National University of Computer and Emerging Sciences



Name: Muhammad Suleman

Roll #: 22F-3350

Lab#: 01

Section: BCS-4B

# Problem 01:

#include <iostream>

using namespace std;

//function prototypes

int\* inputArray(int& size);

void swapArrays(int\*& arr1, int\*& arr2, int& size1, int& size2);

void displayArray(int\* arr, int size);

int main()

{

int size1 = 100;

int\* arr1 = inputArray(size1);

int size2 = 100;

int\* arr2 = inputArray(size2);

cout << "Before swapping:" << endl;

cout << "Array 1" << endl;

displayArray(arr1, size1);

cout << "Array 2" << endl;

displayArray(arr2, size2);

swapArrays(arr1, arr2, size1, size2);

cout << "After swapping:" << endl;

cout << "Array 1" << endl;

displayArray(arr1, size1);

cout << "Array 2" << endl;

displayArray(arr2, size2);

delete[] arr1;

delete[] arr2;

system("pause");

return 0;

}

// takes input form user

int\* inputArray(int& size)

{

int\* arr = new int[size];

int i = 0;

int value;

cout << "Enter elements for array (enter -1 to stop): ";

cin >> value;

while (value != -1)

{

arr[i++] = value;

cin >> value;

}

size = i;

return arr;

}

// swap array data

void swapArrays(int\*& arr1, int\*& arr2, int& size1, int& size2)

{

int\* temp = arr1;

arr1 = arr2;

arr2 = temp;

int tempSize = size1;

size1 = size2;

size2 = tempSize;

}

// outputs array

void displayArray(int\* arr, int size)

{

cout << "Array Data: ";

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

{

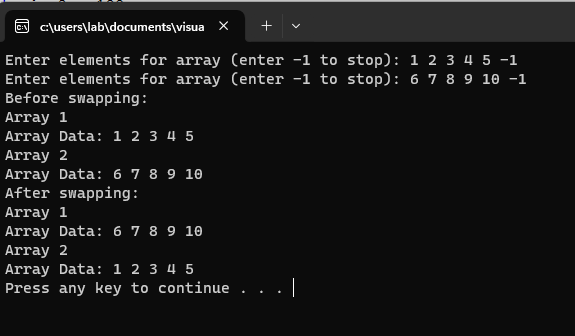
cout << arr[i] << " ";

}

cout << endl;

}

## Output:



# Problem 02:

#include <iostream>

using namespace std;

//function prototypes

int\*\* createMatrix();

void inputMatrix(int\*\* matrix);

void addition(int\*\* matrix1, int\*\* matrix2, int\*\* result);

void multiplication(int\*\* matrix1, int\*\* matrix2, int\*\* result);

void displayMatrix(int\*\* matrix);

void deallocateMemory(int \*\*matrix);

int main()

{

int\*\* matrix1 = createMatrix();

int\*\* matrix2 = createMatrix();

int\*\* resultMatrix = createMatrix();

cout << "Input Matrix 1" << endl;

inputMatrix(matrix1);

cout << "Input Matrix 2" << endl;

inputMatrix(matrix2);

addition(matrix1, matrix2, resultMatrix);

cout << "Result of Matrix Addition:" << endl;

displayMatrix(resultMatrix);

multiplication(matrix1, matrix2, resultMatrix);

cout << "Result of Matrix Multiplication:" << endl;

displayMatrix(resultMatrix);

deallocateMemory(matrix1);

deallocateMemory(matrix2);

deallocateMemory(resultMatrix);

system("pause");

return 0;

}

//alocatte memory

int\*\* createMatrix()

{

int\*\* matrix = new int\*[3];

for (int i = 0; i < 3; i++)

{

matrix[i] = new int[3];

}

return matrix;

}

//takes input from user

void inputMatrix(int\*\* matrix)

{

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

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < 3; j++)

{

cin >> matrix[i][j];

}

}

}

// perform addition

void addition(int\*\* matrix1, int\*\* matrix2, int\*\* result)

{

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < 3; j++)

{

result[i][j] = matrix1[i][j] + matrix2[i][j];

}

}

}

// perform multiplication

void multiplication(int\*\* matrix1, int\*\* matrix2, int\*\* result)

{

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < 3; j++)

{

result[i][j] = 0;

for (int k = 0; k < 3; k++)

{

result[i][j] += matrix1[i][k] \* matrix2[k][j];

}

}

}

}

// display 2D array in Matrix form

void displayMatrix(int\*\* matrix)

{

for (int i = 0; i < 3; i++)

{

for (int j = 0; j < 3; j++)

{

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

}

cout << endl;

}

}

// deallocates memory

void deallocateMemory(int \*\*matrix)

{

for (int i = 0; i < 3; i++)

