**WEEK #5**

**OBJECTIVES**

* To help the students in learning the concepts of arrays (1D and 2D) in C++.
* To help the students in implementing and handling 1D and 2D arrays with theρ help on various examples.
* To help the students in understanding various operations associated with theρ arrays.

**OUTCOMES**

After completing this, the students would be able to:

* Handle arrays (1D and 2D) in the programs.
* Developρ & implement simple real-life examples of arrays.

1# Write a C++ program to enter elements in the array and display the array elements.

#include <iostream>

using namespace std;

const int MAX\_SIZE = 100;

void print\_array(int arr[], int size){

cout << "[";

for (int i = 0; i < size; i++){

cout << arr[i] << " ";

}

cout << "]" << endl;

}

int main(){

int arr[MAX\_SIZE];

int n;

cout << "Enter the size of the array: ";

cin >> n;

cout << "Enter the elements of the array:" << endl;

for (int i = 0; i < n; i++){

cout << "-> ";

cin >> arr[i];

}

cout << "Actual Array: ";

print\_array(arr, n);

int positive[MAX\_SIZE];

int negative[MAX\_SIZE];

int even[MAX\_SIZE];

int odd[MAX\_SIZE];

int positiveCount = 0;

int negativeCount = 0;

int evenCount = 0;

int oddCount = 0;

for (int i = 0; i < n; i++){

if (arr[i] > 0){

positive[positiveCount] = arr[i];

positiveCount++;

} else if (arr[i] < 0){

negative[negativeCount] = arr[i];

negativeCount++;

}

if (arr[i] % 2 != 0){

odd[oddCount] = arr[i];

oddCount++;

} else {

even[evenCount] = arr[i];

evenCount++;

}

}

cout << "Number of Positive Numbers: " << positiveCount << endl;

cout << "Positive numbers: ";

print\_array(positive, positiveCount);

cout << "Number of Negative Numbers: " << negativeCount << endl;

cout << "Negative numbers: ";

print\_array(negative, negativeCount);

cout << "Number of Even Numbers: " << evenCount << endl;

cout << "Even numbers: ";

print\_array(even, evenCount);

cout << "Number of Odd Numbers: " << oddCount << endl;

cout << "Odd numbers: ";

print\_array(odd, oddCount);

}

2# Write a C++ program to find the sum of the all-array element.

#include <iostream>

using namespace std;

int sum\_array(int arr[], int size){

int sum = 0;

for (int i = 0; i < size; i++){

sum = arr[i] + sum;

}

cout << sum;

}

int print\_array(int arr[], int size){

cout << "[ ";

for (int i = 0; i < size; i++){

cout << arr[i] << " ";

}

cout << "]" << endl;

}

int main(){

int size;

int \*arr = new int[size];

cout << "Enter size for the array: ";

cin >> size;

for (int i = 0; i < size; i++){

cout << "-> ";

cin >> arr[i];

}

cout << "The array is ";

print\_array(arr, size);

cout << "The sum of the elements of the array is ";

sum\_array(arr, size);

}

3# Write a C++ program to find the length of the array.

#include <iostream>

using namespace std;

int length\_array(int arr[], int size){

int count = 0;

for (int i = 0; i < size; i++){

count++;

}

cout << count;

return count;

}

void print\_array(int arr[], int size){

cout << "[ ";

for (int i = 0; i < size; i++){

cout << arr[i] << " ";

}

cout << "]" << endl;

}

int main(){

int size;

cout << "Enter size for the array: ";

cin >> size;

int \*arr = new int[size]; // Allocate memory after getting the size

for (int i = 0; i < size; i++){

cout << "-> ";

cin >> arr[i];

}

cout << "The array is ";

print\_array(arr, size);

cout << "The length of the array is ";

length\_array(arr, size);

// Don't forget to release the dynamically allocated memory

delete[] arr;

}

4# Write a C++ program to find the second-largest integer in a list of integers.

#include <iostream>

using namespace std;

void print\_array(int arr[], int size){

cout << "[";

for (int i = 0; i < size; i++){

cout << arr[i] << " ";

}

cout << "]" << endl;

}

int second\_largest(int arr[], int size){

if (size < 2){

cout << "Array size is too small to find the second largest element." << endl;

return -1;

}

int max1 = arr[0];

int max2 = arr[1];

if (max1 < max2){

swap(max1, max2);

}

for (int i = 2; i < size; i++){

if (arr[i] > max1){

max2 = max1;

max1 = arr[i];

}else if (arr[i] > max2 && arr[i] != max1){

max2 = arr[i];

}

}

return max2;

}

int main(){

int size;

cout << "Enter size for the array: ";

cin >> size;

int \*arr = new int[size];

for (int i = 0; i < size; i++){

cout << "-> ";

cin >> arr[i];

}

cout << "The array is ";

print\_array(arr, size);

int secondLargest = second\_largest(arr, size);

if (secondLargest != -1){

cout << "Second Largest element in the array is " << secondLargest;

}

delete[] arr;

}

5# Write a C++ Program to reverse the position of the array element (Hint: First eminent to

the last element.)

#include <iostream>

using namespace std;

void print\_array(int arr[], int size){

cout << "[";

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

{

cout << arr[i] << " ";

}

cout << "]" << endl;

}

void reverse\_array(int arr[], int size){

int \*arr\_new = new int[size];

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

arr\_new[size - 1 - i] = arr[i];

}

print\_array(arr\_new, size);

delete[] arr\_new;

}

int main(){

int size;

cout << "Enter size for the array: ";

cin >> size;

int \*arr = new int[size]; // Allocate memory after getting the size

for (int i = 0; i < size; i++){

cout << "-> ";

cin >> arr[i];

}

cout << "The array is ";

print\_array(arr, size);

cout << "The reversed array is ";

reverse\_array(arr, size);

delete[] arr;

}

6# Write a C++ program to perform the following:

a. Addition of two matrices

b. Multiplication of two matrices.

#include <iostream>

using namespace std;

const int MAX\_SIZE = 100;

void inputMatrix(int matrix[][MAX\_SIZE], int rows, int cols, const char\* name){

cout << "Enter elements for " << name << " matrix:" << endl;

for (int i = 0; i < rows; i++){

for (int j = 0; j < cols; j++){

cout << name << "[" << i << "][" << j << "]: ";

cin >> matrix[i][j];

}

}

}

void displayMatrix(int matrix[][MAX\_SIZE], int rows, int cols){

for (int i = 0; i < rows; i++){

for (int j = 0; j < cols; j++){

cout << matrix[i][j] << " ";

}

cout << endl;

}

}

void addMatrices(int A[][MAX\_SIZE], int B[][MAX\_SIZE], int C[][MAX\_SIZE], int rows, int cols){

for (int i = 0; i < rows; i++){

for (int j = 0; j < cols; j++){

C[i][j] = A[i][j] + B[i][j];

}

}

}

void multiplyMatrices(int A[][MAX\_SIZE], int B[][MAX\_SIZE], int C[][MAX\_SIZE], int rowsA, int colsA, int colsB){

for (int i = 0; i < rowsA; i++){

for (int j = 0; j < colsB; j++){

C[i][j] = 0;

for (int k = 0; k < colsA; k++){

C[i][j] += A[i][k] \* B[k][j];

}

}

}

}

int main(){

int A[MAX\_SIZE][MAX\_SIZE], B[MAX\_SIZE][MAX\_SIZE], C[MAX\_SIZE][MAX\_SIZE];

int rowsA, colsA, rowsB, colsB;

cout << "Enter the number of rows and columns for matrix A:" << endl;

cin >> rowsA >> colsA;

inputMatrix(A, rowsA, colsA, "A");

cout << "Enter the number of rows and columns for matrix B:" << endl;

cin >> rowsB >> colsB;

inputMatrix(B, rowsB, colsB, "B");

if (colsA != rowsB){

cout << "Matrix multiplication is not possible. Number of columns in A must be equal to the number of rows in B." << endl;

return 1;

}

if (rowsA == rowsB && colsA == colsB){

addMatrices(A, B, C, rowsA, colsA);

cout << "Matrix A + Matrix B:" << endl;

displayMatrix(C, rowsA, colsA);

}else{

cout << "Matrix addition is not possible. Matrices must have the same dimensions." << endl;

}

if (colsA == rowsB){

multiplyMatrices(A, B, C, rowsA, colsA, colsB);

cout << "Matrix A \* Matrix B:" << endl;

displayMatrix(C, rowsA, colsB);

}else{

cout << "Matrix multiplication is not possible. Number of columns in A must be equal to the number of rows in B." << endl;

}

return 0;

}

7# Write a C++ program to count and display positive, negative, odd and even numbers in

an array.

