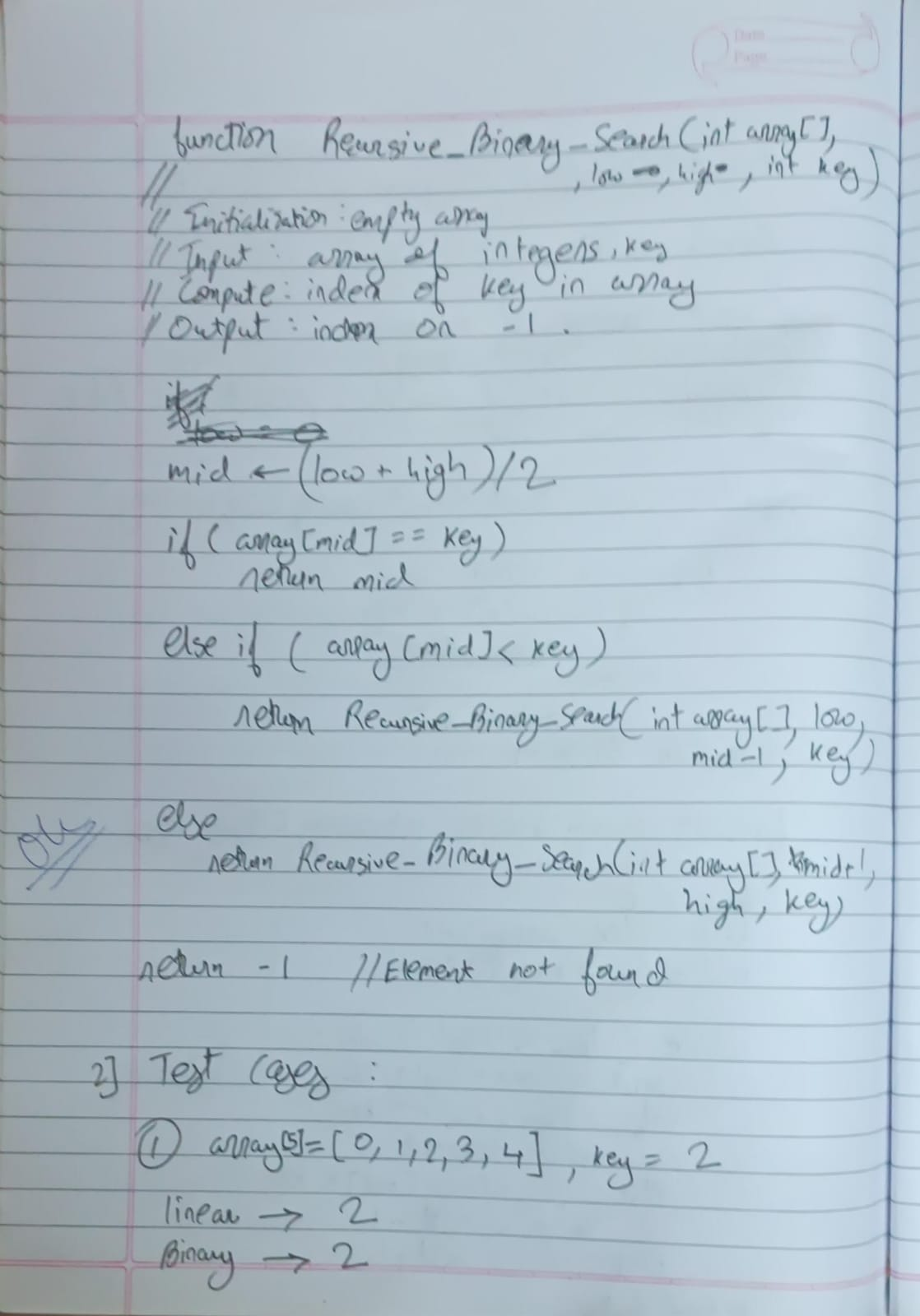
