Федеральное государственное образовательное бюджетное учреждение

высшего образования «Финансовый университет при Правительстве Российской Федерации»

**Колледж информатики и программирования**

**Специальность 09.02.03 Программирование в компьютерных системах**

**ПМ.03 Участие в интеграции программных модулей**

**МДК.03.01. Технология разработки программного обеспечения**

ПРАКТИЧЕСКАЯ РАБОТА №8

Выполнил:

студент группы: 3ПКС-115

Черников А. В.

Проверила:

Морозова М. В.

Оценка\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Москва 2018

**Практическая работа №8**

**Задача №1:**

**Условие задачи:**

Доработать программу с графическим итерфейсом для того чтобы в ней выдавались ошибки при различных исключительных ситуациях а также проверить результат работы.

**Код программы:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace TestClient

{

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

}

private void Button1\_Click(object sender, RoutedEventArgs e)

{

this.textBlock1.Text = "Initiating test sequence: " + DateTime.Now.ToLongTimeString();

SwitchDevices.Switch sd = new SwitchDevices.Switch();

try

{if (sd.DisconnectPowerGenerator() == SwitchDevices.SuccessFailureResult.Fail)

{this.textBlock1.Text += "\nStep 1: Failed to disconnect power generation system";}

else

{this.textBlock1.Text += "\nStep 1: Successfully disconnected power generation system";}

}

catch (SwitchDevices.PowerGeneratorCommsException ex)

{this.textBlock1.Text += "\n\*\*\*Exception in step 1: " + ex.Message;}

try

{switch (sd.VerifyPrimaryCoolantSystem())

{case SwitchDevices.CoolantSystemStatus.OK:

this.textBlock1.Text += "\nStep 2: Primary coolant system OK";

break;

case SwitchDevices.CoolantSystemStatus.Check:

this.textBlock1.Text += "\nStep 2: Primary coolant system requires manual check";

break;

case SwitchDevices.CoolantSystemStatus.Fail:

this.textBlock1.Text += "\nStep 2: Problem reported with primary coolant system";

break;}

}

catch (SwitchDevices.CoolantPressureReadException ex)

{this.textBlock1.Text += "\n\*\*\*Exception in step 2: " + ex.Message;}

catch (SwitchDevices.CoolantTemperatureReadException ex)

{this.textBlock1.Text += "\n\*\*\*Exception in step 2: " + ex.Message;}

try

{switch (sd.VerifyBackupCoolantSystem())

{case SwitchDevices.CoolantSystemStatus.OK:

this.textBlock1.Text += "\nStep 3: Backup coolant system OK";

break;

case SwitchDevices.CoolantSystemStatus.Check:

this.textBlock1.Text += "\nStep 3: Backup coolant system requires manual check";

break;

case SwitchDevices.CoolantSystemStatus.Fail:

this.textBlock1.Text += "\nStep 3: Backup reported with primary coolant system";

break;}

}

catch (SwitchDevices.CoolantPressureReadException ex)

{this.textBlock1.Text += "\n\*\*\*Exception in step 3: " + ex.Message;}

catch (SwitchDevices.CoolantTemperatureReadException ex)

{this.textBlock1.Text += "\n\*\*\*Exception in step 3: " + ex.Message;}

try

{this.textBlock1.Text += "\nStep 4: Core temperature before shutdown: " + sd.GetCoreTemperature();}

catch (SwitchDevices.CoreTemperatureReadException ex)

{this.textBlock1.Text += "\n\*\*\*Exception in step 4: " + ex.Message;}

try

{if (sd.InsertRodCluster() == SwitchDevices.SuccessFailureResult.Success)

{this.textBlock1.Text += "\nStep 5: Control rods successfully inserted";}

else

{this.textBlock1.Text += "\nStep 5: Control rod insertion failed";}

}

catch (SwitchDevices.RodClusterReleaseException ex)

{this.textBlock1.Text += "\n\*\*\*Exception in step 5: " + ex.Message;}

try

{this.textBlock1.Text += "\nStep 6: Core temperature after shutdown: " + sd.GetCoreTemperature();}

catch (SwitchDevices.CoreTemperatureReadException ex)

{this.textBlock1.Text += "\n\*\*\*Exception in step 6: " + ex.Message;}

try

{this.textBlock1.Text += "\nStep 7: Core radiation level after shutdown: " + sd.GetRadiationLevel();}

catch (SwitchDevices.CoreRadiationLevelReadException ex)

{this.textBlock1.Text += "\n\*\*\*Exception in step 7: " + ex.Message;}

try

{sd.SignalShutdownComplete();

this.textBlock1.Text += "\nStep 8: Broadcasting shutdown complete message";}

catch (SwitchDevices.SignallingException ex)

{this.textBlock1.Text += "\n\*\*\*Exception in step 8: " + ex.Message;}

this.textBlock1.Text += "\nTest sequence complete: " + DateTime.Now.ToLongTimeString();

}

}

}

**Результаты:**

Результат задачи №1 представлен на рисунке 1, 2

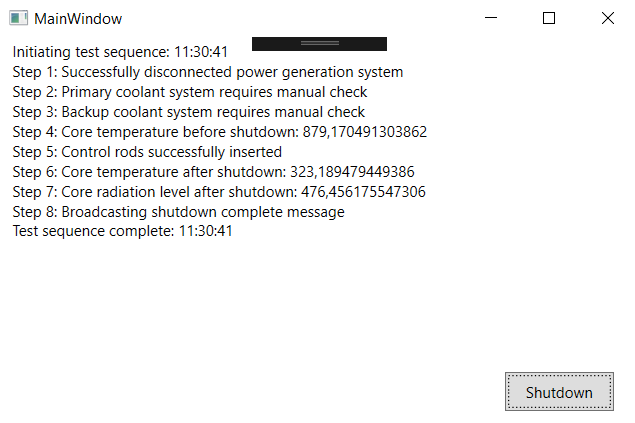


Рисунок 1 (без ошибок)

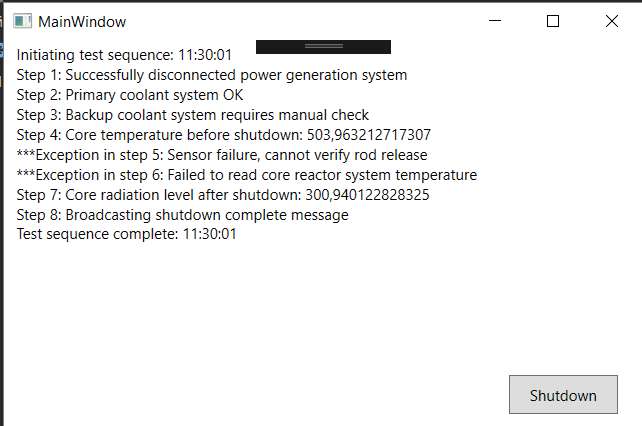


Рисунок 2 (с ошибками)

**Задача №2:**

**Условие задачи:**

Доработать программу c графическим интерфейсом которая перемножает две матрицы для того, чтобы в ней выдавались ошибки при некоторых ситуациях.

**Код программы:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace MatrixMultiplication

{

public partial class MainWindow : Window

{

double[,] matrix1;

double[,] matrix2;

double[,] result;

public MainWindow()

{

InitializeComponent();

}

private void MatrixDimensions\_Changed(object sender, SelectionChangedEventArgs e)

{

int m1rows = 1;

int m1columns\_m2rows = 1;

int m2columns = 1;

if (matrix1width != null) m1columns\_m2rows = matrix1width.SelectedIndex +1;

if (matrix1height != null) m1rows = matrix1height.SelectedIndex + 1;

if (matrix2width != null) m2columns = matrix2width.SelectedIndex + 1;

matrix1 = new double[m1columns\_m2rows, m1rows];

matrix2 = new double[m2columns, m1columns\_m2rows];

result = new double[m2columns,m1rows];

InitializeGrid(grid1, matrix1);

InitializeGrid(grid2, matrix2);

InitializeGrid(grid3, result);

}

private void InitializeGrid(Grid grid, double[,] matrix)

{

if (grid != null)

{

grid.Children.Clear();

grid.ColumnDefinitions.Clear();

grid.RowDefinitions.Clear();

int columns = matrix.GetLength(0);

int rows = matrix.GetLength(1);

for (int x = 0; x < columns; x++)

