**Lab 3**

**Mark lopes**

**9913**

**SE-Comps\_A Batch-C**

**Linear search:-**

#include <stdio.h>

int linearSearch(int arr[], int size, int target) {

    for (int i = 0; i < size; i++) {

        if (arr[i] == target) {

            return i;  // Return the index if the target is found

        }

    }

    return -1;  // Return -1 if the target is not found in the array

}

int main() {

   int arr\_size;

    printf("Enter the size of the array: ");

    scanf("%d", &arr\_size);

    int arr[arr\_size];

    printf("Enter %d elements of the array:\n", arr\_size);

    for (int i = 0; i < arr\_size; i++) {

        scanf("%d", &arr[i]);

    }

    int target;

    printf("Enter the target element to search: ");

    scanf("%d", &target);

    int result = linearSearch(arr, arr\_size, target);

    if (result != -1) {

        printf("Element %d found at index %d\n", target, result);

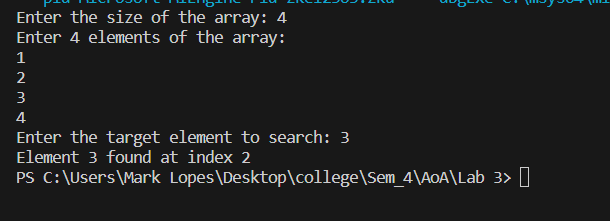
    } else {

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

    }

    return 0;

}

****

**Binary search:-**

#include <stdio.h>

int binarySearch(int arr[], int n, int target) {

    int low = 0;

    int high = n - 1;

    while (low <= high) {

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

        // If the target is found at the middle return

        if (arr[mid] == target)

            return mid;

        // If the target is greater than mid, go to the right half of array

        else if (arr[mid] < target)

            low = mid + 1;

        // If the target is smaller than mid, go to the left half of the array

        else

            high = mid - 1;

    }

    // Target not found

    return -1;

}

int main() {

    int n;

    printf("Enter the size of the array: ");

    scanf("%d", &n);

    int arr[n];

    printf("Enter the elements of the array in sorted order: ");

    for (int i = 0; i < n; i++)

        scanf("%d", &arr[i]);

    int target;

    printf("Enter the target value to search: ");

    scanf("%d", &target);

    int result = binarySearch(arr, n, target);

    if (result != -1)

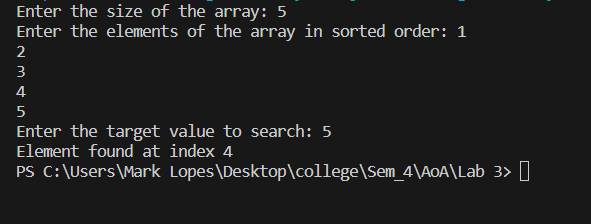
        printf("Element found at index %d\n", result);

    else

        printf("Element not found in the array\n");

    return 0;

}

****

**Binary recursion:-**

#include<stdio.h>

int binary\_rec(int a[], int low, int high, int x) {

    if (low > high)

        return 0;  // Element not found

    else {

        int mid = (low + high) / 2;

        if (a[mid] == x)

            return mid + 1;  // Element found, returning position (index + 1)

        else if (a[mid] > x)

            return binary\_rec(a, low, mid - 1, x);

        else

            return binary\_rec(a, mid + 1, high, x);

    }

}

int main() {

    int n, x, answer, low = 0;

    printf("Enter size of array:");

    scanf("%d", &n);

    int a[n];

    int high = n - 1;

    printf("Enter data in the array:");

    for (int i = 0; i < n; i++) {

        scanf("%d", &a[i]);

    }

    printf("Enter the data to be searched:");

    scanf("%d", &x);

    answer = binary\_rec(a, low, high, x);

    if (answer != 0) {

        printf("The data %d is at position %d\n", x, answer);

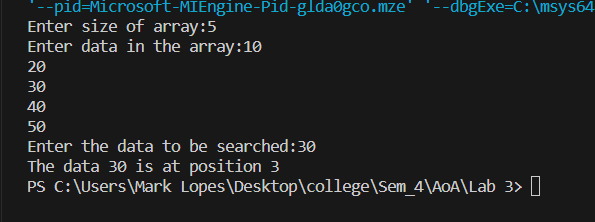
    } else {

        printf("The data %d is not present in the array\n", x);

    }

    return 0;

}

****

**Ternary:-**

#include<stdio.h>

int ternarySearch(int a[], int low, int high, int ele) {

    while(low <= high) {

     int mid1 = low + (high - low)/3;    //mid1 calculation

     int mid2 = high - (high - low)/3;    //mid2 calculation

     if(a[mid1] == ele)    //if ele == element at mid1

         return mid1;

     else if(a[mid2] == ele)     //if ele == element at mid2

         return mid2;

     else if(a[mid1] > ele)      //part 1 of array

         high = mid1-1;

     else if(ele > a[mid2])      //part 3

         low = mid2+1;

     else {              //part 2, middle part

         low = mid1+1;

         high = mid2-1;

     }

    }

    return -1; // if element not found

}

int main() {

    int n;

    printf("Enter the size of the array: ");

    scanf("%d", &n);

    int myNumbers[n];

    printf("Enter the elements of the array in sorted order: ");

    for (int i = 0; i < n; i++)

        scanf("%d", &myNumbers[i]);

    int target;

    printf("Enter the target value to search: ");

    scanf("%d", &target);

    int index = ternarySearch(myNumbers, 0, n - 1, target);

    if (index != -1)