{

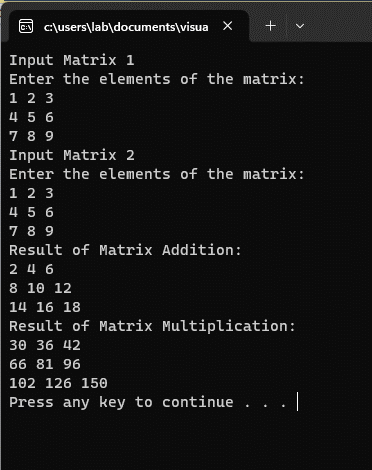
delete[] matrix[i];

}

delete[] matrix;

}

## Output:



# Problem 03:

#include <iostream>

using namespace std;

//function prototypes

int max(int \*arr, int size);

int min(int \*arr, int size);

int main()

{

int size;

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

cin >> size;

int \*arr = new int[size];

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

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

{

cin >> arr[i];

}

cout << "Maximum value: " << max(arr, size) << endl;

cout << "Minimum value: " << min(arr, size) << endl;

delete[] arr;

system("pause");

return 0;

}

//finding max

int max(int \*arr, int size)

{

int Max = INT\_MIN;

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

{

if (arr[i] > Max)

{

Max = arr[i];

}

}

return Max;

}

//finding min

int min(int \*arr, int size)

{

int Min = INT\_MAX;

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

{

if (arr[i] < Min)

{

Min = arr[i];

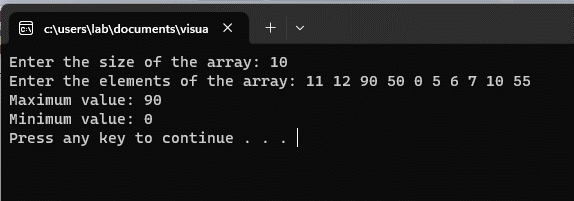
}

}

return Min;

}

## Output:



# Problem 04:

#include <iostream>

using namespace std;

// function prototypes

int maxByPointer(int \*arr, int size, int \*Max);

int minByPointer(int \*arr, int size, int \*Min);

int maxByReference(int \*arr, int size, int &Max);

int minByReference(int \*arr, int size, int &Min);

int main()

{

int size;

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

cin >> size;

int \*arr = new int[size];

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

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

{

cin >> arr[i];

}

// by pointer variables

int max\_val;

int min\_val;

int \*max = &max\_val;

int \*min = &min\_val;

cout << "Maximum value by Pointer: " << maxByPointer(arr, size, max) << endl;

cout << "Minimum value by Pointer: " << minByPointer(arr, size, min) << endl;

// by reference variables

int max1;

int min1;

cout << "Maximum value by reference: " << maxByReference(arr, size, max1) << endl;

cout << "Minimum value by reference: " << minByReference(arr, size, min1) << endl;

// deallocate memory

delete[] arr;

return 0;

}

// finding maximum by pointer

int maxByPointer(int \*arr, int size, int \*Max)

{

\*Max = INT\_MIN;

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

{

if (arr[i] > \*Max)

{

\*Max = arr[i];

}

}

return \*Max;

}

// finding minimum by pointer

int minByPointer(int \*arr, int size, int \*Min)

{

\*Min = INT\_MAX;

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

{

if (arr[i] < \*Min)

{

\*Min = arr[i];

}

}

return \*Min;

}

// finding maximum by reference

int maxByReference(int \*arr, int size, int &Max)

{

Max = INT\_MIN;

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

{

if (arr[i] > Max)

{

Max = arr[i];

}

}

return Max;

}

// finding minimum by reference

int minByReference(int \*arr, int size, int &Min)

{

Min = INT\_MAX;

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

{

if (arr[i] < Min)

{

Min = arr[i];

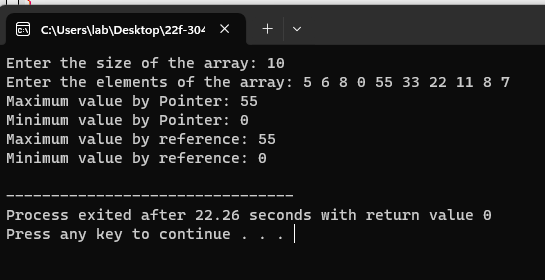
}

}

return Min;

}

## Output:



# Problem 05:

#include <iostream>

using namespace std;

//function prototypes

void bubbleSort(int\* arr, int size);

void displayArray(int\* arr, int size);

int main()

{

int size;

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

cin >> size;

int \*arr = new int[size];

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

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

{

cin >> arr[i];

}

cout << "Array before sorting: ";

displayArray(arr, size);

bubbleSort(arr, size);

cout << "Array after sorting: ";

displayArray(arr, size);

delete[] arr;

system("pause");

return 0;

}

//bubble sort algorithm

void bubbleSort(int\* arr, int size)

{

bool swapping;

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

{

swapping = false;

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

{

if (\*(arr + j) > \*(arr + j + 1))

{

int temp = \*(arr + j);

\*(arr + j) = \*(arr + j + 1);

\*(arr + j + 1) = temp;

swapping = true;

}

}

if (swapping == false)

{

break;

}

}

}

//function to display array

void displayArray(int\* arr, int size)

{

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

{

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

}

cout << endl;

}

## Output:

