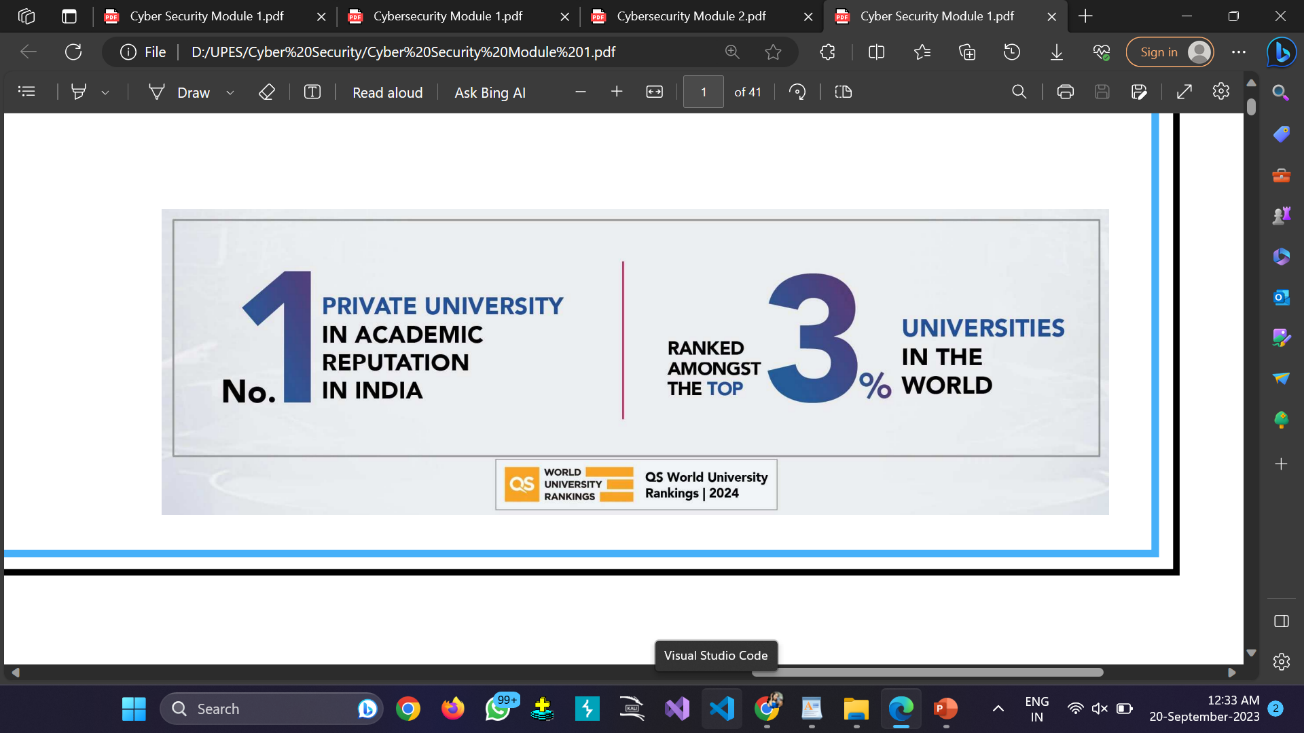
A picture containing text, clipart

Description automatically generated

1

**Lab Experiment: 07**

**Student Detail:**

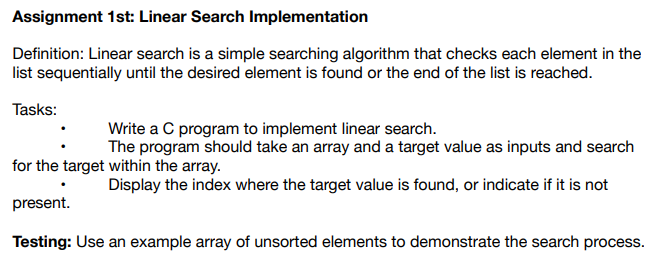
**• Name:** Prashant Joshi

**• Student ID:** 590010879

**• Branch:** MCA

**• Batch:** B1

**• Instructor:** Dr. Sourbh Kumar



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 {

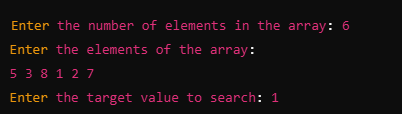
printf("Target %d not found in the array.\n", target);

}

return 0;

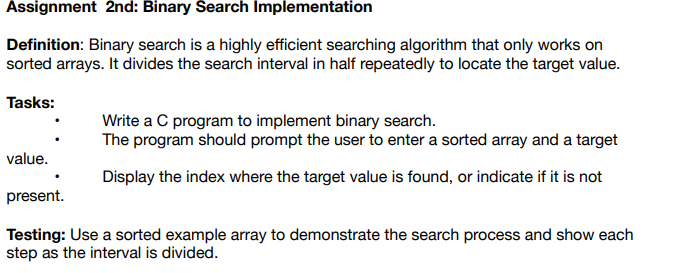
}

Input:



Output:





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:

