**Data Structures**

**Lab 1**

**Q.1) Write a program to find max and min elements in a 1-D array.**

**Ans. *Program:***

#include<stdio.h>

#include<time.h>

int main(){

    int size;

    printf("Enter the size of an Array: ");

    scanf("%d", &size);

    int arr[size];

    printf("Enter the elements of an Array: ");

    for(int i=0; i<size; i++){

        scanf("%d", &arr[i]);

    }

    clock\_t start = clock();

    int min=arr[0], max=arr[0];

    for(int i=0; i<size; i++){

        if(arr[i]<min){

            min=arr[i];

        }

        if(arr[i]>max){

            max=arr[i];

        }

    }

    clock\_t end = clock();

    double time = (double)(end-start)/CLOCKS\_PER\_SEC;

    printf("Minimum element of an Array is: %d\n", min);

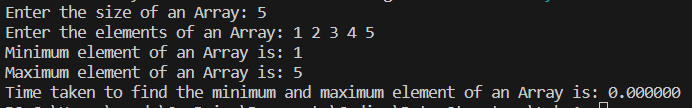
    printf("Maximum element of an Array is: %d\n", max);

    printf("Time taken to find the minimum and maximum element of an Array is: %lf\n", time);

    return 0;

}

*Output:*



**Q.2) Write a program to create a 1-D Integer Array using dynamic memory allocation. Enter the values of Array elements using the keyboard. Perform the following operations on it:**

**a) Traverse the Array from first to last.**

**Ans. *Program:***

#include<stdio.h>

#include<time.h>

#include<stdlib.h>

int main() {

    int size;

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

    scanf("%d", &size);

    clock\_t start = clock();

    int \*arr = (int\*)malloc(size \* sizeof(int));

    printf("Enter the elements of the array:\n");

    for (int i = 0; i < size; i++) {

        scanf("%d", &arr[i]);

    }

    printf("Traversing the array from first to last:\n");

    for (int i = 0; i < size; i++) {

        printf("%d ", arr[i]);

    }

    clock\_t end = clock();

    double time = (double)(end - start) / CLOCKS\_PER\_SEC;

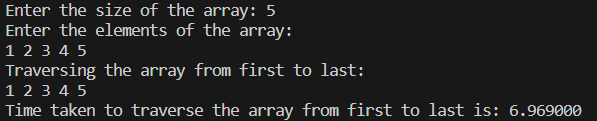
    printf("\nTime taken to traverse the array from first to last is: %lf\n", time);

    free(arr);

    return 0;

}

*Output:*



**b) Traverse the Array from last to first.**

**Ans. *Program:***

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

int main() {

    int n;

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

    scanf("%d", &n);

    int \*arr = (int\*)malloc(n \* sizeof(int));

    printf("Enter the elements of the array:\n");

    for (int i = 0; i < n; i++) {

        scanf("%d", &arr[i]);

    }

    clock\_t start = clock();

    printf("Traversing the array from last to first:\n");

    for (int i = n - 1; i >= 0; i--) {

        printf("%d ", arr[i]);

    }

    free(arr);

    clock\_t end = clock();

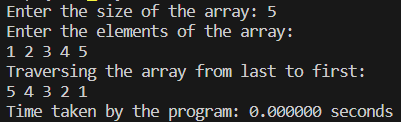
    double time\_taken = (double)(end - start) / CLOCKS\_PER\_SEC;

    printf("\nTime taken by the program: %f seconds\n", time\_taken);

    return 0;

}

*Output:*

**

**c) Search a particular number in the Array.**

**Ans. Program:**

#include <stdio.h>

#include <stdlib.h>

#include <time.h>

int main() {

    int n, i, search, flag = 0;

    int \*arr;

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

    scanf("%d", &n);

    arr = (int \*)malloc(n \* sizeof(int));

    printf("Enter the elements of the array:\n");

    for (i = 0; i < n; i++) {

        scanf("%d", &arr[i]);

    }

    clock\_t start = clock();

    printf("Enter the number to search: ");

    scanf("%d", &search);

    for (i = 0; i < n; i++) {

        if (arr[i] == search) {

            flag = 1;

            break;

        }

    }

    if (flag){

        printf("Number %d found at index %d\n", search, i);

    }

    else{

        printf("Number %d not found in the array\n", search);

    }

    free(arr);

    clock\_t end = clock();

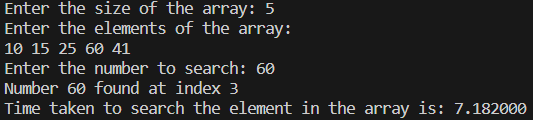
    double time = (double)(end - start) / CLOCKS\_PER\_SEC;

    printf("Time taken to search the element in the array is: %lf\n", time);

    return 0;

}

Output:



**Q.3) Write a program to create a 1-D Integer array. Perform the following operations on it:**

**a) Insert an element at a given position**

**Ans. Program:**

#include <stdio.h>

#include<time.h>

#define MAX\_SIZE 100

int main() {

    int arr[MAX\_SIZE];

    int size, position, element;

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

    scanf("%d", &size);

    printf("Enter the elements of the array:\n");

    for (int i = 0; i < size; i++) {

        scanf("%d", &arr[i]);

    }

    clock\_t start = clock();

    printf("Enter the position to insert the element: ");

    scanf("%d", &position);

    if (position < 0 || position > size) {

        printf("Invalid position!\n");

        return 0;

    }

    printf("Enter the element to insert: ");

    scanf("%d", &element);

    // Shift elements to the right from position to end

    for (int i = size - 1; i >= position; i--) {

        arr[i + 1] = arr[i];

    }

    // Insert the element at the given position

    arr[position] = element;

    size++; // Increase the size of the array

    printf("Array after inserting the element:\n");

    for (int i = 0; i < size; i++) {

        printf("%d ", arr[i]);

    }

    printf("\n");

    clock\_t end = clock();

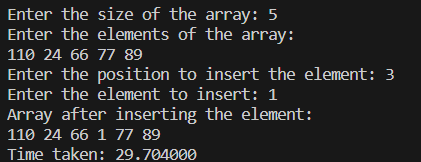
    double time\_taken = ((double)end - start) / CLOCKS\_PER\_SEC;

    printf("Time taken: %f\n", time\_taken);

    return 0;

}

Output:



**b) Delete an element present at a given position.**

**Ans. Program:**

#include <stdio.h>

#include<time.h>

#define MAX\_SIZE 100

int main() {

    int arr[MAX\_SIZE], size, position, i;

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

    scanf("%d", &size);

    printf("Enter the elements of the array:\n");

    for (i = 0; i < size; i++) {

        scanf("%d", &arr[i]);

    }

    clock\_t start = clock();

    printf("Enter the position of the element to be deleted: ");

    scanf("%d", &position);

    // Check if the position is valid

    if (position < 1 || position > size) {

        printf("Invalid position!\n");

    } else {

        // Shift the elements to the left from the given position

        for (i = position - 1; i < size - 1; i++) {

            arr[i] = arr[i + 1];

        }

        // Decrease the size of the array

        size--;

        // Print the updated array

        printf("Array after deleting the element:\n");

        for (i = 0; i < size; i++) {

            printf("%d ", arr[i]);

        }

        printf("\n");

    }

    clock\_t end = clock();

    double time\_spent = (double)(end - start) / CLOCKS\_PER\_SEC;

    printf("Time taken: %f\n", time\_spent);

    return 0;

}

Output:

