Task 50 (advance)

C/C++ Basic Syntax.
Pointers. Multidimensional Dynamic Arrays

LEARN. GROW. SUCCEED.

by Viktor Ivanchenko / ivanvikvik@gmail.com / Minsk

Task #50 (advance)

Базовый синтаксис языка C/C++. Основы использования указателей. Динамические многомерные массивы. Адресная арифметика

Цель работы

Провести работу над ошибками и практически закрепить работу с динамической памятью с использованием указателей и адресной арифметики в языке C/C++ на примере работы с динамическими многомерными массивами.

Требования

1) Для каждого задания в начале рекомендуется разработать блок-схему алгоритма решения.
2) Проект обязательно должен быть сразу реализован и сохранён под управление системой контроля версий *(VCS)* ***git*** и в последующем залит в централизованный репозиторий на облачном хостинг-сервисе ***GitHub***.
3) Все программы должны быть разбиты на отдельные функции. При выполнении задания необходимо по максимуму пытаться разрабатывать универсальный, масштабируемый, легко поддерживаемый и читаемый код.
4) В соответствующих компонентах бизнес-логики необходимо предусмотреть ***«защиту от дурака»*** *(fool-proof)*, т.е. прежде чем выполнять действия с данными нужно проверить, являются ли данные адекватными (непротиворечивыми).
5) Одномерные и многомерные структуры данных рекомендуется реализовывать на базе ***динамических C/C++ массивов***.
6) Также рекомендуется придерживаться ***Single Responsibility Principle***, ***SRP*** (принципа единственной ответственности) – постарайтесь вынести основную бизнес-логику задания в отдельную функцию или функции (т.е. архитектура приложения должна минимум состоять из нескольких функций).

7) Программа должна обязательно быть снабжена комментариями на английском языке, в которых необходимо указать краткое предназначение программы, её версию, ФИО разработчика, номер группы и дату разработки.

8) Исходный текст основного кода и демонстрационной программы рекомендуется также снабжать поясняющими краткими комментариями.

9) Если логически не подразумевается или в задании иного не указано, то входными и выходными данными являются целые числа.

10) Программа должна быть снабжена дружелюбным и интуитивно понятным интерфейсом для взаимодействия с пользователем.

11) Предусмотреть вывод на консоль удобочитаемого результата для пользователя. Рекомендуется для программного интерфейса использовать английский язык.

12) При проверки работоспособности приложения необходимо проверить все тестовые случаи.

13) Для предоставляемого решения задания также необходимо подсчитать его алгоритмическую сложность (***Big O Notation***) для всех типов измерений: худший, средний и лучший случаи.

14) При разработке программ придерживайтесь соглашений по написанию кода на языке C/C++ (C++ *Code-Convention)*.

Дополнительное задание

Сжатие матрицы *[The matrix compression]*. Дана математическая прямоугольная матрица размером N на M. Необходимо разработать функцию (или программу), которая уплотняет заданную матрицу, удаляя из нее строки и столбцы, заполненные нулями. Ниже приведен рекомендуемый вид экрана программы (данные, введенные пользователем, выделены полужирным) и тестовые данные. Первоначальные данные матрицы могут быть автоматически сгенерированы с помощью встроенного генератора псевдослучайных чисел.

Тест 01
Введите размерность матрицы (N и M): ***3 4***
Введите элементы матрицы:
1 2 0 3
2 1 0 1
2 7 0 1
Результирующая матрица после сжатия:
1 2 3
2 1 1
2 7 1

Test 02
Input the matrix dimension (N and M): ***2 3***
Enter the matrix elements:
1 0 2
3 1 0
The resulting matrix after compression:
1 0 2
3 1 0

Test 03
Input the matrix dimension (N and M): ***4 3***
Enter the matrix elements:
0 0 0
2 0 1
0 0 0
2 0 -1

The resulting matrix after compression:

2 1

2 -1

Test04

…

Best of LUCK with it, and remember to HAVE FUN while you're learning :)
Victor Ivanchenko