Лабораторная работа № 7  
на тему  
**«Работа с одномерными массивами»**

# СОДЕРЖАНИЕ

СОДЕРЖАНИЕ 2

1 Задание 3

2 Основная часть 3

2.1 Блок-схемы 3

2.2 Исходный код программы 5

2.2.1 Библиотека 5

2.2.2 Основная программа 6

3 Заключение 8

# 1 Задание

Дан динамический целочисленный массив. Элементы массива могут принимать целые значения от 0 до 1000 включительно. Опишите алгоритм, который подсчитывает количество пар соседних элементов, разница между которыми не кратна 7. Гарантируется, что в массиве есть хотя бы один элемент, кратный 7.

Для выполнения данного задания необходимо создать новую DLL-библиотеку и разработать в ней следующие функции:

1. fillArrayRandomly(int\* array, size\_t length, int min, int max), которая заполняет все элементы массива случайными значениями от 0 до 1000 включительно, и гарантирует нахождение в массиве как минимум одного элемента, кратного 7;
2. fillWithArray(DataGridView^ grid, int\* array, size\_t length), которая заполняет DataGridView, выступающий в роли поля вывода, значениями массива;
3. solve(int\* array, size\_t length), которая находит количество пар соседних элементов массива, разница между которыми не кратна 7.

Также необходимо разработать событийную функцию button\_solve\_click(..), обрабатывающую нажатие кнопки «ЗАПОЛНИТЬ», которая вызывает вышеперечисленные функции и выводит необходимые результаты.

# 2 Основная часть

## 2.1 Блок-схемы

Блок-схемы, реализованные в тексте программы, указаны ниже.

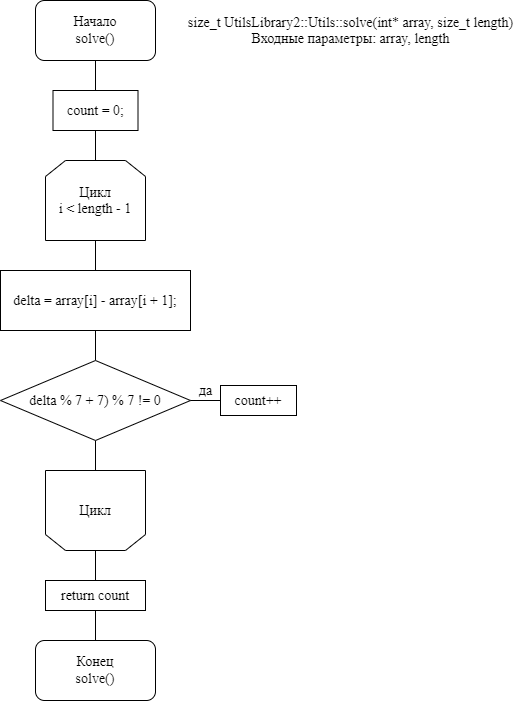


Рисунок 1 — Функция solve()

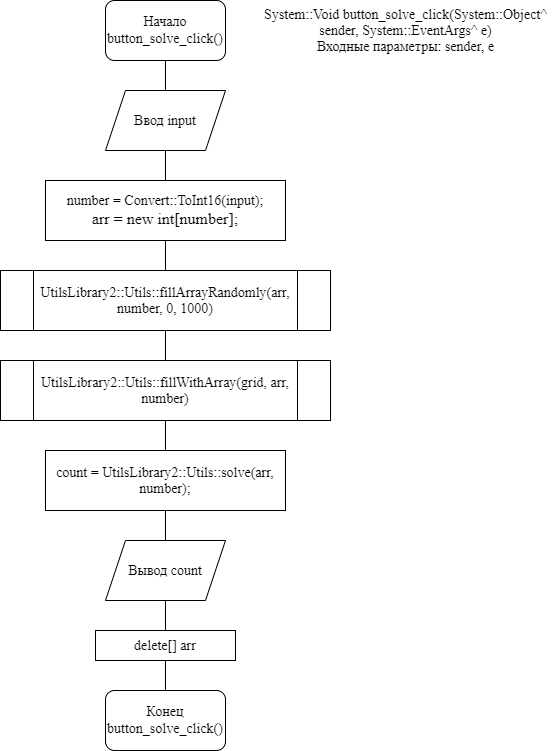


Рисунок 2 — Функция button\_solve\_click()

## 2.2 Исходный код программы

### 2.2.1 Библиотека

void UtilsLibrary2::Utils::fillArrayRandomly(int\* array, size\_t length, int min, int max)

{

// Найдено ли один элемент, делящийся на 7 без остатка

bool found = false;

for (size\_t i = 0; i < length; i++)

{

// Случайное число в промежутке [min;max]

int value = rand() % (max - min + 1) + min;

array[i] = value;

if (value % 7 == 0) found = true;

}

if (!found)

{

// Принудительно устанавливаем значение

array[length / 2] = 7;

}

}

void UtilsLibrary2::Utils::fillWithArray(DataGridView^ grid, int\* array, size\_t length)

{

grid->ColumnCount = length;

grid->RowCount = 2;

for (size\_t i = 0; i < length; i++)

{

grid->Rows[0]->Cells[i]->Value = "[" + i + "]";

grid->Rows[1]->Cells[i]->Value = array[i];

}

int sum = 0;

for (int i = 0; i < grid->ColumnCount; i++)

sum += grid->Columns[i]->Width;

if (sum > 410)

grid->Width = 410;

else

grid->Width = sum;

}

size\_t UtilsLibrary2::Utils::solve(int\* array, size\_t length)

{

size\_t count = 0;

for (size\_t i = 0; i < length - 1; i++)

{

int delta = array[i] - array[i + 1];

// результат оператора % будет неверным, если delta < 0;

// остаток от деления всегда положителен с математической точки зрения

// и записывается в виде (a % b + b) % b

if ((delta % 7 + 7) % 7 != 0) count++;

}

return count;

}

### 2.2.2 Основная программа

System::Void button\_solve\_click(System::Object^ sender, System::EventArgs^ e)

{

String^ input = Interaction::InputBox("Введите количество элементов массива", "Ввод", "10", -1, -1);

int number = Convert::ToInt16(input);

int\* arr = new int[number];

UtilsLibrary2::Utils::fillArrayRandomly(arr, number, 0, 1000);

UtilsLibrary2::Utils::fillWithArray(grid, arr, number);

size\_t count = UtilsLibrary2::Utils::solve(arr, number);

MessageBox::Show("Количество пар соседних элементов, разница\nмежду которыми НЕ кратна 7: " +

Convert::ToString(count), "Вывод", MessageBoxButtons::OK, essageBoxIcon::Information);

delete[] arr;

}

# 3 Заключение

Техническое задание были успешно реализовано на языке программирования Visual C++. Программа работает корректно, полученные результаты, представленные на рисунках ниже, верны.