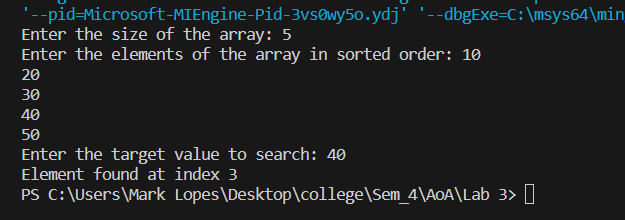
        printf("Element found at index %d\n", index);

    else

        printf("Element not found in the array\n");

    return 0;

}

****

**Ternary recursion:-**

#include<stdio.h>

int ternarySearchRec(int a[], int low, int high, int ele) {

    if (low > high)

        return -1; // If the element is not found

    int mid1 = low + (high - low) / 3;    //mid1 calculation

    int mid2 = high - (high - low) / 3;    //mid2 calculation

    if (a[mid1] == ele)    //if ele == element at mid1

        return mid1;

    else if (a[mid2] == ele)    //if ele == element at mid2

        return mid2;

    else if (a[mid1] > ele)    //part 1 of array

        return ternarySearchRec(a, low, mid1 - 1, ele);

    else if (ele > a[mid2])    //part 3

        return ternarySearchRec(a, mid2 + 1, high, ele);

    else    //part 2, middle part

        return ternarySearchRec(a, mid1 + 1, mid2 - 1, ele);

}

int main() {

    int n;

    printf("Enter the size of the array: ");

    scanf("%d", &n);

    int myNumbers[n];

    printf("Enter the elements of the array in sorted order: ");

    for (int i = 0; i < n; i++)

        scanf("%d", &myNumbers[i]);

    int target;

    printf("Enter the target value to search: ");

    scanf("%d", &target);

    int index = ternarySearchRec(myNumbers, 0, n - 1, target);

    if (index != -1)

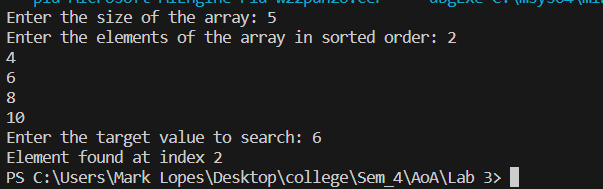
        printf("Element found at index %d\n", index);

    else

        printf("Element not found in the array\n");

    return 0;

}

****

**MinMax Brute force:-**

#include<stdio.h>

void findMaxMin(int a[], int n, int \*max, int \*min) {

    \*min = a[0];

    \*max = a[0];

    for (int i = 1; i < n; i++) {

        if (a[i] > \*max)

            \*max = a[i];

        if (a[i] < \*min)

            \*min = a[i];

    }

    printf("The max is %d and the min is %d\n", \*max, \*min);

}

int main() {

    int n;

    printf("Enter size of array:");

    scanf("%d", &n);

    int a[n];

    printf("Enter data in the array:");

    for (int i = 0; i < n; i++) {

        scanf("%d", &a[i]);

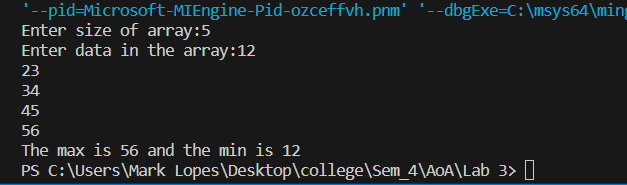
    }

    int max, min;

    findMaxMin(a, n, &max, &min);

    return 0;

}

****

**MinMax recursion:-**

#include <stdio.h>

void findMaxMin(int a[], int low, int high, int \*max, int \*min) {

    int mid;

    if (low == high) {

        \*min = a[low];

        \*max = a[low];

    } else if (low + 1 == high) {

        if (a[low] < a[high]) {

            \*min = a[low];

            \*max = a[high];

        } else {

            \*min = a[high];

            \*max = a[low];

        }

    } else {

        mid = (low + high) / 2;

        int leftMax, leftMin, rightMax, rightMin;

        findMaxMin(a, low, mid, &leftMax, &leftMin);

        findMaxMin(a, mid + 1, high, &rightMax, &rightMin);

        // Combine results from left and right subarrays

        if (leftMax > rightMax)

            \*max = leftMax;

        else

            \*max = rightMax;

        if (leftMin < rightMin)

            \*min = leftMin;

        else

            \*min = rightMin;

    }

}

int main() {

    int n;

    printf("Enter size of array:");

    scanf("%d", &n);

    int a[n];

    printf("Enter data in the array:");

    for (int i = 0; i < n; i++) {

        scanf("%d", &a[i]);

    }

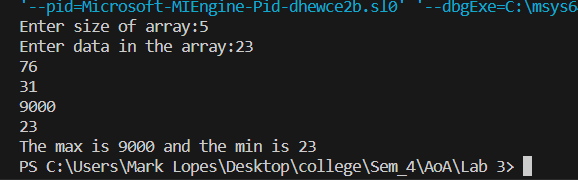
    int max, min;

    findMaxMin(a, 0, n - 1, &max, &min);

    printf("The max is %d and the min is %d\n", max, min);

    return 0;

}

****