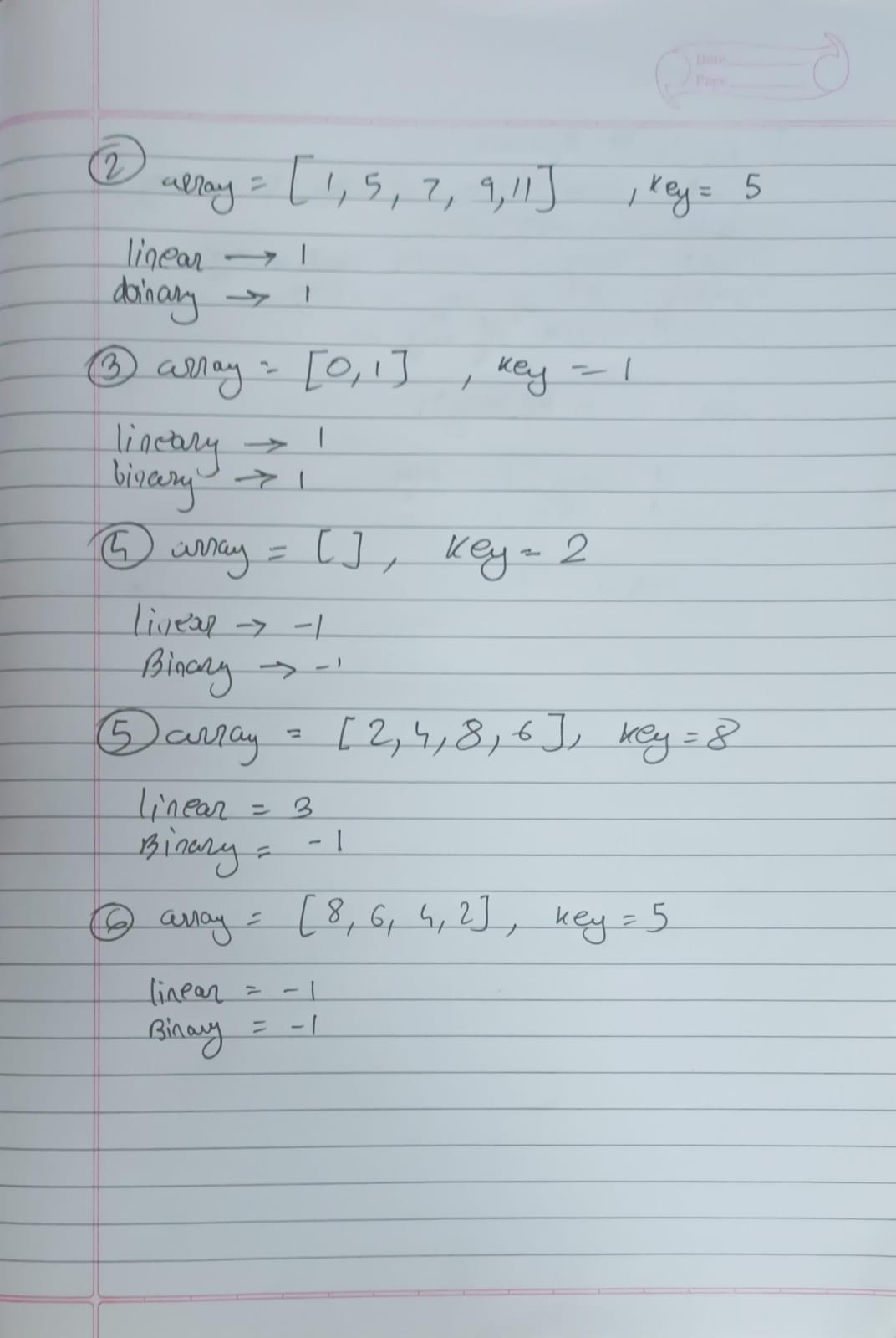
**DAA ASSIGNMENT 2**

AIM: To implement Linear & Binary Search algorithms, find their time complexities and input different test cases.

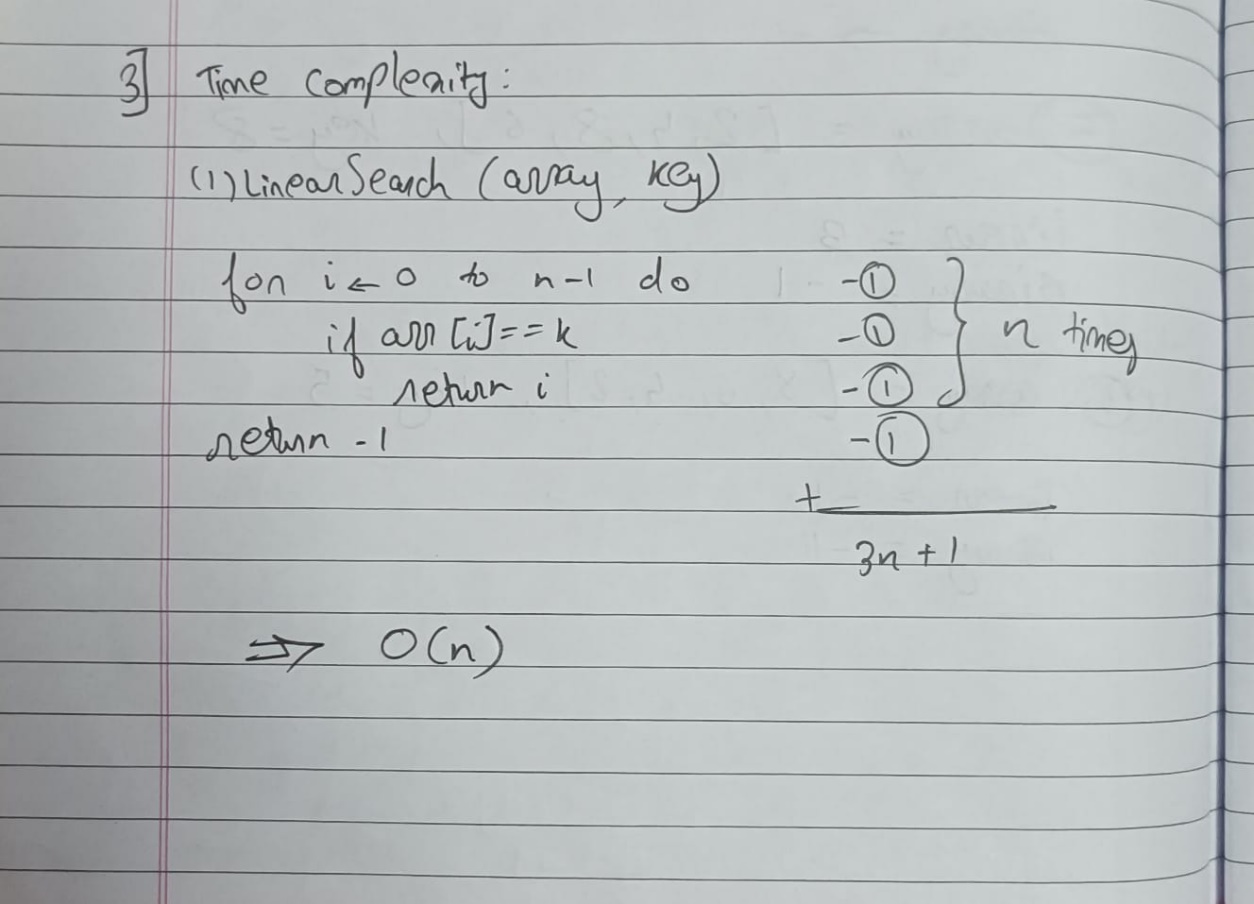
Algorithm and Test Cases:

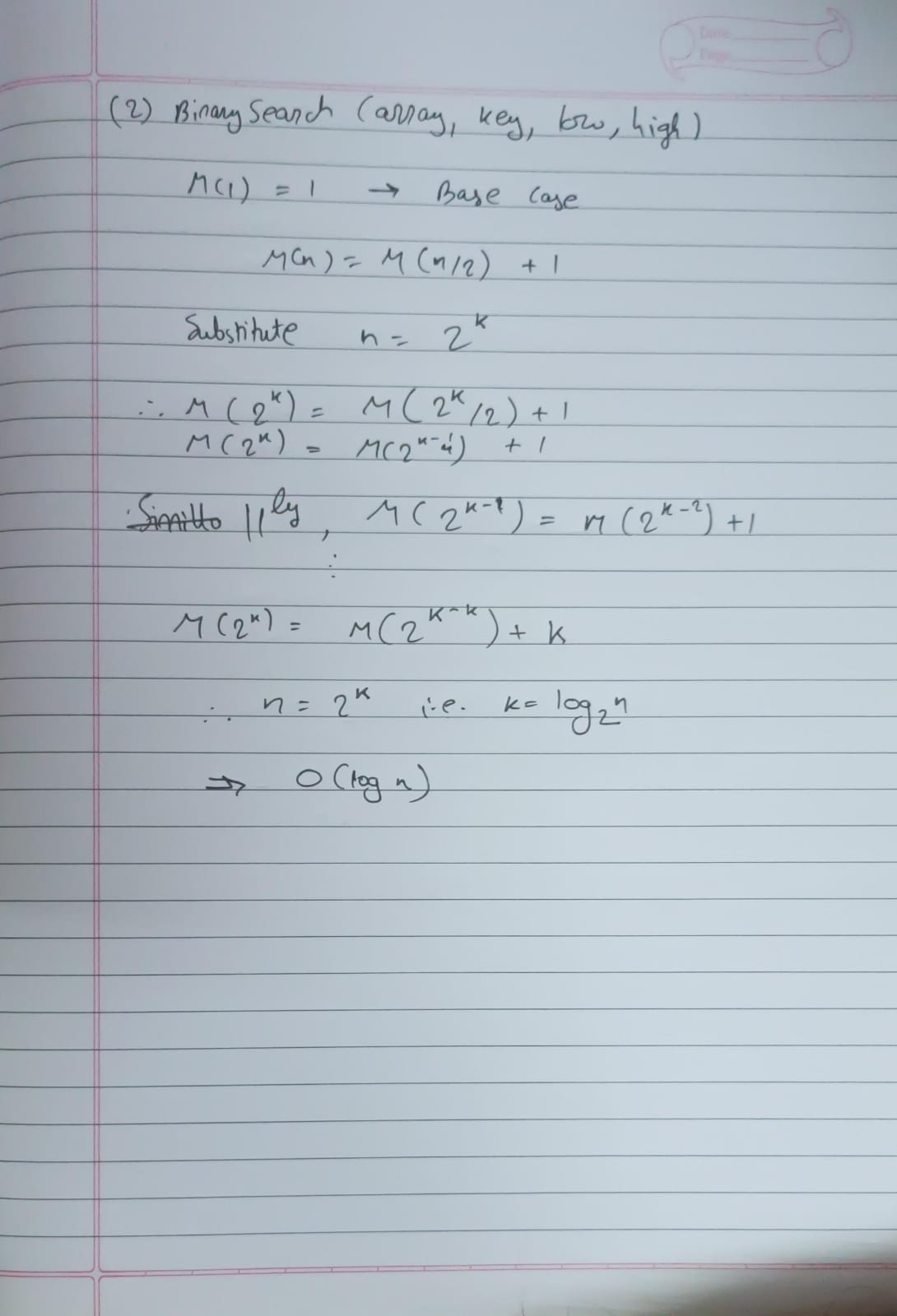






Time Complexity:





Program:

#include <iostream>

using namespace std;

int linearSearch(int array[], int size, int key) {

    for (int i = 0; i < size; i++) {

        if (array[i] == key) {

            return i;

        }

    }

    return -1;

}

int binarySearch(int array[], int key, int low, int high) {

    if (low <= high) {

        int mid = low + (high - low) / 2;

        if (array[mid] == key) {

            return mid;

        } else if (array[mid] < key) {

            return binarySearch(array, key, mid + 1, high);

        } else {

            return binarySearch(array, key, low, mid - 1);

        }

    }

    return -1; // Element not found

}

int main() {

    const int ARRAY\_SIZE = 5;

    int array[ARRAY\_SIZE] = {1, 2, 3, 4, 5};

    int key = 3;

    int size = sizeof(array) / sizeof(array[0]);

    int index1 = linearSearch(array, size, key);

    if (index1 != -1) {

        cout << "Element found at index " << index1 << endl;

    } else {

        cout << "Element not found using linear search" << endl;

    }

    int index2 = binarySearch(array, key, 0, size - 1);

    if (index2 != -1) {

        cout << "Element found at index " << index2 << endl;

    } else {

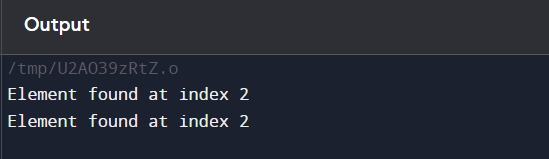
        cout << "Element not found using binary search" << endl;

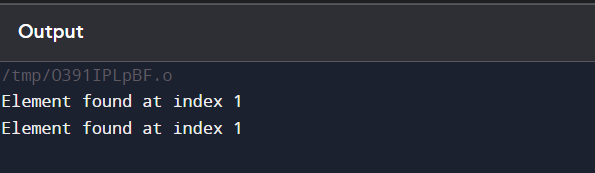
    }

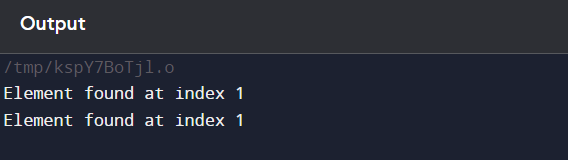
    return 0;

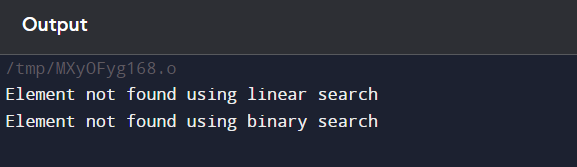
}

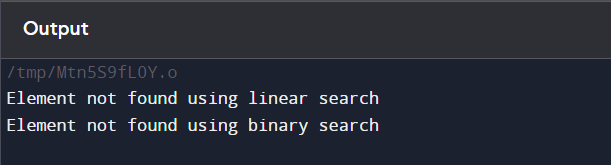
Output:

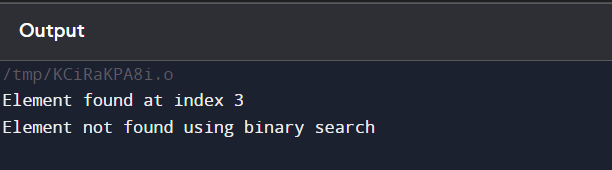


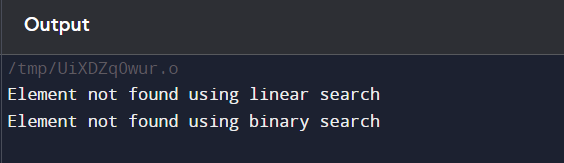












Conclusion:

Hence we implemented linear & binary search using the algorithms and applied coding style in the program. We also calculated their time complexities mathematically and tested different test cases on the program.