|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | |  |  |  | | МИНОБРНАУКИ РОССИИ | | | | Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** | | |   Институт Информационных технологий | |
|  | |
| Кафедра Математического обеспечения и стандартизации информационных технологий | |
|  | |
|  | |

|  |  |
| --- | --- |
| **ОТЧЕТ ПО ПРАКТИЧЕСКОЙ РАБОТЕ № 1** | |
| **по дисциплине** | |
| **«**Структуры и алгоритмы обработки данных**»**  **Тема: «Двумерный массив»** | |
|  | |
| Выполнил студент группы ИКБО-08-22 | Сенькевич Г. Д. |
| Принял преподаватель | Филатов А.С. |

|  |  |  |
| --- | --- | --- |
| Лабораторная работа выполнена | «\_\_»\_\_\_\_\_\_\_202\_\_ г. | *(подпись студента)* |
|  |  |  |
| «Зачтено» | «\_\_»\_\_\_\_\_\_\_202\_\_ г. | *(подпись руководителя)* |

Москва 2023

# **Цель работы**

# получение навыков по определению двумерного массива для структуры данных задачи

* получение навыков по разработке алгоритмов операций на двумерном массиве в соответствии с задачей

# Постановка задачи

1. Разработать программу по обработке данных, представленных в задаче матрицей и реализованной в программе двумерным (многомерным) статическим массивом.
   1. Максимально допустимые размеры статического массива установить через глобальные константы Rows, Cols.
   2. Если тип элемента массива не определен в варианте, определить массив для хранения значений целого типа.
   3. Разработать функции по реализации алгоритмов заполнения матрицы: с клавиатуры, датчиком случайных чисел. Разработать функции по реализации алгоритма вывода массива на экран построчно.
   4. Выполнить декомпозицию задачи варианта, разработать алгоритм решения. Реализовать функцию, выполняющую задачу и отладить ее.
   5. Разработать программу, демонстрирующую работу всех функций.
2. Разработать программу по обработке данных, представленных в задаче матрицей и реализованной в программе двумерным (многомерным) динамическим массивом.
   1. Размеры массива должны определяться пользователем.
   2. Двумерный массив определить как двойной указатель и выполнить его создание операцией new.
   3. Разработать функции по реализации алгоритмов заполнения матрицы: с клавиатуры, датчиком случайных чисел. Разработать функции по реализации алгоритма вывода массива на экран построчно.
   4. Выполнить декомпозицию задачи варианта, разработать алгоритм решения. Реализовать функцию, выполняющую задачу и отладить ее.
   5. Разработать программу, демонстрирующую работу всех функций.
3. Разработать программу по задаче варианта с реализацией данных задачи с применением контейнера vector библиотеки STL.
   1. Реализовать структуру хранения данных на основе шаблона <vector>, размеры определить при вводе с клавиатуры.
   2. Разработать функции: заполнение структуры хранения исходных данных, вывода структуры хранения.
   3. Выполнить декомпозицию задачи варианта, разработать алгоритм решения. Реализовать функцию, выполняющую задачу и отладить ее.
   4. Разработать программу, демонстрирующую работу всех функций.
4. Составить отчет, отобразив в нем описание выполнения всех этапов разработки, тестирования и код всей программы со скриншотами результатов тестирования.

Вариант №23. Условие задания:

|  |  |
| --- | --- |
| Упражнение 1 | Дана квадратная матрица. Найти определитель данной матрицы методом Гаусса. |
| Упражнение 2 | Текст вводится как последовательность символов, заканчивающаяся точкой. Определить количество различных пар букв в этом тесте. |

# **Решение**

Двумерный массив предназначен для представления структур данных элементы которых подчиняются бинарным (2-арным) отношениям. Каждый элемент структуры данных располагается в массиве в ячейке в соответствии с двумя параметрами (их еще называют координатами) - номер строки и номер столбца.  Например, для хранения матрицы размером 3×4 очень удобно использовать двумерный массив.

Имя двумерного массива – это указатель на массив указателей. Двумерный массив можно рассматривать как одномерный массив, элементы которого - одномерные массивы.

Метод Гаусса — это один из наиболее распространенных методов решения систем линейных уравнений. Он основан на представлении системы уравнений в виде матрицы и последовательном преобразовании этой матрицы к верхнетреугольному виду. Преобразования производятся с помощью элементарных операций над строками матрицы.

Элементарными операциями над строками матрицы называются следующие действия:

1. умножение одной строки на ненулевое число;
2. прибавление к одной строке другой, умноженной на некоторое число;
3. перестановка двух строк местами.

