ИНДИВИДУАЛЬНОЕ ЗАДАНИЕ

на прохождение учебной практики обучающегося направления

подготовки/специальности 09.03.01

1. Фамилия, имя, отчество обучающегося: Круглов Арсений Игоревич

1. Группа: 4143
2. Тема индивидуального задания: изучение методов сортировки структур данных;

совершенствование навыков процедурного программирования на языке *C*/*C*++

при решении задач обработки статических и динамических массивов; разработка

пользовательских интерфейсов.

1. Исходные данные:

- разработать математическую модель описания поставленной задачи и структурировать её с целью достижения требуемого решения;

- используя технологию процедурного программирования реализовать заданный

метод сортировки и применить его для указанных фрагментов числовой матрицы

в соответствии с индивидуальным заданием;

- разработать пользовательский интерфейс, позволяющий одновременно наблюдать

на экране исходную и результирующую матрицу.

1. Содержание отчетной документации:
   1. индивидуальное задание;
   2. отчёт, включающий в себя:
   * титульный лист (отчет по практике);
   * материалы о выполнении индивидуального задания;
   * выводы по результатам практики;
   * список использованных источников.
   1. отзыв руководителя от профильной организации (при прохождении практики в профильной организации).
2. Срок представления отчета на кафедру: «\_06\_»\_\_\_\_\_09\_\_\_\_\_\_\_\_2022 г.

Руководитель практики

доцент, канд. техн. наук, доцент Л.Н. Бариков

должность, уч. степень, звание подпись, дата инициалы, фамилия

Задание принял к исполнению:

Обучающийся

А.И. Круглов

дата подпись инициалы, фамилия

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное автономное образовательное учреждение высшего образования

«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

(ГУАП)

Кафедра 43 Компьютерных технологий и программной инженерии

(наименование)

ОТЧЁТ ПО ПРАКТИКЕ

ЗАЩИЩЁН С ОЦЕНКОЙ

Руководитель

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| доцент, канд. техн. наук, доцент |  |  |  | Л.Н. Бариков |
| должность, уч. степень, звание |  | подпись, дата |  | инициалы, фамилия |

ОТЧЁТ ПО ПРАКТИКЕ

|  |  |  |
| --- | --- | --- |
| вид практики | учебная | |
| тип практики | вычислительная | |
| на тему индивидуального задания | | изучение методов сортировки структур данных; |
| совершенствование навыков процедурного программирования на языке *C*/*C*++ | | | |
| при решении задач обработки статических и динамических массивов. | | | |

|  |  |
| --- | --- |
| выполнен | Кругловым Арсением Игоревичем |
| фамилия, имя, отчество обучающегося в творительном падеже | |

|  |  |  |  |
| --- | --- | --- | --- |
| по направлению подготовки | 09.03.01 |  | Информатика и вычислительная техника |
|  | код |  | наименование направления |
|  | | | |
| наименование направления | | | |
| направленности | 02 |  | Вычислительные машины, комплексы, |
|  | код |  | наименование направленности |
| системы и сети | | | |
| наименование направленности | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Обучающийся группы № | 4143 |  |  |  | А.И. Круглов |
|  | номер |  | подпись, дата |  | инициалы, фамилия |

Санкт–Петербург 2022

***Цель работы:*** *изучение методов сортировки структур данных; совершенствование навыков процедурного программирования на языке C/C++ при решении задач обработки статических и динамических массивов. Разработка пользовательского интерфейса.*

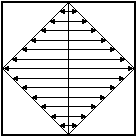
***Задание на программирование:*** *используя технологию процедурного программирования реализовать заданный метод сортировки и применить его для указанных фрагментов числовой матрицы в соответствии с индивидуальным заданием. Разработать пользовательский интерфейс, позволяющий:*

*- осуществлять ввод исходных данных;*

*- наблюдать на экране монитора одновременно исходную и отсортированную матрицу.*

***Метод сортировки:*** *Быстрая сортировка по убыванию.*

***Области сортировки элементов матриц:***



***Математическая модель***

**Сортировка матрицы**

Сортировка матрицы происходит следующим образом: для каждого фрагмента области сортировки матрицы создаётся массив, в который записываются соответствующие значения элементов матрицы. Полученный массив сортируется при помощи функции *QuickSort*, полученные значения записываются в соответствующие элементы результирующей матрицы, которая перед этим была заполнена нулями.

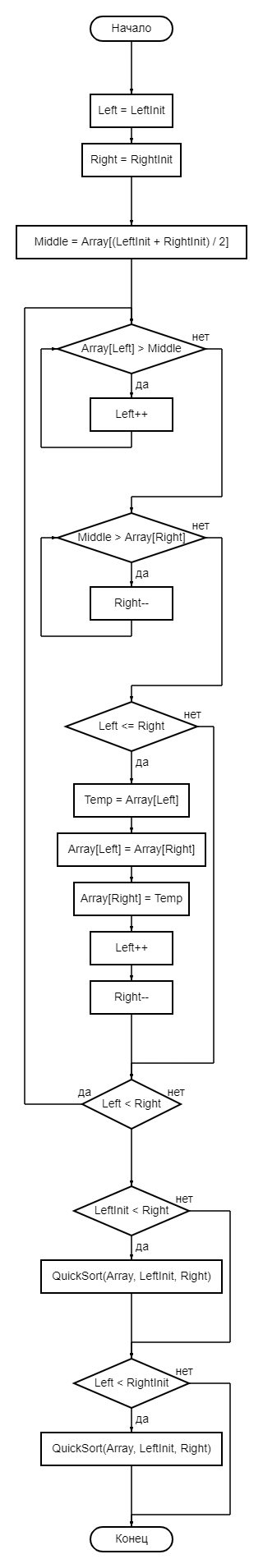
**Быстрая сортировка по убыванию**

Исходный массив делится на две части – правую и левую. Для конкретности в приводимом примере деление производится относительно элемента, стоящего посредине массива.

Затем, в левую относительно него часть массива помещаются все элементы массива, значения которых больше значения этого среднего элемента, а в правую – значения всех элементов массива, которые меньше значения среднего.

После этого элемент, относительно которого проводились все действия, оказывается на своём месте, и его больше не трогаем.

Далее повторяем указанные действия отдельно для левой и правой относительно него частей массива. В каждой из них находим её средний по номеру элемент, и повторяем все указанные действия относительно этого элемента. Так продолжается до тех пор, пока очередная обрабатываемая часть содержит более одного элемента. В результате повторения этих действий массив оказывается упорядоченным.

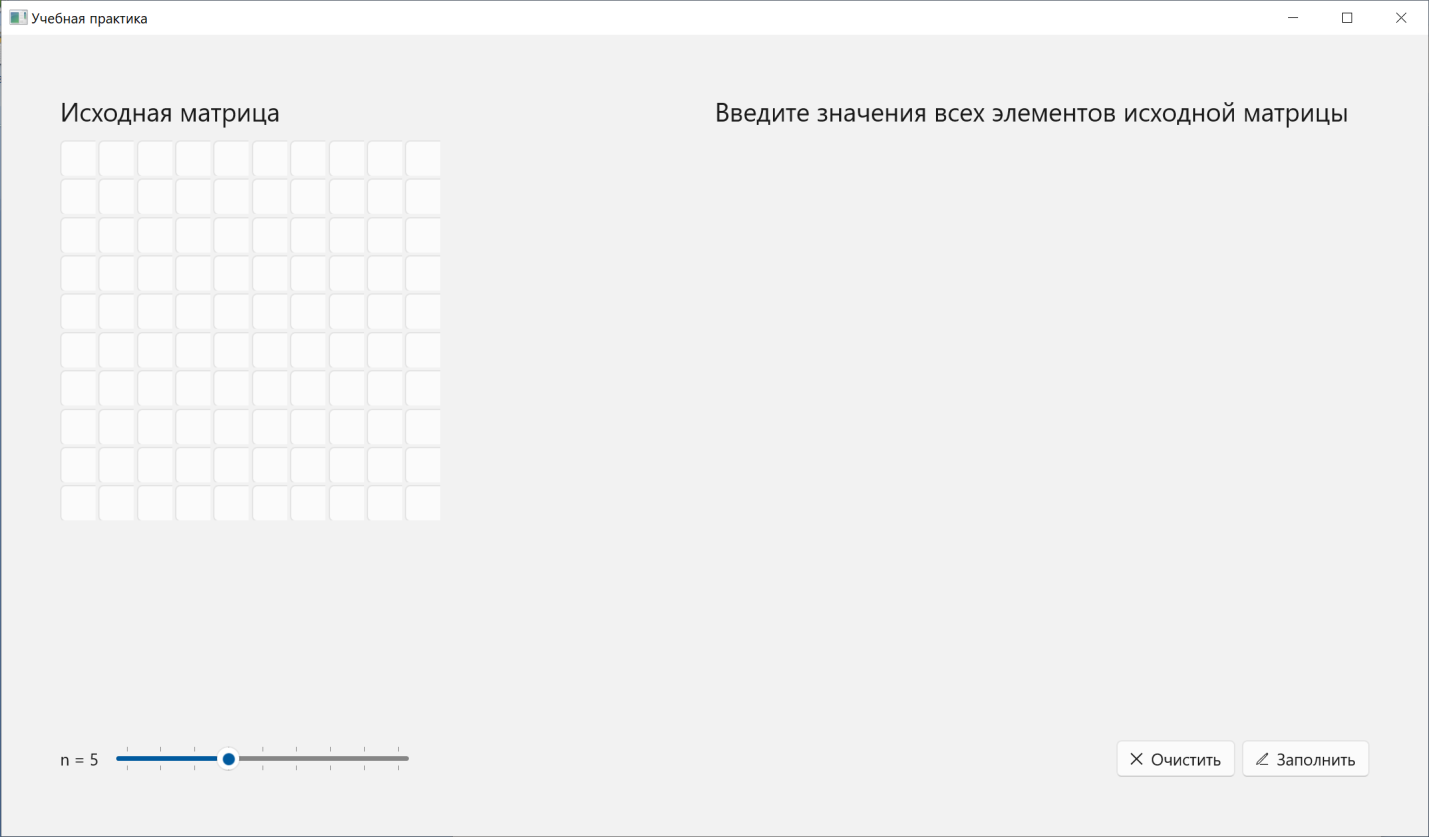


***Пользовательский интерфейс***

**WinUI 3**

Фреймворк WinUI 3 был выбран из-за его повсеместного использования в современных приложения Microsoft и приятного внешнего вида. Также в Visual Studio 2022 присутствует множество инструментов для работы с данным фреймворком, что значительно упрощает его использование.

**Обзор**

****

Элементы пользовательского интерфейса:

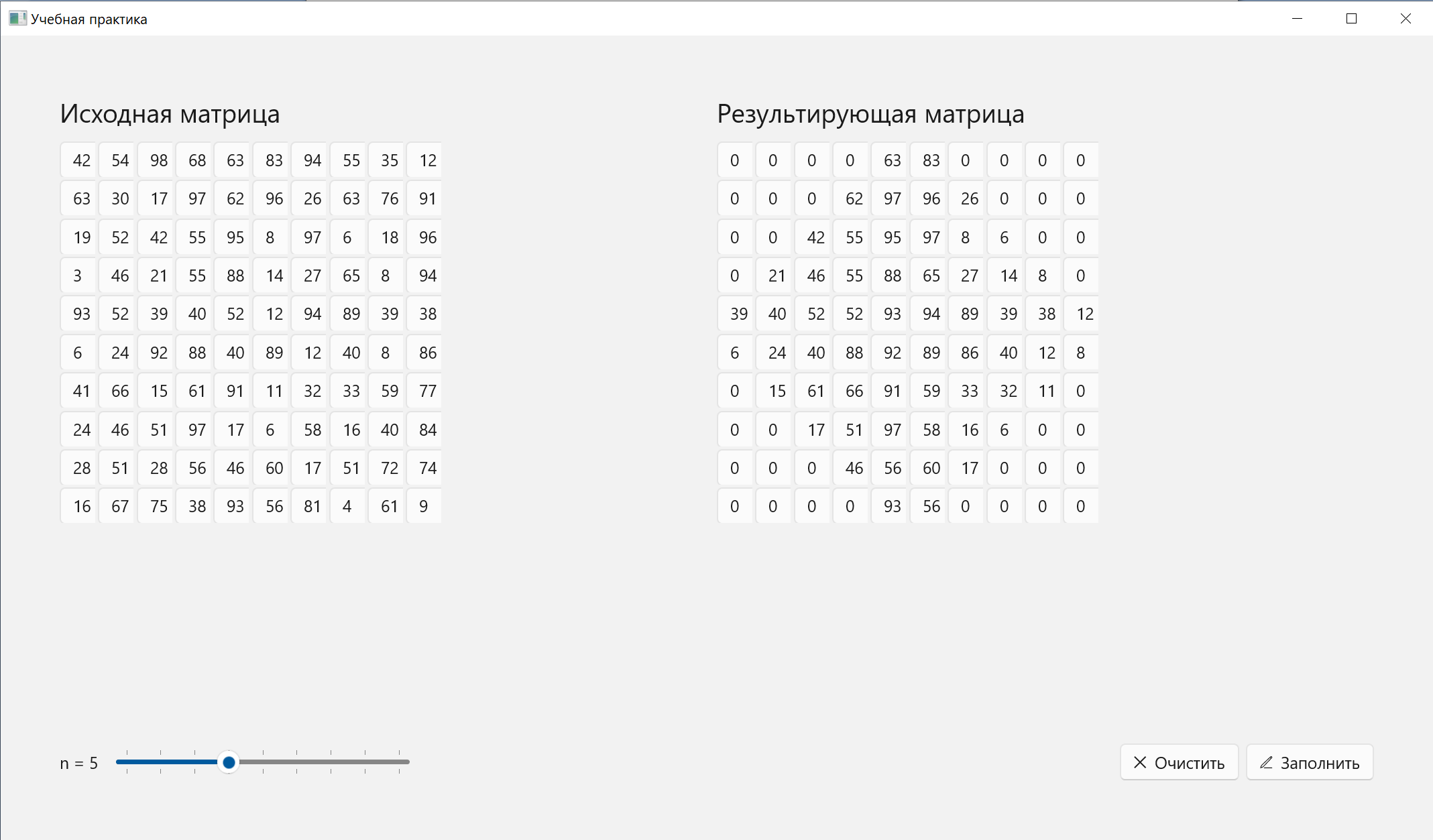
- Ползунок – Позволяет пользователь устанавливать значение *n* в диапазоне от 2 до 10.

- Кнопка «Очистить» – Опустошает исходную матрицу.

- Кнопка «Заполнить» – Заполняет исходную матрицу случайными целыми значениями от 1 до 99.

- Исходная матрица – Поля ввода значений элементов исходной матрицы.

- Результирующая матрица – Если исходная матрица заполнена, её отсортированная соответственно заданию версия. В ином случае – надпись «Введите значения всех элементов исходной матрицы».



Сортировка происходит после каждого изменения исходной матрицы, что позволяет пользователю сразу же видеть результат внесённых изменений.

***Исходный код***

Разметка *MainWindow.xaml*

<Window>

<Grid Padding="50" Background="{ThemeResource SystemChromeMediumLowColor}">

<Grid ColumnDefinitions="\*, \*">

<StackPanel Orientation="Vertical" Spacing="10" Grid.Column="0">

<TextBlock FontSize="22">Исходная матрица</TextBlock>

<StackPanel x:Name="Matrix\_Input\_Element" Spacing="2.5" />

</StackPanel>

<StackPanel Orientation="Vertical" Spacing="10" Grid.Column="1">

<TextBlock FontSize="22" x:Name="Matrix\_Output\_Title"></TextBlock>

<StackPanel x:Name="Matrix\_Output\_Element" Spacing="2.5" />

</StackPanel>

</Grid>

<Grid VerticalAlignment="Bottom">

<StackPanel Orientation="Horizontal" Spacing="15">

<TextBlock VerticalAlignment="Center" x:Name="nValue">n = 3</TextBlock>

<Slider VerticalAlignment="Center" Width="250" Minimum="2" Maximum="10" Value="3" TickFrequency="1" TickPlacement="Outside" x:Name="Slider" ValueChanged="Slider\_ValueChanged" />

</StackPanel>

<StackPanel Orientation="Horizontal" Spacing="5" HorizontalAlignment="Right">

<Button Click="Input\_Matrix\_Empty">

<StackPanel Orientation="Horizontal" Spacing="7.5">

<FontIcon FontFamily="Segoe MDL2 Assets" Glyph="&#xE8BB;" FontSize="10" VerticalAlignment="Center"/>

