//------------------------------------------------------------------

// File name: Exercise.cpp

// Assign ID:

// Due Date: 13/08/24 at 11pm

//

// Purpose: Multidimensional array and function.

//

// Author: Mr. KEO Sopahnit

//------------------------------------------------------------------

**Exercise\_1**

#include <iostream>

using namespace std;

int main()

{

// 1. Store

int number, rows, cols;

// 2. Input

cout << "Enter a number: ";

cin >> number;

cout << "Enter the number of rows: ";

cin >> rows;

cout << "Enter the number of columns: ";

cin >> cols;

// 3. Process

const int ROW = 10;

const int COL = 10;

int array[ROW][COL];

int current\_value = number;

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

{

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

{

array[i][j] = current\_value;

current\_value \*= 2;

}

}

// 4. Output

cout << "The created 2D array is:\n";

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

{

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

{

cout << array[i][j] << "\t";

}

cout << endl;

}

return 0;

}

**Exercise\_2**

#include <iostream>

using namespace std;

int main() {

//1. Store

int number, rows, cols;

//2. Input

cout << "Enter a number: ";

cin >> number;

cout << "Enter the number of rows: ";

cin >> rows;

cout << "Enter the number of columns: ";

cin >> cols;

//3. Process

const int ROW = 10;

const int COL = 10;

int array[ROW][COL];

int current\_value = number;

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

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

array[i][j] = current\_value;

current\_value += 1;

}

}

//4. Output

cout << "The created 2D array is:\n";

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

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

cout << array[i][j] << "\t";

}

cout << endl;

}

return 0;

}

**Exercise\_3**

#include <iostream>

#include <cstdlib>

#include <ctime>

using namespace std;

int main() {

//1. Store

int row, cols, shifts;

char direction;

//2. Input

cout << "Enter the number of row: ";

cin >> row;

cout << "Enter the number of columns: ";

cin >> cols;

//3. Process

// Initialize the 2D array

const int ROW = 20;

const int COL = 20;

int array[ROW][COL];

//Random numbers

srand(time(0));

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

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

array[i][j] = rand() % 10;

}

}

// Display Random array

cout << "Initial array:\n";

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

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

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

}

cout << endl;

}

cout << endl;

// The number of shifts and direction

cout << "Enter the number of shifts: ";

cin >> shifts;

cout << "Enter the direction (L for left, R for right, U for up, D for down): ";

cin >> direction;

switch (direction) {

case 'R':

case 'r':

// Shift right

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

for (int s = 0; s < shifts; s++) {

int temp = array[i][cols - 1];

for (int j = cols - 1; j > 0; j--) {

array[i][j] = array[i][j - 1];

}

array[i][0] = temp;

}

}

break;

case 'L':

case 'l':

// Shift left

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

for (int s = 0; s < shifts; s++) {

int temp = array[i][0];

for (int j = 0; j < cols - 1; j++) {

array[i][j] = array[i][j + 1];

}

array[i][cols - 1] = temp;

}

}

break;

case 'U':

case 'u':

// Shift up

for (int s = 0; s < shifts; s++) {

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

int temp = array[0][j];

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

array[i][j] = array[i + 1][j];

}

array[row - 1][j] = temp;

}

}

break;

case 'D':

case 'd':

// Shift down

for (int s = 0; s < shifts; s++) {

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

int temp = array[row - 1][j];

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

array[i][j] = array[i - 1][j];

}

array[0][j] = temp;

}

}

break;

default:

cout << "Invalid direction!" << endl;

return 1;

}

//4. Output

cout << "Array after shift:\n";

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

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

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

}

cout << endl;

}

return 0;

}

**Exercise\_4**

#include <iostream>

#include <cstdlib>

#include <ctime>

using namespace std;

int main() {

//1. Store

int rows, cols;

int sum = 0, minElement, maxElement;

double mean;

//2. Input

cout << "Enter the number of rows: ";

cin >> rows;

cout << "Enter the number of columns: ";

cin >> cols;