{

grid.ColumnDefinitions.Add(new ColumnDefinition() { Width = new GridLength(1, GridUnitType.Star), });

}

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

{

grid.RowDefinitions.Add(new RowDefinition() { Height = new GridLength(1, GridUnitType.Star), });

}

for (int x = 0; x < columns; x++)

{

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

{

double cell = (double)matrix[x, y];

TextBox t = new TextBox();

t.Text = cell.ToString();

t.VerticalAlignment = System.Windows.VerticalAlignment.Center;

t.HorizontalAlignment = System.Windows.HorizontalAlignment.Center;

t.SetValue(Grid.RowProperty, y);

t.SetValue(Grid.ColumnProperty, x);

grid.Children.Add(t);

}

}

}

}

private void ButtonCalculate\_Click(object sender, RoutedEventArgs e)

{

GetValuesFromGrid(grid1, matrix1);

GetValuesFromGrid(grid2, matrix2);

try

{

result = Matrix.MatrixMultiply(matrix1, matrix2);

throw new ArgumentException();

}

catch(ArgumentException ex)

{

MessageBox.Show(ex.Message);

}

InitializeGrid(grid3, result);

}

private void GetValuesFromGrid(Grid grid, double[,] matrix)

{

int columns = grid.ColumnDefinitions.Count;

int rows = grid.RowDefinitions.Count;

if (columns != matrix.GetLength(0)) throw new ArgumentException("Grid and matrix have different number of columns");

if (rows != matrix.GetLength(1)) throw new ArgumentException("Grid and matrix have different number of rows");

for (int c = 0; c < grid.Children.Count; c++)

{

TextBox t = (TextBox)grid.Children[c];

int row = Grid.GetRow(t);

int column = Grid.GetColumn(t);

matrix[column, row] = double.Parse(t.Text);

}

}

}

}

namespace MatrixMultiplication

{

static class Matrix

{

public static double[,] MatrixMultiply(double[,] matrix1, double[,] matrix2)

{

try

{

if (matrix1.GetLength(0) != matrix2.GetLength(1))

{

throw new ArgumentException();

}

}

catch (ArgumentException)

{

MessageBox.Show("Число столбцов матрицы 1 и число строк матрицы 2");

}

int m1columns\_m2rows = matrix1.GetLength(0);

int m1rows = matrix1.GetLength(1);

int m2columns = matrix2.GetLength(0);

double[,] result = new double[m2columns, m1rows];

try

{

for (int row = 0; row < m1rows; row++)

{

for (int column = 0; column < m2columns; column++)

{

double accumulator = 0;

for (int cell = 0; cell < m1columns\_m2rows; cell++)

{

if (matrix1[column, cell] < 0)

{

throw new ArgumentException(string.Format(" Matrix1 содержит неправильное значение в ячейке[{0}, {1}].", column+1, cell+1));

}

if (matrix2[column, cell] < 0)

{

throw new ArgumentException(string.Format(" Matrix2 содержит неправильное значение в ячейке[{0}, {1}].", column + 1, cell + 1));

}

accumulator += matrix1[cell, row] \* matrix2[column, cell];

}

result[column, row] = accumulator;

}

}

}

catch (ArgumentException ex)

{MessageBox.Show( ex.Message);}

return result;