<TextBlock>Очистить</TextBlock>

</StackPanel>

</Button>

<Button Click="Input\_Matrix\_Fill">

<StackPanel Orientation="Horizontal" Spacing="7.5">

<FontIcon FontFamily="Segoe MDL2 Assets" Glyph="&#xEC87;" FontSize="10" VerticalAlignment="Center"/>

<TextBlock>Заполнить</TextBlock>

</StackPanel>

</Button>

</StackPanel>

</Grid>

</Grid>

</Window>

Программа *MainWindow.xaml.cpp*

#include "pch.h"

#include "MainWindow.xaml.h"

#if \_\_has\_include("MainWindow.g.cpp")

#include "MainWindow.g.cpp"

#include <string>

#endif

using namespace winrt;

using namespace Microsoft::UI::Xaml;

void QuickSort(TArray Array, int LeftInit, int RightInit)

{

int Left = LeftInit,

Right = RightInit;

TElement Middle, Temp;

Middle = Array[(LeftInit + RightInit) / 2];

do

{

while (Array[Left] > Middle)

Left++;

while (Middle > Array[Right])

Right--;

if (Left <= Right)

{

Temp = Array[Left];

Array[Left] = Array[Right];

Array[Right] = Temp;

Left++;

Right--;

}

}

while (Left < Right);

if (LeftInit < Right)

QuickSort(Array, LeftInit, Right);

if (Left < RightInit)

QuickSort(Array, Left, RightInit);

}

void Empty(int& n, TMatrix Matrix)

{

for (int i = 0; i < 2 \* n; i++)

for (int j = 0; j < 2 \* n; j++)

Matrix[i][j] = 0;

}

void Process(int n, TMatrix Matrix\_Input, TMatrix Matrix\_Output)

{

for (int Row = 0; Row < 2 \* n; Row++)

{

{

TArray Group;

int GroupLength = 0;

for (int Column = n - 1; Column > (Row < n ? (n - Row - 2) : (Row - n - 1)); Column--)

Group[GroupLength++] = Matrix\_Input[Row][Column];

QuickSort(Group, 0, GroupLength - 1);

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

Matrix\_Output[Row][n - i - 1] = Group[i];

}

{

TArray Group;

int GroupLength = 0;

for (int Column = n; Column < (Row < n ? (n + Row + 1) : (3 \* n - Row)); Column++)

Group[GroupLength++] = Matrix\_Input[Row][Column];

QuickSort(Group, 0, GroupLength - 1);

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

Matrix\_Output[Row][n + i] = Group[i];

}

}

}

namespace winrt::WinUI::implementation

{

MainWindow::MainWindow()

{

InitializeComponent();

this->Title(L"Учебная практика");

this->Output\_Error();

}

int32\_t MainWindow::MyProperty()

{

throw hresult\_not\_implemented();

}

void MainWindow::MyProperty(int32\_t)

{

throw hresult\_not\_implemented();

}

// Main

void MainWindow::Sort()

{

for (int Row = 0; Row < 2 \* n; Row++)

for (int Column = 0; Column < 2 \* n; Column++)

{

Controls::NumberBox NumberBox = Matrix\_Input\_Element().Children().GetAt(Row).as<Controls::StackPanel>().Children().GetAt(Column).as<Controls::NumberBox>();

if (NumberBox.Text() != L"")

{

Matrix\_Input[Row][Column] = NumberBox.Value();

}

else

{

this->Output\_Error();

return;

}

}

Empty(this->n, this->Matrix\_Output);

Process(this->n, this->Matrix\_Input, this->Matrix\_Output);

this->Output\_Success();

}

// Input

void MainWindow::Slider\_ValueChanged(IInspectable const&, Controls::Primitives::RangeBaseValueChangedEventArgs const& Arguments)

{

this->n = Arguments.NewValue();

this->Input\_Matrix\_DrawEmpty();

this->Input\_UpdateNValue();

this->Output\_Error();

}

void MainWindow::NumberBox\_ValueChange(IInspectable const&, Controls::NumberBoxValueChangedEventArgs const&)

{

this->Sort();

}

void MainWindow::Input\_Matrix\_DrawEmpty()

{

Controls::StackPanel MatrixElement = Matrix\_Input\_Element();

MatrixElement.Children().Clear();

for (int Row = 0; Row < 2 \* this->n; Row++)