Применение метода Гаусса к матрице позволяет быстро вычислить ее определитель. Определитель матрицы равен произведению элементов главной диагонали матрицы после преобразования ее к верхнетреугольному виду.

Для решения первого и второго упражнений были написаны две функцим determinant - для статического и для динамического массивов, находящие определитель матрицы методом Гаусса. Функции принимают на вход массив (статический в одном случае и динамический во втором) и два целых числа – координаты левого верхнего угла подматрицы, определитель которой нужно найти (по умолчанию 0 и 0 – то есть матрица не усекается) и возвращает целое число – определитель.

|  |
| --- |
| // Функция нахождения определителя для статического двумерного  // массива  int determinant(int matrix[ROWS][COLUMNS], int from\_row=0, int from\_column=0)  {  if (ROWS - from\_row == 1 && COLUMNS - from\_column == 1)  {  return matrix[from\_row][from\_column];  }  for (int i = from\_row + 1; i < ROWS; i++)  {  for (int j = from\_column + 1; j < COLUMNS; j++)  {  matrix[i][j] \*= matrix[from\_row][from\_column];  matrix[i][j] -= matrix[i][from\_column] \* matrix[from\_row][j];  }  matrix[i][from\_column] = 0;  }  int det = determinant(matrix, from\_row + 1, from\_column + 1);  for (int i = from\_row + 2; i < ROWS; i++)  {  det /= matrix[from\_row][from\_column];  }  return det;  }  // Функция нахождения определителя для динамического двумерного  // массива  int determinant(int\*\* matrix, int rows, int columns,  int from\_row=0, int from\_column=0)  {  if (rows - from\_row == 1 && columns - from\_column == 1)  {  return matrix[from\_row][from\_column];  }  for (int i = from\_row + 1; i < rows; i++)  {  for (int j = from\_column + 1; j < columns; j++)  {  matrix[i][j] \*= matrix[from\_row][from\_column];  matrix[i][j] -= matrix[i][from\_column] \* matrix[from\_row][j];  }  matrix[i][from\_column] = 0;  }  int det = determinant(matrix, rows, columns,  from\_row + 1, from\_column + 1);  for (int i = from\_row + 2; i < rows; i++)  {  det /= matrix[from\_row][from\_column];  }  return det;  } |

Для решения третьего упражнения была написана функция count\_different\_pairs, находящая количество различных пар букв в тексте. Функции принимает на вход массив строку (std::string) и возвращает целое число – количество различных пар букв в ней.

|  |
| --- |
| // Функция нахождения количества различных пар букв в тексте  int count\_different\_pairs(std::string str)  {  const char\* letters = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ";  std::vector<std::vector<bool>> matrix (52, std::vector<bool>(52, false));  for (int i = 0; i < str.length() - 1; i++)  {  int first\_index = -1;  if (str[i] >= 'a' && str[i] <= 'z')  {  first\_index = str[i] - 'a';  }  else if (str[i] >= 'A' && str[i] <= 'Z')  {  first\_index = str[i] - 'A' + 26;  }  int second\_index = -1;  if (str[i + 1] >= 'a' && str[i + 1] <= 'z')  {  second\_index = str[i + 1] - 'a';  }  else if (str[i + 1] >= 'A' && str[i + 1] <= 'Z')  {  second\_index = str[i + 1] - 'A' + 26;  }  if (first\_index != -1 && second\_index != -1)  {  matrix[first\_index][second\_index] = true;  }  }  int count = 0;  for (int i = 0; i < matrix.size(); i++)  {  for (int j = 0; j < matrix[i].size(); j++)  {  if (matrix[i][j])  {  count++;  }  }  }  return count;  } |

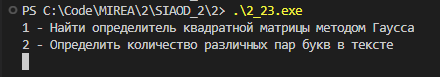
При запуске программы пользователь видит пользовательское меню, позволяющее выбрать между двумя функциями – вычислением определителя матрицы с помощью метода Гаусса и определением количества различных пар символов в тексте.

Рисунок 1. Интерфейс программы

# **Тестирование**

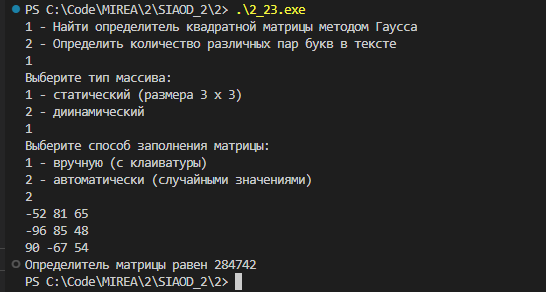
Протестируем выполнение программой первого упражнения. Для этого выберем в меню функцию вычисления определителя методом Гаусса, тип массива выберем статический, а ввод автоматический (случайными значениями. На рисунке 2 видим, как программа вывела верный ответ.

Рисунок 2. Тестирование программы

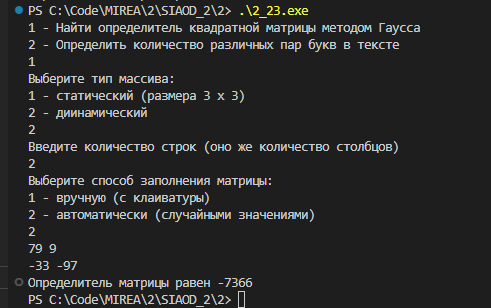
Протестируем выполнение программой второго упражнения. Для этого выберем в меню функцию вычисления определителя методом Гаусса, тип массива выберем динамический, размерность - 2, а ввод автоматический (случайными значениями. На рисунке 3 видим, как программа вывела верный ответ.

Рисунок 3. Тестирование программы

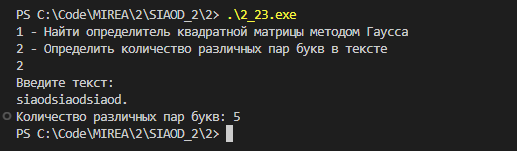
Протестируем выполнение программой третьего упражнения. Для этого выберем в меню функцию определения количества различных пар символов в тексте, и введём текст “siaodsiaodsiaod” – программа должна вывести число 5. На рисунке 4 видим, как программа вывела верный ответ.

Рисунок 4. Тестирование программы

Из результатов выполнения программы видно, что программа работает корректно, решая все поставленные задачи

# **Вывод**

В результате выполнения работы я:

1. Освоил алгоритмы работы двумерными массивами на языке программирования C++.

# **Исходный код программы**

|  |
| --- |
| #include <iostream>  #include <ctime>  #include <random>  #include <string>  #include <vector>  #define ROWS 3  #define COLUMNS 3  int rand\_int()  {  return (std::rand() % 201) - 100;  }  // Static  void print\_matrix(int matrix[ROWS][COLUMNS])  {  for (int i = 0; i < ROWS; i++)  {  for (int j = 0; j < COLUMNS; j++)  {  std::cout << matrix[i][j] << ' ';  }  std::cout << std::endl;  }  }  void read\_matrix(int matrix[ROWS][COLUMNS])  {  for (int i = 0; i < ROWS; i++)  {  for (int j = 0; j < COLUMNS; j++)  {  std::cin >> matrix[i][j];  }  }  }  void fill\_matrix\_random(int matrix[ROWS][COLUMNS])  {  for (int i = 0; i < ROWS; i++)  {  for (int j = 0; j < COLUMNS; j++)  {  matrix[i][j] = rand\_int();  }  }  }  int determinant(int matrix[ROWS][COLUMNS], int from\_row=0, int from\_column=0)  {  if (ROWS - from\_row == 1 && COLUMNS - from\_column == 1)  {  return matrix[from\_row][from\_column];  }  for (int i = from\_row + 1; i < ROWS; i++)  {  for (int j = from\_column + 1; j < COLUMNS; j++)  {  matrix[i][j] \*= matrix[from\_row][from\_column];  matrix[i][j] -= matrix[i][from\_column] \* matrix[from\_row][j];  }  matrix[i][from\_column] = 0;  }  int det = determinant(matrix, from\_row + 1, from\_column + 1);  for (int i = from\_row + 2; i < ROWS; i++)  {  det /= matrix[from\_row][from\_column];  }  return det;  }  // Dynamic  void print\_matrix(int\*\* matrix, int rows, int columns)  {  for (int i = 0; i < rows; i++)  {  for (int j = 0; j < columns; j++)  {  std::cout << matrix[i][j] << ' ';  }  std::cout << std::endl;  }  }  void read\_matrix(int\*\* matrix, int rows, int columns)  {  for (int i = 0; i < rows; i++)  {  for (int j = 0; j < columns; j++)  {  std::cin >> matrix[i][j];  }  }  }  void fill\_matrix\_random(int\*\* matrix, int rows, int columns)  {  for (int i = 0; i < rows; i++)  {  for (int j = 0; j < columns; j++)  {  matrix[i][j] = rand\_int();  }  }  }  int determinant(int\*\* matrix, int rows, int columns,  int from\_row=0, int from\_column=0)  {  if (rows - from\_row == 1 && columns - from\_column == 1)  {  return matrix[from\_row][from\_column];  }  for (int i = from\_row + 1; i < rows; i++)  {  for (int j = from\_column + 1; j < columns; j++)  {  matrix[i][j] \*= matrix[from\_row][from\_column];  matrix[i][j] -= matrix[i][from\_column] \* matrix[from\_row][j];  }  matrix[i][from\_column] = 0;  }  int det = determinant(matrix, rows, columns,  from\_row + 1, from\_column + 1);  for (int i = from\_row + 2; i < rows; i++)  {  det /= matrix[from\_row][from\_column];  }  return det;  }  // std::vector  int count\_different\_pairs(std::string str)  {  const char\* letters = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ";  std::vector<std::vector<bool>> matrix (52, std::vector<bool>(52, false));  for (int i = 0; i < str.length() - 1; i++)  {  int first\_index = -1;  if (str[i] >= 'a' && str[i] <= 'z')  {  first\_index = str[i] - 'a';  }  else if (str[i] >= 'A' && str[i] <= 'Z')  {  first\_index = str[i] - 'A' + 26;  }  int second\_index = -1;  if (str[i + 1] >= 'a' && str[i + 1] <= 'z')  {  second\_index = str[i + 1] - 'a';  }  else if (str[i + 1] >= 'A' && str[i + 1] <= 'Z')  {  second\_index = str[i + 1] - 'A' + 26;  }  if (first\_index != -1 && second\_index != -1)  {  matrix[first\_index][second\_index] = true;  }  }  int count = 0;  for (int i = 0; i < matrix.size(); i++)  {  for (int j = 0; j < matrix[i].size(); j++)  {  if (matrix[i][j])  {  count++;  }  }  }  return count;  }  int main()  {  std::srand(std::time(nullptr));  int mode = 0;  std::cout << "1 - Найти определитель квадратной матрицы методом Гаусса\n"  << "2 - Определить количество различных пар букв в тексте\n";  std::cin >> mode;    if (mode == 1)  {  int matrix[ROWS][COLUMNS];  int array\_type = 0;  std::cout << "Выберите тип массива:\n"  << "1 - статический (размера "  << ROWS << " x " << COLUMNS << ")\n"  << "2 - диинамический\n";  std::cin >> array\_type;  if (array\_type == 1)  {  int fill\_method = 0;  std::cout << "Выберите способ заполнения матрицы:\n"  << "1 - вручную (с клаиватуры)\n"  << "2 - автоматически (случайными значениями)\n";  std::cin >> fill\_method;    if (fill\_method == 1)  {  read\_matrix(matrix);  }  else if (fill\_method == 2)  {  fill\_matrix\_random(matrix);  print\_matrix(matrix);  }  else  {  std::cout << "Такого варианта нет!" << std::endl;  }  std::cout << "Определитель матрицы равен "  << determinant(matrix);  }  else if (array\_type == 2)  {  int rows = 0, columns = 0;  std::cout << "Введите количество строк (оно же количество столбцов)\n";  std::cin >> rows;  columns = rows;  int\*\* matrix = new int\*[rows];  for (int row = 0; row < rows; row++)  {  matrix[row] = new int[columns];  }  int fill\_method = 0;  std::cout << "Выберите способ заполнения матрицы:\n"  << "1 - вручную (с клаиватуры)\n"  << "2 - автоматически (случайными значениями)\n";  std::cin >> fill\_method;    if (fill\_method == 1)  {  read\_matrix(matrix, rows, columns);  }  else if (fill\_method == 2)  {  fill\_matrix\_random(matrix, rows, columns);  print\_matrix(matrix, rows, columns);  }  else  {  std::cout << "Такого варианта нет!" << std::endl;  }  std::cout << "Определитель матрицы равен "  << determinant(matrix, rows, columns);  for (int row = 0; row < rows; row++)  {  delete[] matrix[row];  }  delete[] matrix;  }  else  {  std::cout << "Такого варианта нет!" << std::endl;  }  }  else if (mode == 2)  {  std::string text;  std::cout << "Введите текст:\n";  char tmp = ' ';  while (tmp != '.')  {  tmp = std::cin.get();  text += tmp;  }  std::cout << "Количество различных пар букв: "  << count\_different\_pairs(text);  }  else  {  std::cout << "Такого варианта нет!" << std::endl;  }  return 0;  } |