|  |  |
| --- | --- |
| C:\Documents and Settings\admin\Рабочий стол\МИРЭА_ЭМБЛЕМА_приказ.JPG | |
| МИНОБРАНАУКИ РОССИИ | |
| Федеральное государственное бюджетное образовательное учреждение  высшего образования  «**МОСКОВСКИЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»**  **МИРЭА** | |
| Институт информационных технологий  Кафедра математического обеспечения и стандартизации информационных технологий | |
| **Отчет по лабораторной работе №2**  **Тема работы «МНОГОМЕРНЫЙ МАССИВ»** | |
| Выполнил: |  |
| Студент группы ИКБО-07-17  Отчет принял:  Преподаватель кафедры МОСИТ | Акжигитов Р. Р.  Миронов Антон Николаевич |
| Москва  2018 | |

Оглавление

[Задание 1 3](#_Toc510297868)

[Условия задания 3](#_Toc510297869)

[Декомпозиция 3](#_Toc510297870)

[Реализация функций 3](#_Toc510297871)

[Задание 2 3](#_Toc510297872)

[Условия задания 3](#_Toc510297873)

[Декомпозиция 3](#_Toc510297874)

[Реализация функций 3](#_Toc510297875)

[Задание 3 4](#_Toc510297876)

[Условия задания 4](#_Toc510297877)

[Декомпозиция 4](#_Toc510297878)

[Реализация функций 4](#_Toc510297879)

[Функции ввода\вывода массива\значений 4](#_Toc510297880)

[Кодирование алгоритма программы 5](#_Toc510297881)

[Вывод 9](#_Toc510297882)

# Задание 1

# Условия задания

В двумерном массиве храниться информация о количестве учащихся в каждом классе каждой параллели с первой по одиннадцатую. Классы параллели нумеруются буквами А, Б, В, Г (т.е. четыре класса в каждой параллели). Каждая строка массива хранит информацию о классах одной параллели: в первой строке по первой параллели, во второй по второй и т.д. Определить общее число учащихся в заданной параллели.

# Декомпозиция

1. Выбор параллели.
2. Прохождение по столбцу 2-мерного массива с суммированием.

# Реализация функций

// получить сумму (line)-ной строки 2-мерного массива

int getColSum(int\* arr, int n, int m, int line) {

int sum = 0;

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

sum += arr[line \* m + j];

return sum;

}

# Задание 2

# Условия задания

Дан двухмерный массив. Переформировать его таким образом, чтобы строки располагались по возрастанию их поэлементных сумм.

# Декомпозиция

1. Получить массив сумм строк 2-мерного массива.
2. Сортировка пузырьком по массиву сумм с одновременным перемещением строк 2-мерного массива

# Реализация функций

// получение массива сумм строк дин. 2-мерного массива

int\* getColsSum(int\*\* arr, int\* result, int n, int m) {

int sum = 0;

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

sum = 0;

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

sum += arr[i][j];

result[i] = sum;

}

return result;

}

// сортировка пузырьком сумм строк + строк 2-мерного массива

void sortRows(int\*\* arr, int\* det, int n, int m) {

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

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

if (det[j] > det[j + 1]) {

swap(det[j], det[j + 1]);

swap(arr[j], arr[j + 1]);

}

}

# Задание 3

# Условия задания

Дан квадратный двухмерный массив, в котором выделены четыре четверти, ограниченные главной и побочной диагоналями. Найти количество простых чисел в правой четверти массива.

# Декомпозиция

1. Получить массив правой четверти 2-мерного массива.
2. Найти в нем простые числа.

# Реализация функций

// получение массива правой четверти (ниже побочной, выше главной) 2-мерного массива

void createArrayOfRightQuater(int\* matrix, int n, int\*& result, int& length) {

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

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

if (i < j && i > n - 1 - j) {

result = (int\*)realloc(result, sizeof(int) \* ++length);

result[length - 1] = \*(matrix + i \* n + j);

}

}

// простое ли число

int isPrime(int number) {

if (number < 2) return 0;

for (int i = 2; i \* i <= number; i++)

if (!(number % i)) return 0;

return 1;

}

// заполнение массива простыми числами из массива чисел правой четверти

void createPrimeNumbersArray(int\* right, int len, int\*& prime, int& pLen) {

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

if (isPrime(right[i])) {

prime = (int\*)realloc(prime, sizeof(int) \* ++pLen);

prime[pLen - 1] = right[i];

}

}

# Функции ввода\вывода массива\значений

// создание статического 2-мерного массива для первого задания

void createStaticArray(int arr[nForms][nParallels], int n, int m, int minRange, int maxRange) {

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

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

arr[i][j] = minRange + rand() % (maxRange - minRange);

}

// создание динамического 2-мерного массива с помощью косвенного указателя

void createDynamicArray(int\*\* arr, int n, int m, int minRange, int maxRange) {

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

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

arr[i][j] = minRange + rand() % (maxRange - minRange);

}

// создание дин. 2-мерного массива, как одномерный дин. массив

void createDynamicArray(int\* arr, int n, int minRange, int maxRange) {

for (int i = 0; i < n \* n; i++)

\*(arr + i) = minRange + rand() % (maxRange - minRange);

}

// вывод 2-мерного массива, как одномерного

void printArray(int\* arr, int n, int m) {

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

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

cout << arr[i \* m + j] << " ";

cout << endl;

}

}

// вывод 2-мерного массива с помощью косвенного указателя

void printArray(int\*\* arr, int n, int m) {

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

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

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

cout << endl;

}

}

// вывод одномерного массива

void printArray(int\* arr, int length) {

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

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

cout << endl;

}

// ввод с проверкой значения (мин, макс, соответствие типу)

int input(int min, int max) {

string str;

int num = min - 1;

do {

cout << "Please enter an integer number [" << min << ".." << max << "]: ";

getline(cin, str);

try { num = stoi(str); }

catch (exception e) { cerr << "Error: " << e.what() << endl; }

} while (num < min || num > max);

return num;

}

# Кодирование алгоритма программы

// laba2.cpp : Defines the entry point for the console application.

//

#include "stdafx.h"

#include "iostream"

#include "stdlib.h"

#include "time.h"

#include "string"

using namespace std;

#define nForms 11

#define nParallels 4

// создание статического 2-мерного массива для первого задания

void createStaticArray(int arr[nForms][nParallels], int n, int m, int minRange, int maxRange) {

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

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

arr[i][j] = minRange + rand() % (maxRange - minRange);

}

// создание динамического 2-мерного массива с помощью косвенного указателя

void createDynamicArray(int\*\* arr, int n, int m, int minRange, int maxRange) {

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

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

arr[i][j] = minRange + rand() % (maxRange - minRange);

}

// создание дин. 2-мерного массива, как одномерный дин. массив

void createDynamicArray(int\* arr, int n, int minRange, int maxRange) {

for (int i = 0; i < n \* n; i++)

\*(arr + i) = minRange + rand() % (maxRange - minRange);

}

// вывод 2-мерного массива, как одномерного

void printArray(int\* arr, int n, int m) {

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

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

cout << arr[i \* m + j] << " ";

cout << endl;

}

}

// вывод 2-мерного массива с помощью косвенного указателя

void printArray(int\*\* arr, int n, int m) {

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

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

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

cout << endl;

}

}

// вывод одномерного массива

void printArray(int\* arr, int length) {

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

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

cout << endl;

}

// получить сумму (line)-ной строки 2-мерного массива

int getColSum(int\* arr, int n, int m, int line) {

int sum = 0;

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

sum += arr[line \* m + j];

return sum;

}

// ввод с проверкой значения (мин, макс, соответствие типу)

int input(int min, int max) {

string str;

int num = min - 1;

do {

cout << "Please enter an integer number [" << min << ".." << max << "]: ";

getline(cin, str);

try { num = stoi(str); }

catch (exception e) { cerr << "Error: " << e.what() << endl; }

} while (num < min || num > max);

return num;

}

// получение массива сумм строк дин. 2-мерного массива

int\* getColsSum(int\*\* arr, int\* result, int n, int m) {

int sum = 0;

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

sum = 0;

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

sum += arr[i][j];

result[i] = sum;

}

return result;

}

// сортировка пузырьком сумм строк + строк 2-мерного массива

void sortRows(int\*\* arr, int\* det, int n, int m) {

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

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

if (det[j] > det[j + 1]) {

swap(det[j], det[j + 1]);

swap(arr[j], arr[j + 1]);

}

}

// получение массива правой четверти (ниже побочной, выше главной) 2-мерного массива

void createArrayOfRightQuater(int\* matrix, int n, int\*& result, int& length) {

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

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

if (i < j && i > n - 1 - j) {

result = (int\*)realloc(result, sizeof(int) \* ++length);

result[length - 1] = \*(matrix + i \* n + j);

}

}

// простое ли число

int isPrime(int number) {

if (number < 2) return 0;

for (int i = 2; i \* i <= number; i++)

if (!(number % i)) return 0;

return 1;

}

// заполнение массива простыми числами из массива чисел правой четверти

void createPrimeNumbersArray(int\* right, int len, int\*& prime, int& pLen) {

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

if (isPrime(right[i])) {

prime = (int\*)realloc(prime, sizeof(int) \* ++pLen);

prime[pLen - 1] = right[i];

}

}

int main()

{

srand(time(NULL));

int choice = 0;

cout << "Lab.2 2dArrays Akzhigitov Radmir IKBO-07-17 (10.03.18)\n";

while (choice != 4) {

cout << "1) Forms task\n2) Transpose cols of arrays\n3) Right quater\n4) Exit\n";

choice = input(1, 4);

switch (choice) {

case 1 : {

int classes[nForms][nParallels];

createStaticArray(classes, nForms, nParallels, 1, 10);

printArray(\*classes, nForms, nParallels);

cout << "Input a number of form..." << endl;

int lineNo = input(1, 11);

cout << "Sum of " << lineNo << " parallels: "

<< getColSum(\*classes, nForms, nParallels, lineNo - 1) << endl;

system("pause");

break;

}

case 2 : {

int\*\* matrix;

int n, m;

cout << "Input lenghts:\n";

n = input(1, 30);

m = input(1, 30);

matrix = new int\*[n];

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

matrix[i] = new int[m];

createDynamicArray(matrix, n, m, 1, 9);

printArray(matrix, n, m);

int\* linesSum = new int[n];

getColsSum(matrix, linesSum, n, m);

cout << "\nSum of cols: ";

printArray(linesSum, n);

sortRows(matrix, linesSum, n, m);

cout << "\nSorted cols: ";

printArray(linesSum, n);

cout << "\nSorted matrix:\n";

printArray(matrix, n, m);

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

delete[] matrix[i];

delete[] matrix;

delete[] linesSum;

system("pause");

break;

}

case 3 : {

int\* arr;

int n;

cout << "Length:\n";

n = input(1, 30);

arr = new int[n \* n];

createDynamicArray(arr, n, 10, 99);

printArray(arr, n, n);

int\* rightNums = NULL; int length = 0;

int\* primeNums = NULL; int primeLength = 0;

createArrayOfRightQuater(arr, n, rightNums, length);

cout << "\nRight quater: ";

printArray(rightNums, length);

createPrimeNumbersArray(rightNums, length, primeNums, primeLength);

cout << "Prime numbers: ";

printArray(primeNums, primeLength);

cout << "Amount of prime numbers: " << primeLength << endl;

delete[] arr;

delete[] rightNums;

delete[] primeNums;

system("pause");

break;

}

case 4: { choice = 4; break; }

}

system("cls");

}

system("pause");

return 0;

}

# Вывод

Выполнив работу, я узнал, как работать с двумерными и многомерными массивами, существует статический и динамические 2-мерные массивы. Научился создавать и обращаться к элементам разными способами: через двумерный массива arr[i][j], через одномерный массив arr[j \* size + i], через арифметические операции указателей \*(arr + j \* size + i). Также понял, как передавать 2-мерный массив в функцию, как очищать память.