{

Controls::StackPanel StackPanel;

StackPanel.Orientation(Controls::Orientation::Horizontal);

StackPanel.Spacing(2.5);

for (int Column = 0; Column < 2 \* this->n; Column++)

{

Controls::NumberBox NumberBox;

NumberBox.Width(30);

NumberBox.Height(30);

NumberBox.ValueChanged({ this, &MainWindow::NumberBox\_ValueChange });

StackPanel.Children().Append(NumberBox);

}

MatrixElement.Children().Append(StackPanel);

}

}

void MainWindow::Input\_Matrix\_Fill(IInspectable const&, RoutedEventArgs const&)

{

Controls::StackPanel MatrixElement = Matrix\_Input\_Element();

for (int Row = 0; Row < 2 \* this->n; Row++)

for (int Column = 0; Column < 2 \* this->n; Column++)

MatrixElement.Children().GetAt(Row).as<Controls::StackPanel>().Children().GetAt(Column).as<Controls::NumberBox>().Value(rand() % 99 + 1);

this->Sort();

}

void MainWindow::Input\_Matrix\_Empty(IInspectable const&, RoutedEventArgs const&)

{

Controls::StackPanel MatrixElement = Matrix\_Input\_Element();

for (int Row = 0; Row < 2 \* this->n; Row++)

for (int Column = 0; Column < 2 \* this->n; Column++)

MatrixElement.Children().GetAt(Row).as<Controls::StackPanel>().Children().GetAt(Column).as<Controls::NumberBox>().Value(NAN);

this->Output\_Error();

}

void MainWindow::Input\_UpdateNValue()

{

nValue().Text(L"n = " + to\_hstring(this->n));

}

// Output

void MainWindow::Output\_Error()

{

Matrix\_Output\_Title().Text(L"Введите значения всех элементов исходной матрицы");

Matrix\_Output\_Element().Children().Clear();

}

void MainWindow::Output\_Success()

{

if (Matrix\_Output\_Element().Children().Size() == 0)

{

Matrix\_Output\_Title().Text(L"Результирующая матрица");

this->Output\_Matrix\_DrawEmpty();

}

this->Output\_Matrix\_Fill();

}

void MainWindow::Output\_Matrix\_DrawEmpty()

{

Controls::StackPanel MatrixElement = Matrix\_Output\_Element();

MatrixElement.Children().Clear();

for (int Row = 0; Row < 2 \* this->n; Row++)

{

Controls::StackPanel StackPanel;

StackPanel.Orientation(Controls::Orientation::Horizontal);

StackPanel.Spacing(2.5);

for (int Column = 0; Column < 2 \* this->n; Column++)

{

Controls::NumberBox NumberBox;

NumberBox.Width(30);

NumberBox.Height(30);

StackPanel.Children().Append(NumberBox);

}

MatrixElement.Children().Append(StackPanel);

}

}

void MainWindow::Output\_Matrix\_Fill()

{

Controls::StackPanel MatrixElement = Matrix\_Output\_Element();

for (int Row = 0; Row < 2 \* this->n; Row++)

for (int Column = 0; Column < 2 \* this->n; Column++)

MatrixElement.Children().GetAt(Row).as<Controls::StackPanel>().Children().GetAt(Column).as<Controls::NumberBox>().Value(this->Matrix\_Output[Row][Column]);

}

}

***Выводы по результатам практики***

В процессе выполнения практической работы была разработана математическую модель описания поставленной задачи, которая была структурирована с целью достижения требуемого решения.

Используя технологию процедурного программирования, был реализован заданный метод сортировки, который был применен для указанных фрагментов числовой матрицы в соответствии с индивидуальным заданием.

Был разработан пользовательский интерфейс, позволяющий одновременно наблюдать на экране исходную и результирующую матрицу.

***Список использованной литературы***

1. Библиотека пользовательского интерфейса Windows (WinUI) 3 // Microsoft URL: https://docs.microsoft.com/ru-ru/windows/apps/winui/winui3/ (дата обращения: 04.09.2022).
2. ГОСТ 2.105-2019. Общие требования к текстовым документам, (дата обращения 04.09.2022)
3. ГОСТ 19.701-90. Схемы алгоритмов, программ, данных и систем. (дата обращения 04.09.2022)