#include <iostream>

using namespace std;

const int MAX\_SIZE = 100;

void print\_array(int arr[], int size){

cout << "[";

for (int i = 0; i < size; i++){

cout << arr[i] << " ";

}

cout << "]" << endl;

}

int main(){

int arr[MAX\_SIZE];

int n;

cout << "Enter the size of the array: ";

cin >> n;

cout << "Enter the elements of the array:" << endl;

for (int i = 0; i < n; i++){

cout << "-> ";

cin >> arr[i];

}

cout << "Actual Array: ";

print\_array(arr, n);

int positive[MAX\_SIZE];

int negative[MAX\_SIZE];

int even[MAX\_SIZE];

int odd[MAX\_SIZE];

int positiveCount = 0;

int negativeCount = 0;

int evenCount = 0;

int oddCount = 0;

for (int i = 0; i < n; i++){

if (arr[i] > 0){

positive[positiveCount] = arr[i];

positiveCount++;

} else if (arr[i] < 0){

negative[negativeCount] = arr[i];

negativeCount++;

}

if (arr[i] % 2 != 0){

odd[oddCount] = arr[i];

oddCount++;

} else {

even[evenCount] = arr[i];

evenCount++;

}

}

cout << "Number of Positive Numbers: " << positiveCount << endl;

cout << "Positive numbers: ";

print\_array(positive, positiveCount);

cout << "Number of Negative Numbers: " << negativeCount << endl;

cout << "Negative numbers: ";

print\_array(negative, negativeCount);

cout << "Number of Even Numbers: " << evenCount << endl;

cout << "Even numbers: ";

print\_array(even, evenCount);

cout << "Number of Odd Numbers: " << oddCount << endl;

cout << "Odd numbers: ";

print\_array(odd, oddCount);

}

8# Write a C++ program to merge two sorted arrays into another array in sorted order.

#include <iostream>

using namespace std;

void sorted(int arr[], int size){

for(int i = 0; i < size; i++){

for (int j = i+1; j < size; j++){

if (arr[j] < arr[i]){

int temp = arr[i];

arr[i] = arr[j];

arr[j] = temp;

}

}

}

}

void mergeAndSort(int arr1[], int size1, int arr2[], int size2, int result[]){

int i = 0, j = 0, k = 0;

while (i < size1 && j < size2){

if (arr1[i] <= arr2[j]){

result[k] = arr1[i];

i++;

}else{

result[k] = arr2[j];

j++;

}

k++;

}

while (i < size1){

result[k] = arr1[i];

i++;

k++;

}

while (j < size2){

result[k] = arr2[j];

j++;

k++;

}

sorted(result, k);

}

void insertion(int arr[], int size){

for (int i = 0; i < size; i++){

cout << "-> ";

cin >> arr[i];

}

}

void print\_array(int arr[], int size){

cout << "[";

for (int i = 0; i < size; i++){

cout << arr[i];

if (i < size - 1)

cout << " ";

}

cout << "]" << endl;

}

int main(){

int size1, size2;

cout << "ARRAY 1" << endl;

cout << "Enter size for the array: ";

cin >> size1;

int \*arr1 = new int[size1];

insertion(arr1, size1);

cout << "Original Array: ";

print\_array(arr1, size1);

sorted(arr1, size1);

cout << "\nSorted Array: ";

print\_array(arr1, size1);

cout << "ARRAY 2" << endl;

cout << "Enter size for the array: ";

cin >> size2;

int \*arr2 = new int[size2];

insertion(arr2, size2);

cout << "Original Array: ";

print\_array(arr2, size2);

sorted(arr2, size2);

cout << "\nSorted Array: ";

print\_array(arr2, size2);

int mergedSize = size1 + size2;

int \*mergedArray = new int[mergedSize];

mergeAndSort(arr1, size1, arr2, size2, mergedArray);

cout << "MERGED & SORTED ARRAY " << endl;

cout << "Merged and Sorted Array: ";

print\_array(mergedArray, mergedSize);

}

9# Write a C++ program to find the frequency of a particular number in a list of integers.

#include <iostream>

using namespace std;

const int MAX\_SIZE = 100;

int main(){

int arr[MAX\_SIZE];

int n, target;

cout << "Enter the size of the array: ";

cin >> n;

cout << "Enter the elements of the array:" << endl;

for (int i = 0; i < n; i++){

cout << "--> ";

cin >> arr[i];

}

cout << "Enter the number to find its frequency: ";

cin >> target;

int frequency = 0;

for (int i = 0; i < n; i++){

if (arr[i] == target){

frequency++;

}

}

cout << "The frequency of " << target << " in the array is: " << frequency << endl;

return 0;

}