}

}

}

**Результаты:**

Результат задачи №2 представлен на рисунке 3, 4

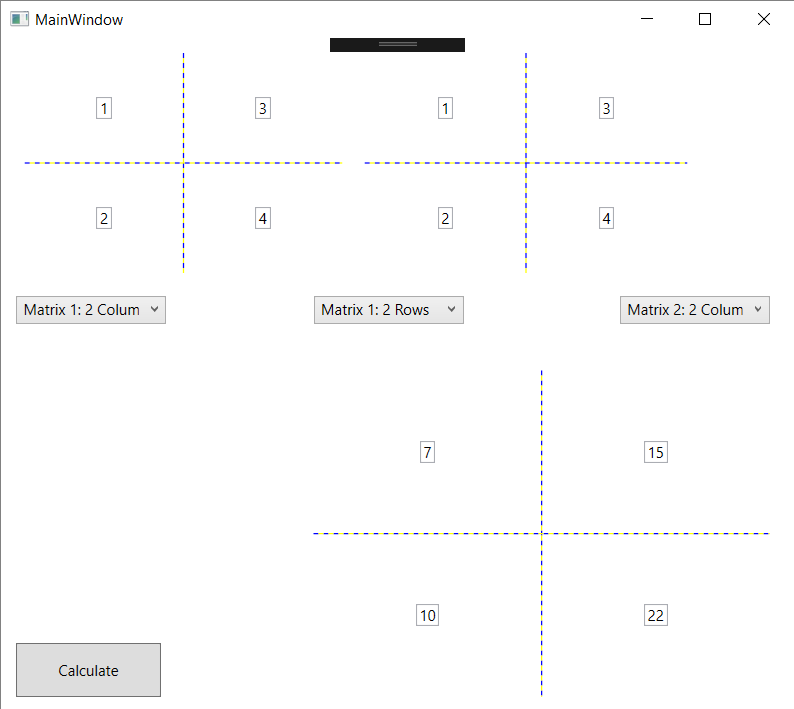


Рисунок 3 (без ошибок)

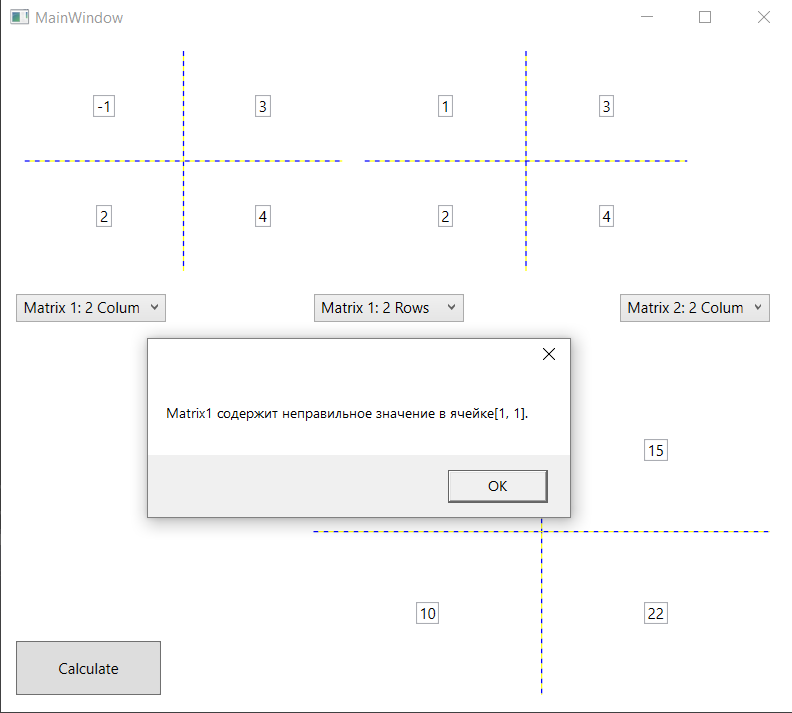


Рисунок 4 (с ошибками)

**Задача №3:**

**Условие задачи:**

Доработать программу c графическим интерфейсом которая перемножает два числа для того, чтобы в ней выдавались ошибки при переполнении разряда числа int.

**Код программы:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace IntegerArithmetic

{

public partial class MainWindow : Window

{

public MainWindow()

{

InitializeComponent();

}

private void Window\_Loaded(object sender, RoutedEventArgs e)

{

intOne.Text = int.MaxValue.ToString();

intTwo.Text = "2";

}

private void DoMultiply\_Click(object sender, RoutedEventArgs e)

{

int x; int y;

if (!int.TryParse(intOne.Text, out x))

{

labelAnswer.Content = "First value not an integer";

return;

}

if (!int.TryParse(intTwo.Text, out y))

{

labelAnswer.Content = "Second value not an integer";

return;

}

try

{

labelAnswer.Content = (x \* y).ToString();

throw new OverflowException();

}

catch (OverflowException ex)

{MessageBox.Show(ex.Message);}}

}

}

**Результаты:**

Результат задачи №3 представлен на рисунке 5

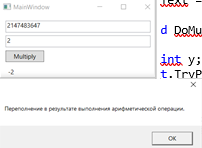


Рисунок 5 (с ошибками)