Assignment – 1

1. **Write a program to find the maximum element in an array.**

#include <stdio.h>

#include <stdlib.h>

int main()

{

    int n;

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

    scanf("%d", &n);

    int arr[n];

    if (n < 1)

    {

        printf("Invalid input\n");

        exit(0);

    }

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

    for (int i = 0; i < n; i++)

    {

        scanf("%d", &arr[i]);

    }

    int max = arr[0];

    for (int i = 1; i < n; i++)

    {

        if (arr[i] > max)

        {

            max = arr[i];

        }

    }

    printf("The maximum element in the array is: %d\n", max);

    return 0;

}

Source Code

Set 1

Enter the number of elements in the array: 0

Invalid input

Set 2

Enter the number of elements in the array: 5

Enter the elements of the array: 2 1 9 7 3

The maximum element in the array is: 9

Ouput

1. **Implement a function to revere an array in place.**

#include <stdio.h>

#include <stdlib.h>

#define max 10

void reverse(int arr[], int n);

int main()

{

    int arr[max], num, pos;

    printf("Enter how many elements you want: ");

    scanf("%d", &num);

    if (num < 1)

    {

        printf("Invalid input\n");

        exit(0);

    }

    printf("Enter the array elements: ");

    for (int i = 0; i < num; ++i)

    {

        scanf("%d", &arr[i]);

    }

    printf("The array is:");

    for (int i = 0; i < num; ++i)

    {

        printf(" %d", arr[i]);

    }

    printf("\n");

    reverse(arr, num);

    printf("The reverse array is:");

    for (int i = 0; i < num; ++i)

    {

        printf(" %d", arr[i]);

    }

    printf("\n");

    return 0;

}

Source Code: main()

void reverse(int arr[], int n)

{

    int temp;

    for (int i = 0; i < n / 2; i++)

    {

        temp = arr[i];

        arr[i] = arr[n - i - 1];

        arr[n - i - 1] = temp;

    }

}

Source Code: reverse()

Set 1

Enter the number of elements in the array: 0

Invalid input

Set 2

Enter how many elements you want: 5

Enter the array elements: 1 2 3 4 5

The array is: 1 2 3 4 5

The reverse array is: 5 4 3 2 1

Ouput

1. **Implement a function to reverse an array in place.**

#include <stdio.h>

#include <stdlib.h>

#define max 10

int \*intersection(int arr1[], int arr2[], int n1, int n2, int \*size);

void main()

{

    int arr1[max], arr2[max], n1, n2, size = 0;

    printf("Enter the number of elements in the first array: ");

    scanf("%d", &n1);

    printf("Enter the elements in the first array: ");

    for (int i = 0; i < n1; i++)

        scanf("%d", &arr1[i]);

    printf("Enter the number of elements in the second array: ");

    scanf("%d", &n2);

    printf("Enter the elements in the second array: ");

    for (int i = 0; i < n2; i++)

        scanf("%d", &arr2[i]);

    int \*temp = intersection(arr1, arr2, n1, n2, &size);

    printf("The intersection of the two arrays is: ");

    for (int i = 0; i < size; i++)

        printf("%d ", temp[i]);

    printf("\n");

}

Source Code: main()

int \*intersection(int arr1[], int arr2[], int n1, int n2, int \*size)

{

    int \*temp = (int \*)malloc(max \* sizeof(int)), k = 0;

    if (n1 > n2)

        intersection(arr2, arr1, n2, n1, size);

    for (int i = 0; i < n1; i++)

        for (int j = 0; j < n2; j++)

            if (arr1[i] == arr2[j])

            {

                int found = 0;

                for (int l = 0; l < k; l++)

                {

                    if (temp[l] == arr1[i])

                    {

                        found = 1;

                        break;

                    }

                }

                if (!found)

                    temp[k++] = arr1[i];

                break;

            }

    \*size = k;

    return temp;

}

Source Code: \*intersection()

()

Enter the number of elements in the first array: 5

Enter the elements in the first array: 6 4 5 8 2

Enter the number of elements in the second array: 3

Enter the elements in the second array: 2 4 3

The intersection of the two arrays is: 4 2

Ouput

1. **Write an algorithm to rotate an array given number of positions.**

#include <stdio.h>

#define max 10

void rotate(int arr[], int n, int pos);

int main()

{

    int num, arr[max], pos;

    printf("Enter how many elements you want: ");

    scanf("%d", &num);

    printf("Enter the array elements: ");

    for (int i = 0; i < num; ++i)

        scanf("%d", &arr[i]);

    printf("The position of rotation: ");

    scanf("%d", &pos);

if (pos < 0)

    {

        printf("Invalid input\n");

        return 0;

    }

    printf("The array is:");

    for (int i = 0; i < num; ++i)

        printf(" %d", arr[i]);

    printf("\n");

    rotate(arr, num, pos);

    printf("The rotated array is:");

    for (int i = 0; i < num; ++i)

        printf(" %d", arr[i]);

    printf("\n");

    return 0;

}

Source Code: main()

void rotate(int arr[], int n, int pos)

{

    // Adjust position to be within bounds

    if (pos > n)

        pos = pos % n;

    // Create a temporary array to hold the rotated values

    int temp[max];

    for (int i = 0; i < n; i++)

        temp[(i + pos) % n] = arr[i];

    // Step 3: Copy back from temp to arr

    for (int i = 0; i < n; i++)

        arr[i] = temp[i];

}

Source Code: rotate()

()

Enter how many elements you want: 5

Enter the array elements: 1 2 3 4 5

The position of rotation: 3

The array is: 1 2 3 4 5

The rotated array is: 3 4 5 1 2

Ouput

1. **Implement an algorithm to find the missing number in an array of integers from 1 to N.**

#include <stdio.h>

#include <stdlib.h>

#define max 10

int main(int argc, char const \*argv[])

{

int arr[max], num, sum = 0, expected\_sum = 0;

printf("Enter how many elements you want (up to %d): ", max);

scanf("%d", &num);

if (num > max)

{

printf("Number exceeds maximum limit of %d.\n", max);

return 1; // Exit if the number exceeds the limit

}

printf("Enter the elements (from 1 to %d): ", num);

for (int i = 0; i < num; i++)

scanf("%d", &arr[i]);

for (int i = 0; i < num - 1; i++)

sum += arr[i]; // Sum of entered numbers

// Calculate the expected sum of the first 'num' natural numbers

expected\_sum = num \* (num + 1) / 2;

// Find the missing number

printf("The missing number is: %d\n", expected\_sum - sum);

return 0;

}

Source Code: main()

Enter how many elements you want (up to 10): 5

Enter the elements (from 1 to 5): 1 3 4 5 6

The missing number is: 2

Ouput

Assignment – 2

1. **Write a function to remove duplicates from sorted array.**

#include <stdio.h>

#define MAX 10

void remove\_duplicate(int \*arr, int \*num);

int main()

{

    int arr[MAX], num;

    printf("Enter how many elements you want (max %d): ", MAX);

    scanf("%d", &num);

    if (num <= 0 || num > MAX)

    {

        printf("\n\tInvalid input!\n\n");

        return 1; // Exit on invalid input

    }

    printf("Enter the array elements: ");

    for (int i = 0; i < num; i++)

    {

        scanf("%d", &arr[i]);

    }

    remove\_duplicate(arr, &num);

    printf("The array after remove duplicate is: ");

    for (int i = 0; i < num; i++)

    {

        printf(" %d", arr[i]);

    }

    printf("\n");

    return 0;

}

Source Code: main()

void remove\_duplicate(int \*arr, int \*num)

{

    int temp[MAX], k = 0; // Start k at 0

    for (int i = 0; i < \*num; i++)

    {

        int flag = 0;               // Reset flag for each element

        for (int j = 0; j < k; j++) // Check against current unique elements

        {

            if (arr[i] == temp[j])

            {

                flag = 1;

                break; // Break early if a duplicate is found

            }

        }

        if (!flag)

        {

            temp[k++] = arr[i]; // Only add if not a duplicate

        }

    }

    \*num = k; // Update the count of unique elements

    for (int i = 0; i < k; i++)

    {

        arr[i] = temp[i]; // Copy unique elements back to original array

    }

}

Source Code: remove\_duplicate()

()

Enter how many elements you want (max 10): 5

Enter the array elements: 1 2 2 8 8

The array after remove duplicate is: 1 2 8

Ouput

1. **Implement an algorithm to find the majority element in an array.**

#include <stdio.h>

#define MAX 10

int main(int argc, char const \*argv[])

{

int arr[MAX], num;

printf("How many elements you want: ");

scanf("%d", &num);

printf("Enter the array elements: ");

for (int i = 0; i < num; ++i) // For inserting array elements

scanf("%d", &arr[i]);

int foundMajority = 0; // Flag to track if a majority element is found

for (int i = 0; i < num; ++i) // Calculating the frequency of each element

{

int frequency = 0;

for (int j = 0; j < num; ++j)

{

if (arr[i] == arr[j])

{

frequency++;

}

}

if (frequency > num / 2)

{

printf("Majority element is: %d", arr[i]);

foundMajority = 1; // Set the flag to indicate a majority element is found

break;

}

}

if (!foundMajority) // Check if no majority element was found

printf("No majority element exists");

return 0;

}

Source Code: main()

How many elements you want: 5

Enter the array elements: 6 5 5 4 2

Majority element is: 5

Ouput

1. **Create a program to find the largest subarray with a sum less than or equal to a given value.**

#include <iostream>

#include <climits> // Use INT\_MIN from <climits>

using namespace std;

int main()

{

    int arr[] = {1, 2, 3, 4, 5}, len = INT\_MIN, givenSum = 6;

    int start = -1, end = -1; // Initialize start and end

    cout << "The array is: ";

    for (int i = 0; i < sizeof(arr) / sizeof(arr[0]); i++)

        cout << arr[i] << " ";

    cout << "\nThe given sum is: " << givenSum;

    for (int i = 0; i < sizeof(arr) / sizeof(arr[0]); i++)

    {

        int sum = 0;

        for (int j = i; j < sizeof(arr) / sizeof(arr[0]); j++)

        {

            // for (int k = i; k <= j; k++) // Change to <= to include arr[j]

            sum += arr[j];

            if (sum <= givenSum)

            {

                if (j - i + 1 > len) // Update length and start/end indices

                {

                    len = j - i + 1;

                    start = i;

                    end = j;

                }

            }

        }

    }

    if (start == -1 && end == -1)

        cout << "No subarray found with the given sum." << endl;

    else

    {

        cout << "\nThe longest subarray of sum " << givenSum << " is: [";

        for (int i = start; i <= end; i++)

        {

            cout << arr[i];

            if (i < end)

                cout << ", ";

        }

        cout << "]" << endl;

    }

    return 0;

}

Source Code: main()

The array is: 1 2 3 4 5

The given sum is: 6

The longest subarray of sum 6 is: [1, 2, 3]

Ouput