//3. Process

const int ROW = 20;

const int COL = 20;

int array[ROW][COL];

//Random array numbers

srand(time(0));

cout << "Random array elements:\n";

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

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

array[i][j] = rand() % 100;

sum += array[i][j];

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

if (i == 0 && j == 0) {

minElement = array[i][j];

maxElement = array[i][j];

}

if (array[i][j] < minElement) {

minElement = array[i][j];

}

if (array[i][j] > maxElement) {

maxElement = array[i][j];

}

}

cout << endl;

}

mean = static\_cast<double> (sum) / (rows \* cols);

// 4. Output the results

cout << "\nSum of all elements: \t\t" << sum << endl;

cout << "Arithmetic mean of all elements:\t" << mean << endl;

cout << "Minimum element: \t\t" << minElement << endl;

cout << "Maximum element: \t\t" << maxElement << endl;

return 0;

}

**Exercise\_5**

#include <iostream>

using namespace std;

int main() {

//1. Store

int rows=3, cols=4;

//2. Input

//3. Process

int const ROW = 10;

int const COL = 10;

int array[ROW][COL]={{3,5,6,7},{12,1,1,1},{0,7,12,1}};

//store the sum of rows and columns

int rowSum[ROW] = {0};

int colSum[COL] = {0};

int totalSum = 0;

// Calculate the sum

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

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

rowSum[i] += array[i][j];

colSum[j] += array[i][j];

totalSum += array[i][j];

}

}

//4. Output

//The row sums

cout << "\nArray with row sums:\n";

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

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

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

}

cout << "| " << rowSum[i] << endl;

}

//The column sums

cout << "---------------------\n";

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

cout << colSum[j] << " ";

}

cout << "| " << totalSum << endl;

return 0;

}

**Exercise\_6**

#include <iostream>

#include <cstdlib>

#include <ctime>

using namespace std;

int main() {

//1. Store

const int ROWS1 = 5;

const int COLS1 = 10;

const int ROWS2 = 5;

const int COLS2 = 5;

int array1[ROWS1][COLS1];

int array2[ROWS2][COLS2] = {0};

//2. Input (random)

//the Random Array

srand(time(0));

cout << "5x10 Array (filled with random numbers):\n";

for (int i = 0; i < ROWS1; i++) {

for (int j = 0; j < COLS1; j++) {

array1[i][j] = rand() % 51;

cout << array1[i][j] << "\t";

}

cout << endl;

}

//3. Process

for (int i = 0; i < ROWS2; i++) {

for (int j = 0; j < COLS2; j++) {

// Calculate the corresponding sum

int index1 = i \* 2;

int index2 = j \* 2;

if (index2 + 1 < COLS1) {

array2[i][j] = array1[i][index2] + array1[i][index2 + 1];

}

}

}

//4.Output

cout << "\n5x5 Array (sums of pairs from the 5x10 array):\n";

for (int i = 0; i < ROWS2; i++) {

for (int j = 0; j < COLS2; j++) {

cout << array2[i][j] << "\t";

}

cout << endl;

}

return 0;

}

//------------------------------------------------------------------

// File name: Exercise.cpp

// Assign ID:

// Due Date: 13/08/24 at 11pm

//

// Purpose: Function.

//

// Author: Mr. KEO Sopahnit

//------------------------------------------------------------------

**Exercise\_1**

#include <iostream>

using namespace std;

// Function to calculate the power of a number

double power(double base, int exponent) {

double result = 1.0;

//The exponent is negative

if (exponent < 0) {

base = 1 / base;

exponent = -exponent;

}

// Calculate exponent

while (exponent > 0) {

if (exponent % 2 == 1) {

result \*= base;

}

base \*= base;

exponent /= 2;

}

return result;

}

**Exercise\_2**

// Function to calculate the sum of numbers in a range between two integers

int sumInRange(int start, int end) {

// Swap if start is greater than end

if (start > end) {

int temp = start;

start = end;

end = temp;

}

int sum = 0;

// Calculate the sum

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

sum += i;

}

return sum;

}

**Exercise\_3**

// Function to check if a number is a perfect number

bool isPerfectNumber(int number) {

if (number <= 0) return false;

int sum = 0;

// Find all divisors and sum them up

for (int i = 1; i <= number / 2; ++i) {

if (number % i == 0) {

sum += i;

}

}

return sum == number;

}

// Function to find and display perfect numbers in a given range

void findPerfectNumbersInRange(int start, int end) {

cout << "Perfect numbers in the range [" << start << ", " << end << "] are:\n";

for (int num = start; num <= end; ++num) {

if (isPerfectNumber(num)) {

cout << num << " ";

}

}

cout << endl;

}

**Exercise\_4**

void displayCard(const string& rank, const string& suit) {

// Define the card suits and their symbols

const string suits[] = {"Hearts", "Diamonds", "Clubs", "Spades"};

const char suitSymbols[] = {'♥', '♦', '♣', '♠'};

// Find the suit symbol

char suitSymbol = ' ';

for (int i = 0; i < 4; ++i) {

if (suit == suits[i]) {

suitSymbol = suitSymbols[i];

break;

}

}

if (suitSymbol == ' ') {

cout << "Invalid suit." << endl;

return;

}

// Print the card

cout << "+------+" << endl;

cout << "|" << rank << " |" << endl;

cout << "| |" << endl;

cout << "| " << suitSymbol << " |" << endl;

cout << "| |" << endl;

cout << "| " << rank << "|" << endl;

cout << "+------+" << endl;

}

**Exercise\_5**

// Function that determines whether a six-digit number is "a lucky number" or not.

bool isLuckyNumber(int number) {

if (number < 100000 || number > 999999) {

return false; // Not a six-digit number

}

bool digits[10] = {false}; // Array to track digit occurrences

while (number > 0) {

int digit = number % 10;

if (digits[digit]) {

return false; // Duplicate digit found

}

digits[digit] = true;

number /= 10;

}

return true;

}

**Exercise\_6**

// Function to determine if a year is a leap year

bool isLeapYear(int year) {

return (year % 4 == 0 && (year % 100 != 0 || year % 400 == 0));

}

// Function to calculate the number of days in a given year

int daysInMonth(int month, int year) {

switch (month) {

case 1: case 3: case 5: case 7: case 8: case 10: case 12:

return 31;

case 4: case 6: case 9: case 11:

return 30;

case 2:

return isLeapYear(year) ? 29 : 28;

default:

return 0; // Invalid month

}

}

// Function to calculate the number of days from 01/01/0000 to the given date

int daysFromStart(int day, int month, int year) {

int days = 0;

// Count days for all years up to the year before the given year

for (int y = 0; y < year; ++y) {

days += isLeapYear(y) ? 366 : 365;

}

// Count days for all months in the given year before the given month

for (int m = 1; m < month; ++m) {

days += daysInMonth(m, year);

}

// Add days for the given month

days += day;

return days;

}

// Function to calculate the difference in days between two dates

int daysBetweenDates(int day1, int month1, int year1, int day2, int month2, int year2) {

int days1 = daysFromStart(day1, month1, year1);

int days2 = daysFromStart(day2, month2, year2);

return abs(days2 - days1);

}

**Exercise\_7**

// Function to calculate the arithmetic mean of elements in an array

double calculateMean(const int array[], int size) {

if (size <= 0) {

cout << "Error: Size of the array must be greater than 0." << endl;

return 0.0; // Return 0.0 as an indication of error

}

int sum = 0;

// Calculate the sum of all elements

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

sum += array[i];

}

// Calculate the mean

double mean = static\_cast<double>(sum) / size;

return mean;

}

**Exercsie\_8**

// Function to count positive, negative, and zero elements in an array

void countElements(const int array[], int size, int &positiveCount, int &negativeCount, int &zeroCount) {

positiveCount = 0;

negativeCount = 0;

zeroCount = 0;

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

if (array[i] > 0) {

++positiveCount;

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

++negativeCount;

} else {

++zeroCount;

}

}

}