|  |  |
| --- | --- |
| Пермский национальный исследовательский политехнический университет  Кафедра информационные технологии и автоматизированные системы | |
|  |  |
|  |  |
| Лабораторная работа №10  “Динамические массивы"  Вариант №9 | |
|  |  |
|  | Выполнил:  Студент группы ИВТ-24-2б  А.С. Ваулин  Проверил:  Доцент кафедры ИТАС  О.А. Полякова |
|  |  |
| Пермь 2025 | |

# Постановка задачи

Написать программу, в которой создаются динамические массивы и выполнить их обработку в соответствии со своим вариантом.

Сформировать двумерный массив. Добавить в него столбец с заданным номером;

# Анализ задачи

Программа реализует работу с двумерным массивом, который представлен как массив указателей, где каждый указатель ссылается на одномерный массив элементов (строку). Основные этапы решения задачи включают:

1. **Создание массива**:
   * Выделение памяти под массив указателей с использованием оператора new.
   * Выделение памяти для каждой строки массива.
   * Возвращение указателя на созданный массив.
2. **Заполнение массива**:
   * Последовательное заполнение элементов массива числами от 1 до rows \* cols для наглядности результата.
   * Использование вложенных циклов для обхода всех элементов.
3. **Вывод массива**:
   * Отображение элементов массива в виде таблицы с использованием вложенных циклов.
   * Разделение элементов табуляцией для удобства чтения.
4. **Добавление столбца**:
   * Создание нового массива с увеличенным на 1 количеством столбцов.
   * Копирование элементов из исходного массива в новый:
     + Элементы до заданной позиции копируются без изменений.
     + Вставка нового столбца (например, заполненного нулями).
     + Элементы после позиции сдвигаются на одну позицию вправо.
   * Освобождение памяти исходного массива и обновление количества столбцов.
5. **Освобождение памяти**:
   * Поэтапное освобождение памяти:
     + Сначала освобождаются все строки массива.
     + Затем освобождается массив указателей.
6. **Взаимодействие с пользователем**:
   * Запрос у пользователя номера столбца для вставки.
   * Проверка корректности введённого значения (диапазон от 1 до текущего числа столбцов плюс 1).
   * Выполнение операции добавления столбца только при корректном вводе.

**Структура программы**

Программа состоит из следующих функций:

* createArray(int rows, int cols) — создаёт двумерный массив размером rows × cols.
* fillArray(int\*\* arr, int rows, int cols) — заполняет массив данными.
* printArray(int\*\* arr, int rows, int cols) — выводит массив на экран.
* deleteArray(int\*\* arr, int rows) — освобождает выделенную память.
* addColumn(int\*\* arr, int rows, int& cols, int k) — добавляет столбец в позицию k и возвращает новый массив.
* main() — основная функция, управляющая логикой программы.

**Входные и выходные данные**

* **Входные данные**:
  + Размеры массива: количество строк (rows) и столбцов (cols), заданные в коде (например, 3 × 3).
  + Номер столбца (k), вводимый пользователем.
* **Выходные данные**:
  + Исходный массив, отображённый в виде таблицы.
  + Сообщение об ошибке при некорректном вводе номера столбца.
  + Результирующий массив с добавленным столбцом при корректном вводе.

# Блок-схема

начало

int\*\* createArray(int rows, int cols)

#include <iostream>

#include <windows.h>

using namespace std;

int\*\* arr = new int\* [rows];

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

arr[i] = new int[cols];

return arr;

void fillArray(int\*\* arr, int rows, int cols)

int count = 1;

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

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

arr[i][j] = count++;

return;

void printArray(int\*\* arr, int rows, int cols)

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

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

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

cout << endl;

return;

void deleteArray(int\*\* arr, int rows)

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

delete[] arr[i];

delete[] arr;

return;

int\*\* addColumn(int\*\* arr, int rows, int& cols, int k)

int newCols = cols + 1;

int\*\* newArr = createArray(rows, newCols);

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

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

newArr[i][j] = arr[i][j];

newArr[i][k - 1] = 0;

for (int j = k; j < newCols; j++)

newArr[i][j] = arr[i][j - 1];

deleteArray(arr, rows);

cols = newCols;

return newArr;

int main()

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int rows = 3;

int cols = 3;

int\*\* arr = createArray(rows, cols);

fillArray(arr, rows, cols);

cout << "Исходный массив:" << endl;

printArray(arr, rows, cols);

int k;

cout << "Введите номер столбца для добавления (1-" << cols + 1 << "): ";

α

α

cin >> k;

if (static\_cast<unsigned int>(k - 1) >

static\_cast<unsigned int>(cols))

cout << "Недопустимый номер столбца." << endl;

arr = addColumn(arr, rows, cols, k);

cout << "Массив после добавления столбца:" << endl;

printArray(arr, rows, cols);

deleteArray(arr, rows);

return 0;

