Lab Experiment: 07

Student Detail:

- **Name:** Prashant Joshi
- **Student ID:** 590010879
- **Branch:** MCA
- **Batch:** B1
- **Instructor:** Dr. Sourbh Kumar

Assignment 1st: Linear Search Implementation

Definition: Linear search is a simple searching algorithm that checks each element in the list sequentially until the desired element is found or the end of the list is reached.

Tasks:

- Write a C program to implement linear search.
- The program should take an array and a target value as inputs and search for the target within the array.
- Display the index where the target value is found, or indicate if it is not present.

Testing: Use an example array of unsorted elements to demonstrate the search process.

Solution:

```
#include <stdio.h>


// Function to implement linear search
int linearSearch(int arr[], int size, int target) {
    // Traverse the array sequentially
    for (int i = 0; i < size; i++) {
        if (arr[i] == target) {
            return i;  // Return the index of the target if found
        }
    }
    return -1;  // Return -1 if the target is not found
}


int main() {
    int n, target;


    // Take input for the array size
    printf("Enter the number of elements in the array: ");
    scanf("%d", &n);


    int arr[n];
```

```
    // Take input for the array elements
    printf("Enter the elements of the array:\n");
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }

    // Take input for the target value
    printf("Enter the target value to search: ");
    scanf("%d", &target);

    // Perform linear search
    int result = linearSearch(arr, n, target);

    // Display the result
    if (result != -1) {
        printf("Target %d found at index %d.\n", target, result);
    } else {
        printf("Target %d not found in the array.\n", target);
    }

    return 0;
}
```

Input:

```
Enter the number of elements in the array: 6
Enter the elements of the array:
5 3 8 1 2 7
Enter the target value to search: 1
```

Output:

```
Target 1 found at index 3.
```

Assignment 2nd: Binary Search Implementation

Definition: Binary search is a highly efficient searching algorithm that only works on sorted arrays. It divides the search interval in half repeatedly to locate the target value.

Tasks:

- Write a C program to implement binary search.
- The program should prompt the user to enter a sorted array and a target value.
- Display the index where the target value is found, or indicate if it is not present.

Testing: Use a sorted example array to demonstrate the search process and show each step as the interval is divided.

Solution:

```
#include <stdio.h>


// Function to implement binary search
int binarySearch(int arr[], int size, int target) {
    int left = 0;
    int right = size - 1;


    // Continue searching as long as the interval is valid
    while (left <= right) {
        int mid = left + (right - left) / 2;  // Calculate middle index with overflow protection


        printf("Current search interval: [%d, %d], Middle index: %d, Middle element: %d\n", left, right, mid, arr[mid]);


        // If target is found at mid, return the index
        if (arr[mid] == target) {
            return mid;
        }


        // If target is smaller, search in the left half
        if (arr[mid] > target) {
            right = mid - 1;
        }
```

```
        // If target is larger, search in the right half
        else {
            left = mid + 1;
        }
    }

    // Return -1 if target is not found
    return -1;
}

int main() {
    int n, target;

    // Take input for the size of the array
    printf("Enter the number of elements in the sorted array: ");
    scanf("%d", &n);

    int arr[n];

    // Take input for the sorted array elements
    printf("Enter the sorted elements of the array:\n");
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }

    // Take input for the target value
    printf("Enter the target value to search: ");
    scanf("%d", &target);

    // Perform binary search
    int result = binarySearch(arr, n, target);

    // Display the result
```

```
    if (result != -1) {
        printf("Target %d found at index %d.\n", target, result);
    } else {
        printf("Target %d not found in the array.\n", target);
    }

    return 0;
}
```

Output:

```
Enter the number of elements in the sorted array: 7
Enter the sorted elements of the array:
2 4 6 8 10 12 14
Enter the target value to search: 10
```

```
Current search interval: [0, 6], Middle index: 3, Middle element: 8
Current search interval: [4, 6], Middle index: 5, Middle element: 12
Current search interval: [4, 4], Middle index: 4, Middle element: 10
Target 10 found at index 4.
```