Задача 0. Дано послідовність натуральних числень . Використовуючи підпрограму, яка дозволяє встановити, чи є послідовність із чотирьох чисел арифметичною прогресією, знайти кількість послідовно розміщених четвірок чисел, які утворюють арифметичну прогресію.

// ConsoleApplication19.cpp : Defines the entry point for the console application.

//

#include "stdafx.h"

#include <iostream>

using namespace std;

bool isArProgression(int c1, int c2, int c3, int c4)

{

int difference = c2 - c1;

return (difference == c3 - c2) && (difference == c4 - c3);

}

int getArSeqCount(int \* arr, int length)

{

int seqCount = 0;

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

{

if (isArProgression(arr[i-3],arr[i-2],arr[i-1],arr[i]))

{

seqCount++;

}

}

return seqCount;

}

int main()

{

int a[6] = { 23,5,8,11,14,17 };

printf("Sequences count =%d \n",getArSeqCount(a,6));

system("pause");

return 0;

}

Задача 1. Дано чотири числа. Створити такі функції:

1) підрахувати суму;

2) підрахувати добуток;

3) знайти середнє арифметичне;

4) знайти кількість додатних;

5) знайти суму додатних.

// ConsoleApplication19.cpp : Defines the entry point for the console application.

//

#include "stdafx.h"

#include <iostream>

using namespace std;

double getSum(double c1, double c2, double c3, double c4)

{

return c1 + c2 + c3 + c4;

}

double getProduct(double c1, double c2, double c3, double c4)

{

return c1 \* c2 \* c3 \* c4;

}

double getAverage(double c1, double c2, double c3, double c4)

{

return getSum(c1,c2,c3,c4)/4;

}

int getPositiveCount(double c1, double c2, double c3, double c4)

{

int positiveCount = 0;

if (c1>0)

{

positiveCount++;

}

if (c2>0)

{

positiveCount++;

}

if (c3>0)

{

positiveCount++;

}

if (c4>0)

{

positiveCount++;

}

return positiveCount;

}

double getSumPositive(double c1, double c2, double c3, double c4)

{

double sumPositive= 0;

if (c1>0)

{

sumPositive+=c1;

}

if (c2>0)

{

sumPositive += c2;

}

if (c3>0)

{

sumPositive += c3;

}

if (c4>0)

{

sumPositive += c4;

}

return sumPositive;

}

int main()

{

double a = 9, b=23, c=-91, d=5;

printf("Sum =%f \n",getSum(a,b,c,d));

printf("Product =%f \n", getProduct(a, b, c, d));

printf("Positive count =%d \n", getPositiveCount(a, b, c, d));

printf("Sum positive =%f \n", getSumPositive(a, b, c, d));

system("pause");

return 0;

}

//=====================================================

Задача 2. Дано одновимірний масив. Створити такі функції:

1) знайти кількість додатних;

2) знайти кількість елементів, які більші за останній елемент;

3) підрахувати кількість парних елементів після першого непарного (індекс непарного знайти за також допомогою функції);

// ConsoleApplication19.cpp : Defines the entry point for the console application.

//

#include "stdafx.h"

#include <iostream>

using namespace std;

//1) знайти кількість додатних;

int getPositiveCount(int\* a, int length)

{

int positiveCount = 0;

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

{

if (a[i]>0)

{

positiveCount++;

}

}

return positiveCount;

}

//2) знайти кількість елементів, які більші за останній елемент;

int getCountGreaterThanLast(int\* a, int length)

{

int elementsCountGreaterThanLast = 0;

int lastEmlement = a[length - 1];

for (int i = length-2; i>=0; i--)

{

if (a[i]>lastEmlement)

{

elementsCountGreaterThanLast++;

}

}

return elementsCountGreaterThanLast;

}

//3) підрахувати кількість парних елементів після першого непарного(індекс непарного знайти за також допомогою функції);

//Знаходження індекса першого непарного

int getFirstOddElementIndex(int\* a, int length)

{

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

{

if (a[i] % 2 != 0)

return i;

}

return -1;

}

//Підрахунок кількості парних після першого непарного

int getEvenNumberCount(int \* a, int length)

{

int firstOddElementIndex = getFirstOddElementIndex(a, length);

if (firstOddElementIndex==-1)

{

return -1;

}

else

{

int evenCount = 0;

for (int i = firstOddElementIndex+1; i < length; i++)

{

if (a[i]%2==0)

{

evenCount++;

}

}

return evenCount;

}

}

int main()

{

int\* a = new int[7]{2,11,3,4,6,3,10};

printf("Positive count =%d",getPositiveCount(a,7));

printf("Element count that is greater than the last=%d", getCountGreaterThanLast(a, 7));

system("pause");

return 0;

}

Задача 3. Дано двовимірний масив. Створити такі функції:

1) знайти кількість нулів;

2) знайти кількість рядків, які не містять нулів (при цьому використати

додаткову функцію, яка дозволяє виявити чи є у рядку нуль).

// ConsoleApplication19.cpp : Defines the entry point for the console application.

//

#include "stdafx.h"

#include <iostream>

using namespace std;

int getZeroCount(int\*\* a, int rowsCount, int columnsCount)

{

int zeroCount = 0;

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

{

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

{

if (a[i][j]==0)

{

zeroCount++;

}

}

}

return zeroCount;

}

//Допоміжна функція, яка дозволяє встановти чи є у рядку 0

bool containsRowZero(int\*\* a, int columnsCount, int row)

{

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

{

if (a[row][j] == 0)

{

return true;

}

}

return false;

}

//Підрахунок кількості рядків, які не містять нуль

int getRowsCountWithoutZero(int\*\* a, int rowsCount, int columnsCount)

{

int rowsCountWithoutZero = 0;

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

{

if (containsRowZero(a, columnsCount, i) == false)

rowsCountWithoutZero++;

}

return rowsCountWithoutZero;

}

int main()

{

//1. Вводимо кількість рядків і стовпців

int rowsCount;

printf("Rows count =");

cin >> rowsCount;

int columnsCount;

printf("Columns count =");

cin >> columnsCount;

//2. Виділяємо пам"ять для двовимірного масиву

int\* \* a = new int\*[rowsCount]; //Виділення пам"яті для масиву покажчиків на рядки

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

{

a[i] = new int[columnsCount]; //Виділення пам"яті для рядка

}

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

printf("Zero count=%d\n",getZeroCount(a,rowsCount,columnsCount));

printf("Rows count without zero=%d\n", getRowsCountWithoutZero(a, rowsCount, columnsCount));

int\* a = new int[7]{2,11,3,4,6,3,10};

printf("Positive count =%d",getPositiveCount(a,7));

printf("Element count that is greater than the last=%d", getCountGreaterThanLast(a, 7));

system("pause");

return 0;

}

Задача 4. Використовуючи функцію для знаходження кількості днів у місяці (за номером місяця і роком (використати функцію, яка дозволяє перевірити чи є рік високосним). Знайти загальну кількість днів у заданому діапазоні місяців у межах року (наприклад, від 3-го місяця до 8-го місяця включно).

Задача 5. Використовуючи функцію для виведення назви дня тижня за номером вивести на екран назви днів, номери яких збережено у масиві.