# Код программы

#include <iostream>

#include <windows.h>

using namespace std;

// Создание двумерного массива

int\*\* createArray(int rows, int cols)

{

int\*\* arr = new int\* [rows];

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

{

arr[i] = new int[cols];

}

return arr;

}

// Заполнение массива последовательными числами

void fillArray(int\*\* arr, int rows, int cols)

{

int count = 1;

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

{

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

{

arr[i][j] = count++;

}

}

}

// Вывод массива на экран

void printArray(int\*\* arr, int rows, int cols)

{

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

{

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

{

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

}

cout << endl;

}

}

// Освобождение памяти

void deleteArray(int\*\* arr, int rows)

{

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

{

delete[] arr[i];

}

delete[] arr;

}

// Добавление столбца с заданным номером

int\*\* addColumn(int\*\* arr, int rows, int& cols, int k)

{

int newCols = cols + 1;

int\*\* newArr = createArray(rows, newCols);

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

{

// Копируем элементы до позиции k

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

{

newArr[i][j] = arr[i][j];

}

// Заполняем новый столбец нулями

newArr[i][k - 1] = 0;

// Копируем оставшиеся элементы со сдвигом

for (int j = k; j < newCols; j++)

{

newArr[i][j] = arr[i][j - 1];

}

}

// Освобождаем память старого массива

deleteArray(arr, rows);

cols = newCols; // Обновляем количество столбцов

return newArr;

}

int main()

{

SetConsoleCP(1251);

SetConsoleOutputCP(1251);

int rows = 3;

int cols = 3;

int\*\* arr = createArray(rows, cols);

// Заполнение и вывод исходного массива

fillArray(arr, rows, cols);

cout << "Исходный массив:" << endl;

printArray(arr, rows, cols);

int k;

cout << "Введите номер столбца для добавления (1-" << cols + 1 << "): ";

cin >> k;

if (static\_cast<unsigned int>(k - 1) > static\_cast<unsigned int>(cols))

{

cout << "Недопустимый номер столбца." << endl;

}

else

{

arr = addColumn(arr, rows, cols, k);

cout << "Массив после добавления столбца:" << endl;

printArray(arr, rows, cols);

}

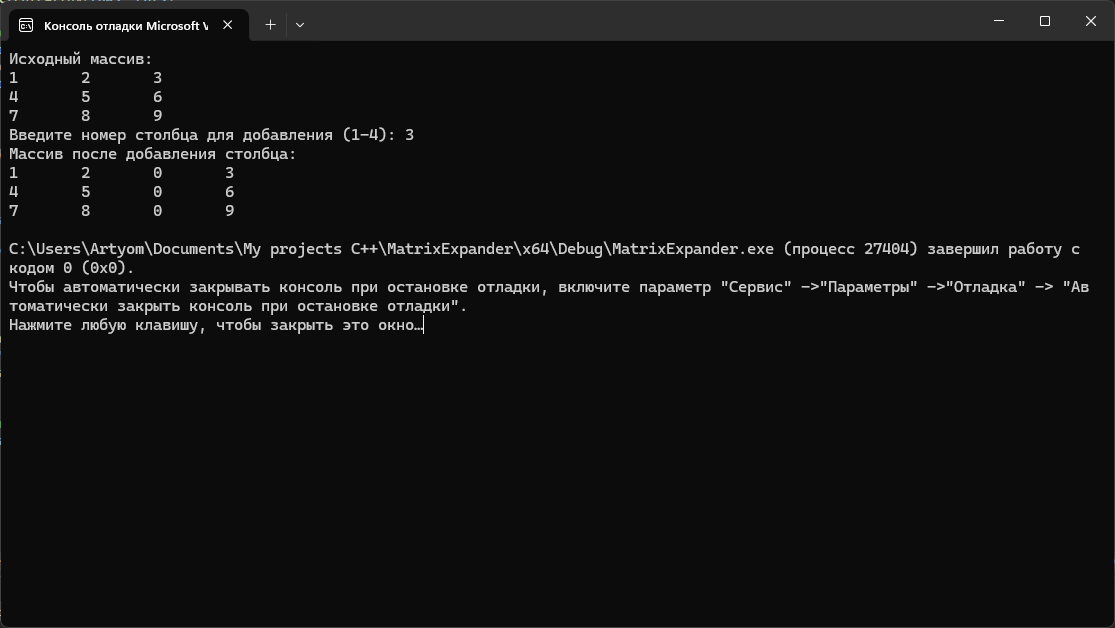
// Освобождение памяти

deleteArray(arr, rows);

return 0;

}

# Результат



Git: <https://github.com/Artivaa/MatrixExpander>