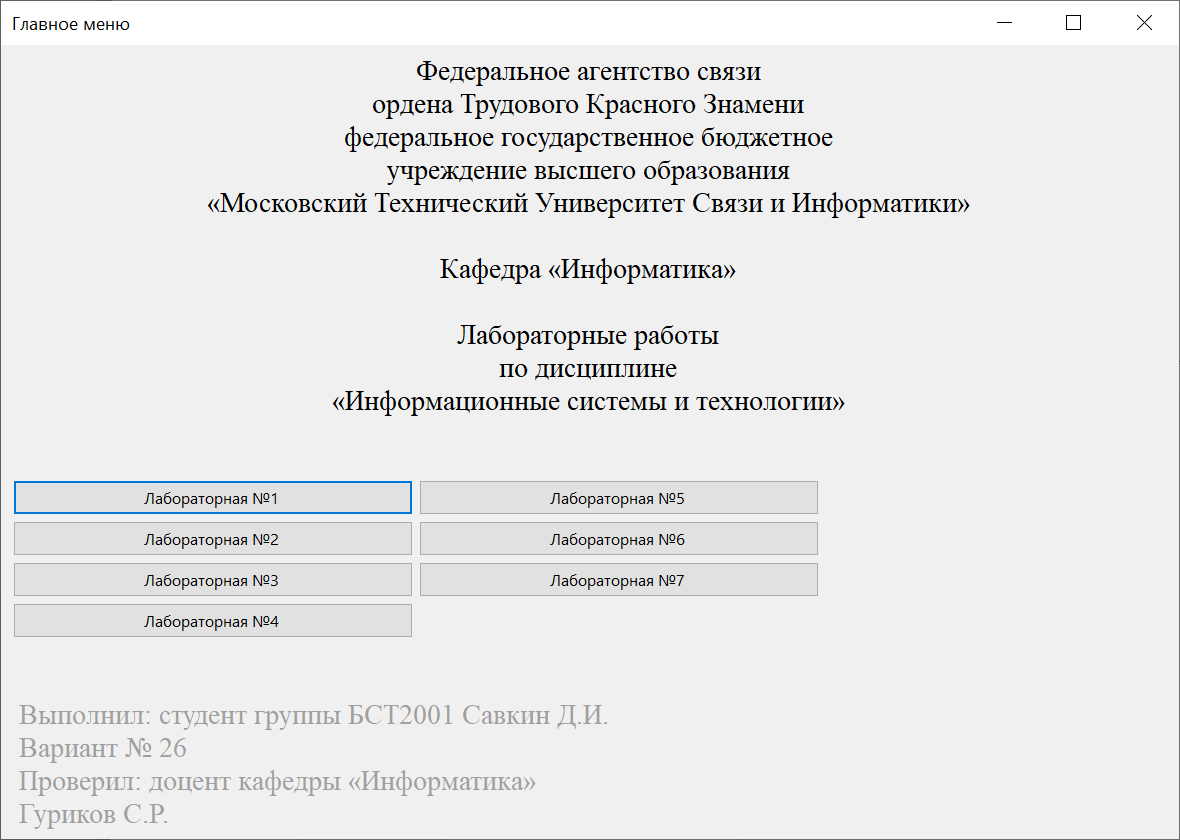


Рисунок 3 — Главное меню программы

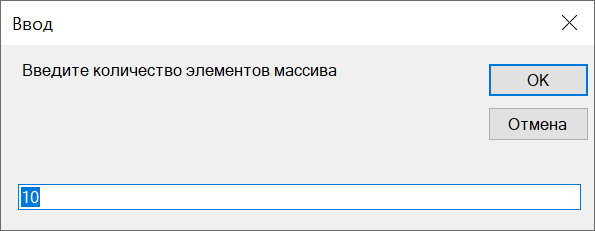


Рисунок 4 — Организация ввода размера массива

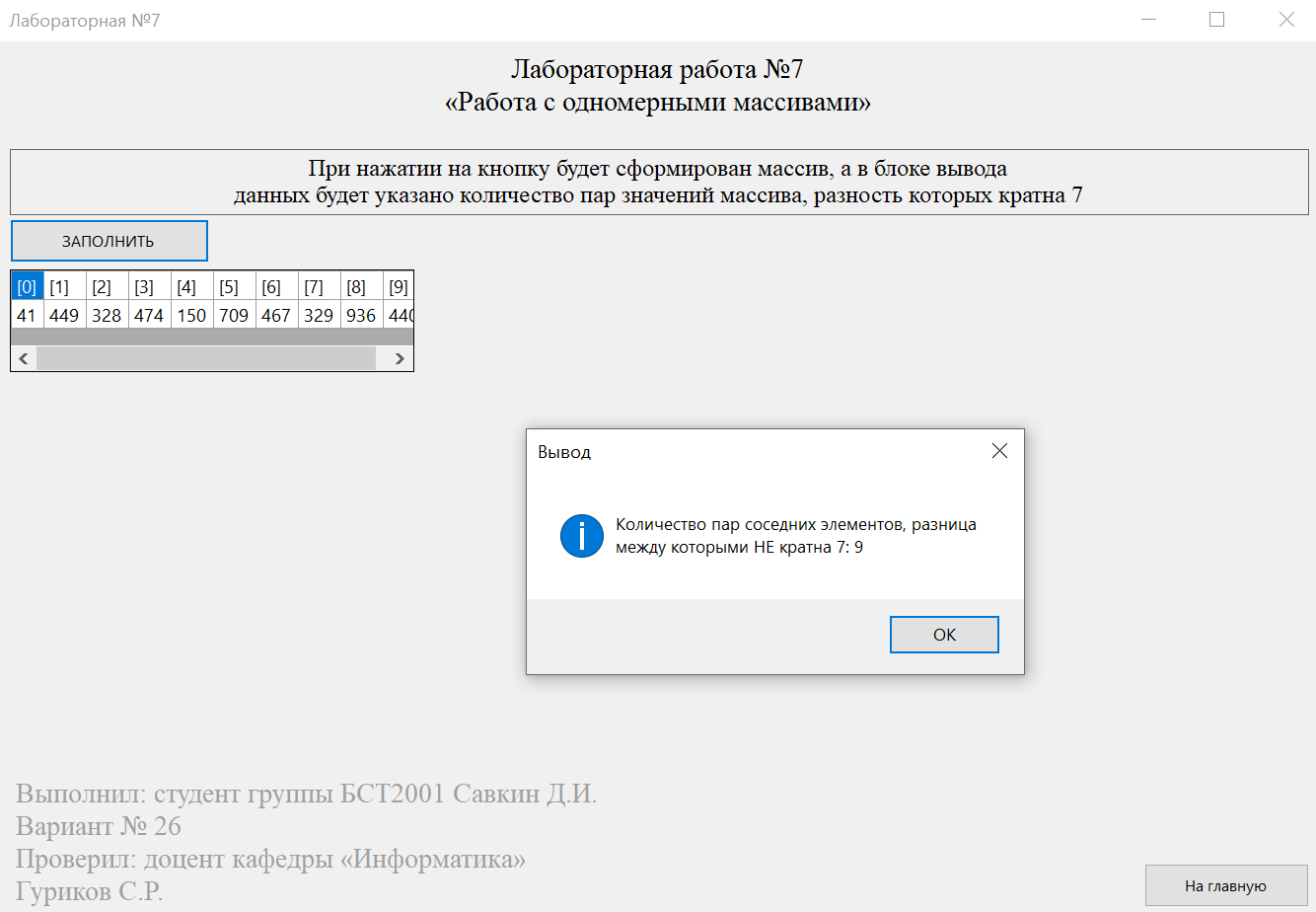


Рисунок 5 — Результат работы программы